Codesign of a Jupyter Notebook for Wood Stove Emissions Analysis

by
Ryan Schofield

A THESIS

submitted to

Oregon State University

Honors College

in partial fulfillment of
the requirements for the
degree of

Honors Baccalaureate of Science in Mechanical Engineering
(Honors Scholar)

Presented November 28, 2023
Commencement June 2024

AN ABSTRACT OF THE THESIS OF

Ryan Schofield for the degree of Honors Baccalaureate of Science in Mechanical Engineering presented on November 28, 2023. Title: Codesign of a Jupyter Notebook for Wood Stove Emissions Analysis.

Abstract approved:____

Nordica MacCarty

Designs for improved cookstoves aim to decrease the detrimental environmental and health impacts that are associated with traditional biomass cooking practiced by nearly 40% of the global population. Programs to provide improved cookstoves are found worldwide, and regional testing centers support development of these through emissions testing services. Aprovecho Research Center is a non-profit organization that provides Laboratory Emissions Monitoring Systems (LEMS) to testing centers around the world to enable accurate and efficient emissions testing capabilities. Currently, emissions data are analyzed with a set of macro-enabled Excel sheets, but they are being converted to a Python-based system to enable greater functionality and data sharing. This project sought to create an appropriate user interface for this Python code that could be used by researchers globally. A Jupyter Notebook was chosen because of its flexible, user-friendly interface with live updating and ability to hide or display code as needed. The Jupyter Notebook that was developed reads data from Excel input sheets and outputs graphs and tables that are easy to analyze. While there is still work to test and improve the Jupyter Notebook with global users, initial testing and feedback indicates the Jupyter Notebook interface provides the automated, user-friendly analysis needed to support the LEMS system.

Keywords: Aprovecho, Cookstoves, Data Analysis, Jupyter Notebook, Python, User Interface

Corresponding e-mail address: schofier@oregonstate.edu

Codesign of a Jupyter Notebook for Wood Stove Emissions Analysis

by
Ryan Schofield

A THESIS

submitted to

Oregon State University

Honors College

in partial fulfillment of
the requirements for the
degree of

Honors Baccalaureate of Science in Mechanical Engineering
(Honors Scholar)

Presented November 28, 2023
Commencement June 2024

Honors Baccalaureate of Science in Mechanical Engineering project of Ryan Schofield presented on November 28, 2023.

APPROVED:

Nordica MacCarty, Mentor, representing MIME and HEST

Sam Bentson, Committee Member, representing Aprovecho

Cindy Grimm, Committee Member, representing MIME

Toni Doolen, Dean, Oregon State University Honors College

I understand that my project will become part of the permanent collection of Oregon State University, Honors College. My signature below authorizes release of my project to any reader upon request.

Ryan Schofield, Author

1. *Introduction*

Improved cookstoves have been a project since the 1970s and has only gained more momentum since then (Urmee et al., 2014). Programs to provide cleaner and more efficient stoves have been introduced in Africa, Asia, and Latin America (Urmee et al., 2014). Even though these programs have been introduced, there are still 2.4 billion people in the world that use traditional cookstoves (WHO, 2022). These tradition cookstoves lead to many health issues (WHO, 2022), as well as contributing to climate change (Alem et al., 2021) and deforestation (Bailis et al., 2015).

Regional testing centers were introduced with a purpose of being able to test stoves worldwide with international standards and work with stove developers in their area to improve cookstoves (Jetter, 2016). These testing centers are all over the world, and Aprovecho designs and distributes low-cost user-friendly emissions testing systems to these testing center to provide more efficient and accurate testing (Aprovecho 2021). It is important to remember that these regional testing centers are run by people all over the world with differing skill levels. That is why it is essential to create a tool that will provide automated result analysis with intuitive instructions. A Jupyter Notebook was co-designed with Aprovecho for that purpose.

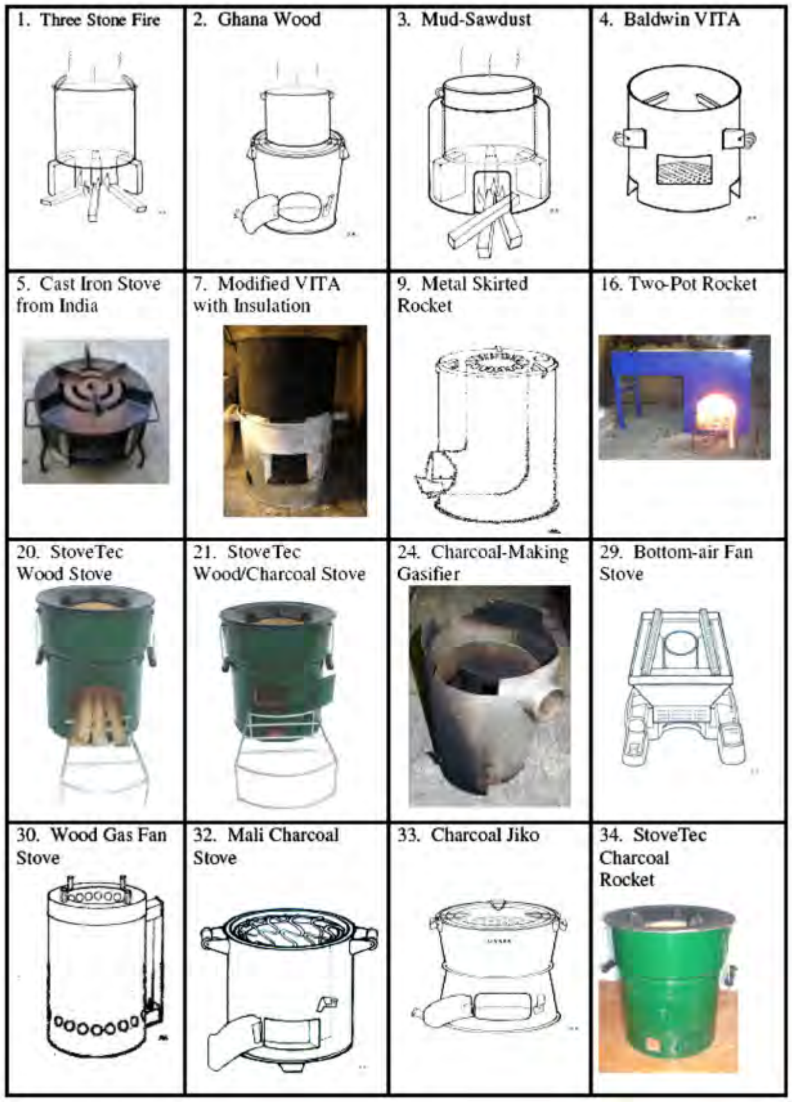
This Jupyter Notebook will take in Excel sheets with testing results, and output tables and graphs that allow for easier analysis of improved cookstove data. The Jupyter Notebook was designed with the idea to be intuitive so that it could be deployed all around the world to provide easier analysis of improved cookstoves data to all testing centers. The more accurate and efficient data analysis allows for better improved cookstoves to be designed faster and more efficiently.

This thesis will document the creation of this Jupyter Notebook. First, the importance of improving cookstoves, user experience design requirements, and who is impacted by improved cookstoves will be discussed. Next, a discussion of how the notebook was intended to be created and what the final notebook looks like. Finally, a discussion of how the Jupyter Notebook can be improved in the future.

2. *Background*

2.1. *Cookstoves*

Biomass cookstoves use fuel combustion to heat a cooking target. There are two types of traditional cookstoves: three-stone fire and built-in stoves (a modification of a three-stone fire) (Kshirsagar et al., 2014). There are 7 major categories of improved cookstoves: simple stoves without combustion chambers, stoves with rocket-type combustion chambers, gasifier stoves, fan-assisted stoves, charcoal-burning stoves, liquid/gas fuel stoves, and wood-burning stoves (MacCarty et al., 2010). Some cookstoves can be seen in figure 1 below.



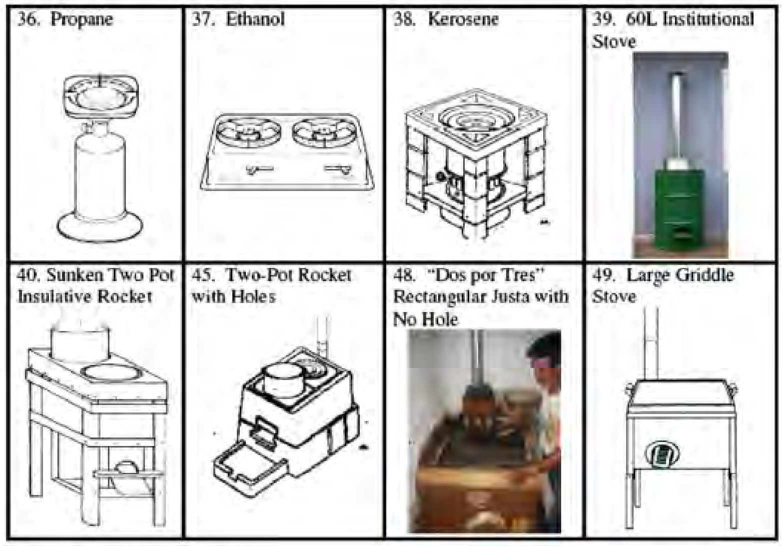


Figure 1. Cookstove examples (MacCarty et al., 2010).

2.2. *Value of Work in Improving Cookstoves*

There are 2.4 billion people in the world that use traditional cookstoves (WHO, 2022). As this affects a large population, there are multiple different organizations that are trying to help the situation, such as the Clean Cooking Alliance (CCA). The CCA was established in 2010 and has been working to make clean cooking available worldwide (CCA, 2021). There are over 9000 partners working with the CCA across the globe in the fields of cookstove design and manufacturer, cookstove distribution, fuel production, and fuel distribution (CCA, n.d).

2.2.1. *Environmental Concerns of Traditional Cookstoves*

Traditional cookstoves burn wood, charcoal, or other biomass fuels to produce heat, which perpetuated climate change. The combustion of biomass leads to the release of carbon dioxide (CO_2), black carbon, and other gasses and particulate matter (PM) into the air (Goldemberg et al, 2018). Carbon dioxide is one of the largest sources of carbon emissions (Alem et al., 2021) and black carbon increases snow and ice's ability to absorb solar radiation (MacCarty et al., 2008). In a study done in 2018, it was found that when 2.5 liters of water is boiled and then simmered for 30 minutes, up to 1.16 particle emission factor of black carbon is released depending on the stove type (MacCarty et al., 2008). Between the five stove types tested, there is a variety of emissions produced, with some stoves producing more emissions than others. For example, a charcoal cook stove had the worst or tied for the worst emissions in four out of the five categories. The N_2O category was not included because all 5 stoves has no emissions (MacCarty et al., 2008).

The use of wood as fuel effects deforestation on top of releasing emissions. In 2015, it was stated that about 55% of the wood harvested worldwide is used for cooking and heating by charcoal and firewood (Bailis et al., 2015). Other causes that contribute to deforestation don't involve harvesting wood, such as agriculture uses. Therefore, while a majority of the wood that is harvested is used for cooking and heating, this does not make the use of traditional cookstoves the top cause of deforestation.

2.2.2. Health Impacts of Traditional Cookstoves

According to the World Health Organization (WHO), household air pollution leads to 3.2 million premature deaths worldwide (WHO, 2022). Household air pollution is from the incomplete combustion of biomasses used in traditional cookstoves. These illnesses that cause death in descending order of occurrence include ischemic heart disease, stroke, lower respiratory infection, chronic obstructive pulmonary disease, and lung cancer. 2019 was an extraordinarily awful year because it was estimated that household air pollution took 86 million healthy lives. This burden unproportionally effects women in low and middle income countries (WHO 2022). In 2010, the International Energy Agency (IEA), estimated that by 2030 there would be more premature deaths due to indoor pollution than HIV/AIDS. Outside of deaths caused by indoor air pollution, it also affects low birth rates, adverse pregnancy outcomes, asthma, child cognitive functions and more (Malla, 2014).

2.3. Performance Analysis of Cookstoves

There are multiple different tests that can be run on cookstoves to collect data including the Water Boiling Test (WBT), Controlled Cooking Test (CCT), and Kitchen Performance Test (KPT). The WBT is used to replicate common cooking techniques, the CCT is used to test a specific cooking task, and the KPT is used to test actual performance in a house (Sutar et al., 2014). The WBT is one of the most common tests and allows for comparison between different stoves' performance. For a WBT, a pot of water (2.5 or 5 liters) is heated until boiling and then simmered for 45 minutes (MacCarty et al., 2010).

Another differing factor in cookstove testing is the collection of emission data, where there are two common methods. The hood method uses a hood to extract the data from the emissions of the cookstove. The chamber method places a cookstove in the chamber and the air inside the chamber is tested throughout the test. (Suter et al., 2014). The data collected is to provide information on emission and efficiency of the cookstove.

2.3.1. Importance of Data Analysis

Data analysis is to extract meaning from the data (Coursera, 2023). Cookstove testing provides a lot of data, making it especially important to use data analysis to understand the meaning of the result. Recommendations on how to improve stove designs are given based on results of testing, found through data analysis (MacCarty et al., 2010). Being able to understand the patterns found in tests, allows for faster identification of areas of improvement.

2.4. Laboratory Emissions Measurement System (LEMS)

Aprovecho is a nonprofit organization that helps to establish labs in developing countries to allow for testing cookstoves, which leads to better cookstoves. To help establish labs, Aprovecho designs and distributes low-cost user-friendly emissions testing systems to organizations around the world. They specialize in rapid prototyping of iterative designs of wood burning cookstoves and measuring emissions and thermal measurements (Aprovecho, 2021). They often run WBT with the hood collection system to obtain data on their designs of cookstoves.

2.4.1. Levels of Data

There are three levels of data within LEMS. Level 1 is one stove tested one time. Level 2 is one stove tested multiple times. Level 3 is multiple stoves, each tested multiple times. The reason for the different levels is to provide different analysis based on the data that is being analyzed. The analysis of a data set with multiple different stoves looks different than the analysis of a stove tested once.

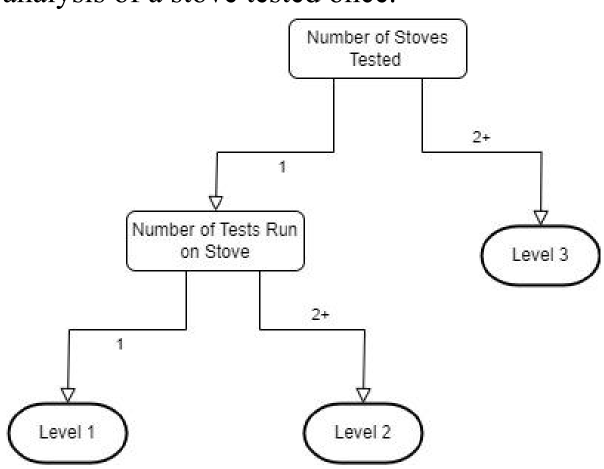


Figure 2. Flowchart of which level of data analysis to use.

2.5. Regional Testing Centers

The purpose of regional testing centers is to be able to test stoves worldwide with international standards. They also work with stove developers in their area to improve cookstoves (Jetter, 2016). This is significant because to make real change, it has to be easy to get stove testing. They work with companies, like Aprovecho, to set up testing protocols and establish a method for analyzing data. Regional testing centers can be found worldwide, and a list of regional testing centers and places that use LEMS can be found below.

Location	Institution
Beijing, China	Beijing University of Chemical Technology
Bolivia	CPC
Burkina Faso	Burkina Faso GIZ
Cambodia	Institute of Technology of Cambodia (ITC)
Cambodia	ITC
China	SSM
China	China Agricultural University
Colorado, USA	University of Colorado Boulder
DRC	IM World Health
Ghana	CSIR
Ghana	TCC KNUST
Ghana	Council For Scientific and Industrial Research
Honduras	Zamorano
India	Itt Bombay
India	BSH India
India	SAI
India	Prakti
Iowa, USA	Iowa State University
Kathmandu	CRT Nepal
Kigali, Rwanda	Rwanda Standards Board
Kinshasa, DRC	BLC America CERERK/ISTA
La Paz, Bolivia	Universidad Mayor de San Andrés (CPC)

Laos	SNV
Mexico	National Autonomous University of Mexico (UNAM)
Mozambique	GIZ
Myanmar	GERES
Nagpur, India	CSIR-National Environmental Engineering Research Institute (CSIR-NEERI)
Nairobi, Kenya	Kenya Industrial Research and Development Institute
Nepal	RETS
New York, USA	Biolite
Nigeria	University of Nigeria
Nigeria	ICEED
North Carolina, USA	North Carolina State University
Ohio, USA	University of Dayton
Oregon, USA	George Fox University
Peru	Servicio Nacional de Capacitación para la Industria de la Construcción
Philippines	Alexis Labs
Port-au-Prince, Haiti	Institut des Sciences, des Technologies
Senegal	Université Cheikh Anta Diop de Dakar
Timor Leste	DIT
Uganda	Nyabyeya Forestry College
Uganda	CREEC
Utah, USA	Brigham Young University (BYU)
Vietnam	SNV Vietnam
Washington, USA	BURN design

Table 1. Labs and Regional testing centers that use LEMS provided by Aprovecho

2.6. *Challenges with current Data Analysis Tool*

Currently, there is a spreadsheet that is used to analyze the data that Aprovecho and other global labs obtain. One problem with the current analysis tool is that it is not automated, which leads to a lot of time spent entering data. Another issue is that there is a lot of room for error and confusion. The formulas within the spreadsheet can easily be changed without the user knowing. This leads to incorrect analysis and confusing data patterns. The current tool is also hard to understand because there is no place for clear instructions within a spreadsheet. There also has to be multiple files for each level of analysis making it confusing for the user.

2.7. User Experience

There are many definitions of user experience and they all touch on how users interact with the interface, how they feel when using the interface, why they use the interface, and how the interface meets the user's needs (Allam et al., 2013). A user interface is what the computer uses to interact with the user and should be designed based on the user's needs and their skill level (Jacob 2003).

In 1994, Jakob Nielsen created 10 usability heuristics for user interface designed, and they have remained relatively unchanged (Nielsen 2020). A factor analysis of 249 usability problems went into defining these 10 principles, which are summarized below.

- *Visibility of System Status*
  Users should be informed on the status of the user interface through feedback in an adequate amount of time (Nielson, 2020).
- *Match Between System and the Real World*
  Users should understand what the interface is saying without prior knowledge of the topic. The information should also appear in logical order (Nielson, 2020).
- *User Control and Freedom*
  Users should be able to exit an unwanted action quickly and easily because users often mistakenly perform actions (Nielson, 2020).
- *Consistency and Standards*
  Users should understand what action they are about to perform. It is suggested that platforms and industry conventions be followed for naming of actions to avoid confusion (Nielson, 2020).
- *Error Prevention*
  Get rid of error-prone conditions or have confirmation messages before doing an action. Having error messages is important, but it would be better to prevent errors from the start (Nielson, 2020).
- *Recognition Rather than Recall*
  Don't require users to memorize elements, actions, and options. Instead, make them visible. When going between interfaces, users shouldn't have

to remember information. All information the user needs should be visible or easy to get to (Nielson, 2020).

- *Flexibility and Efficiency of use*
  Have shortcuts that can be used by experienced users to speed up the process, but be hidden from other users to reduce confusion. This allows for the experienced and inexperienced users to use the interface. All users should be able to customize the interface to their frequent actions (Nielson, 2020).
- *Aesthetic and Minimalist Design*
  User interfaces should only include information that is relevant to not take away from the relevant information (Nielson, 2020).
- *Help Users Recognize, Diagnose, and Recover from Error*
  Error messages should be easily understood, highlight the problem, and offer a solution (Nielson, 2020).
- *Help and Documentation*
  Provide documentation to aid in the user's understand of how to use the interface; although, a system without additional explanation is better (Nielson, 2020).

The user of the interface will define the regulations that the interface must follow, as well as taking into account Nielson's heuristics. For this interface, the stakeholders will be the users that interface is designed for.

2.8. Stakeholders

Stakeholders are entities that have the power to influence the outcome of a given project (Brugha et al., 2000) or are affected by the outcome of a given project. The key stakeholders for the new data analysis interface are regional testing centers, Aprovecho, and traditional cookstove owners. The regional testing centers have the ability to influence the outcome of the project and are affected by the outcome of the project. One objective of the project is to make analysis for regional testing centers easier, therefore, their input will heavily influence the design of the interface. They are also affected by the outcome because they will be using the data analysis interface on a regular basis to perform the data analysis they are currently using the old analysis tool for. Aprovecho has a similar role as a stakeholder with some minor changes. Aprovecho has more control over the outcome of the project because they are providing code and constantly working with the tool to get it ready for regional testing centers and other users. Traditional cookstove owners are only affected by the outcome of the project. If the analysis of the data is easier and more accurate, then more efficient and safer stoves can be designed. This will help to improve the health of the traditional cookstove owners, as talked about in section 2.2.2. All the key stakeholders are mentioned above, however; for similar reasons that traditional cookstove owners are stakeholders, everyone is a stakeholder. As described earlier in section 2.2.1, traditional cookstoves impact climate change and deforestation. Everyone can benefit from more efficient and safer cookstoves for this reason.

3. *Methods*

3.1. *Design Requirements*

For this project, there were many requirements for the interface. The first requirement was to use python as the coding language to create the interface. While working on and prior to the interface, there was code developed in python to perform different cookstove data analyses. Another requirement was to be user-friendly because the interface was intended to be used by people who don't have a technical background and don't speak English as their first language. While being user-friendly, it was also important to be flexible by allowing easy manipulation by the user to match their skill level. The final major requirement was that the interface needed to show graphs, intermediate calculations, and real time data. A house of quality is shown in figure 3 that shows all the design requirements.

		Engineering Specifications (ES)									
Customer Requirements (CR)	Customer Weights	Coded in Python	Export Data to a Savable File	Don't Need a Technical Background	Don't Need English as First Language	Interface can be Manipulated by User	Requires 8 Hours or Less of Training	Can be Distributed Around the World	Graphs Seen Within 2 Minutes	Chart with Intermediate Calculations	Real Time Data Graphs
Uses Python	2	9	1			3					
Easy to Store Data	4		9	1	1						
User Friendly	1	1		9	9	3	3				
Flexible Interface	7	3		3	3	9	3				
Doesn't Require Much Training	5			9	9	3	9				
Easy to Distribute	3							9			
Quickly See a Set of Graphical Output	6	3							9		
Show Intermediate Calculations	6	3								9	
Show Real Time Data Graphs	6	3							1		9

Figure 3. House of quality for cookstove analysis interface. 9 means a very strong relationship, 3 means strong relationship, 1 means weak relationship, blank means no relationship.

3.2. *Selection of Jupyter Notebook*

At the start of the project, a Jupyter Notebook was suggested as the platform for the project over the previously used Excel spreadsheet. To determine if the Jupyter Notebook could fill all requirements, its technical specifications and applications were reviewed. The notebook reviewed was entitled Probabilistic Seismic Hazard Analysis for the Sliding Displacement of Rigid Sliding Masses (Saygili et al., 2018). This notebook computed a hazard curve using Rathje and Saygili probability approach. The inputs for this notebook were required constants, such as yield acceleration of the slope and ground motion hazards. Outputs for this notebook were graphs, such as a graph of the Vector displacement hazard curve (Saygili et al., 2018). This notebook was found within a Jupyter Hub that has other scientific based Jupyter Notebook examples ("Jupyter Hub.", n.d). Other sites were used to give additional information such as how to hide code from the user (Płońska et al., 2022) and how to add interactive elements (Martinelli, 2017). Based on this, several advantages over Excel-based interfaces were identified:

3.2.1. *Code Structure*

In many Jupyter Notebooks, the code is set up with section headers and descriptions throughout. This will be useful for this project's purpose because it allows for information to be provided on what to look for in the graphs or norms/standards that people can compare to. It can also provide instructions for how to upload data correctly and how to run the Jupyter Notebook. The headers with instructions or extra information provided throughout the notebook is something that sets Jupyter Notebook apart from Excel. It will make it easier for the user to use the interface and better interpret the data that is output. A simple example of how this use is provided in figure 4 (Saygili et al., 2018).

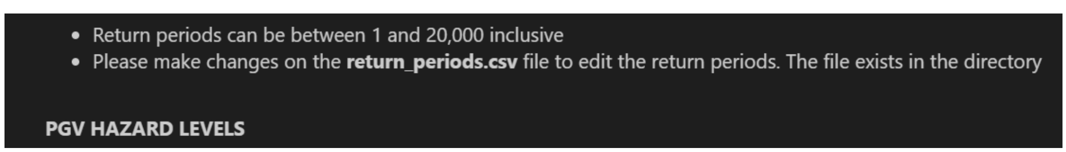


Figure 4. An example of how a header can be used within a Jupyter Notebook (Saygili et al., 2018).

3.2.2. *Code Accessibility*

Most Jupyter Notebooks present the code in the notebook such that it can be accessed by the user. This is helpful for users that want to better understand how the code works and the calculations that are performed. There is a downside to this because it could make the user more confused if they don't understand coding. Luckily, there is a way to hide the code from the user (Płońska et al., 2022). This way, the user can not alter the code. This is a benefit of a Jupyter Notebook over an Excel file because in Excel it is easy to click the interface and alter the equations. With hiding the code, there is added protection to the code and makes the notebook easier to use; however, the user do have a choice to see the code if they wish. This allows people to make small edits as needed and fully understand the calculations performed if wanted.

3.2.3. Graphing Capability

Most of the graphs in the example Jupyter Notebook that were studied are simple and similar to the graphs that are output in the Excel file. They are a few lines with a legend and axes labels. This means that the graphs made in the Excel file can be easily replicated in the Jupyter Notebook interface. An example of a simple graph done in a Jupyter can be seen in figure 5 below (Saygili et al., 2018).

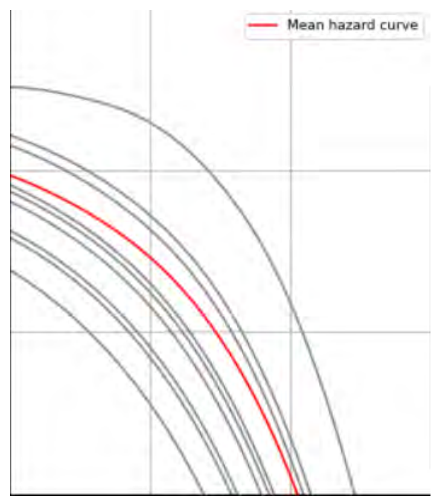
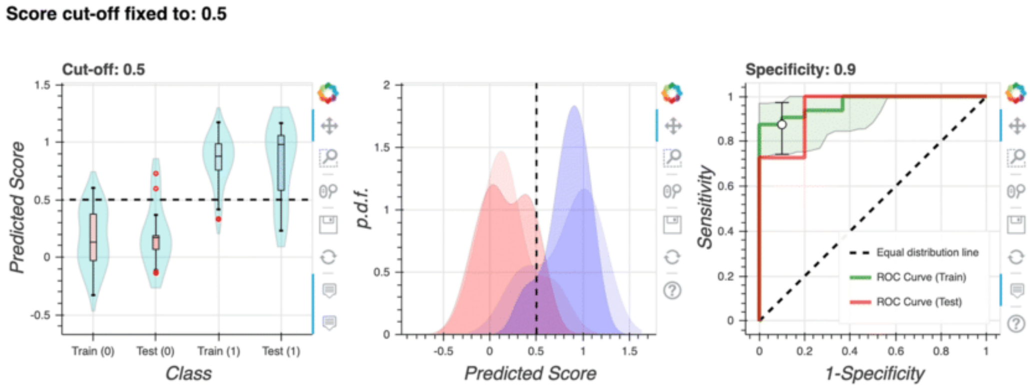


Figure 5. Example of Simple Jupyter Notebook Graph (Saygili et al., 2018)

Although a Jupyter Notebook is capable of the same things as Excel, it also has more capabilities. Figure 6 shows some more complicated possibilities of a Jupyter Notebook. It would be nice to compare the data points from testing with averages, or even show the distribution of multiple test scores using the graph furthest on the left of figure 6. There are also lots of colors used and error bars that will come in handy while making graphs for this project. While in Excel it may be possible to do some of these functions, the graphs in figure 6 look a lot cleaner and more organized.



#	Evaluate	MW-U Pvalue	R2	AUC	Accuracy	Precision	Sensitivity	F1score
0	Train	2.66e-10	0.67 (0.5, 0.77)	0.97 (0.91, 1.0)	0.89 (0.81, 0.94)	0.9 (0.82, 0.95)	0.88 (0.74, 0.97)	0.89 (0.8, 0.95)
1	Test	6.37e-04	0.52	0.95	0.76	0.8	0.73	0.76

Figure 6. Examples of advanced Jupyter Notebook Graphs (Saygili et al., 2018)

3.2.4. Interactivity

Finally, the last feature of a Jupyter Notebook that sets it apart from an Excel interface is the interactive possibilities. It is possible to have interactive graphs and tables in a Jupyter Notebook, which will be very helpful in this project. One example is being able to take a graph with upwards of 8 lines and click which lines you want to see. In figure 7, on the top, the original graph can be seen. On the bottom, the graph once interacted with by the user can be seen (Martinelli, 2017). This can be very useful in this project because some graphs have a lot of lines and information on them, and it would be helpful to the user to be able to zoom in on data as necessary.

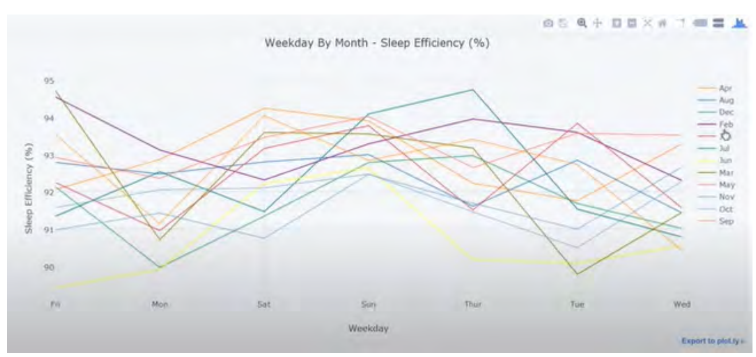


Figure 7. Interactive Graph Example in Jupyter Notebook (Martinelli, 2017)

Also, there can be interactive tables. This is where there is a long data table and the user can search for the item they are looking for or organize by categories. In this project, there are a lot of data tables output with a lot of information. This will be an easy way for the user to have all the information, but efficiently search for the data that they need. An example of this can be seen in figure 8 (Martinelli, 2017). This list is of the USDA National Nutrient Database. After searching eggplant in the interface and grouping by zinc, figure 8 shows all that was left. This makes it very simple for the user to find the data they are looking for. Also, depending on how the data is grouped, it will be a lot easier for the user to compare certain data points. These interactive capabilities really set apart the Jupyter Notebook interface from the Excel interface, as these interactions can make the user experience more meaningful and organized.

```
In [7]: interact(lambda food, nutrient: get_nutritional_info(nutritional_db, food, nutrient),
                 food='',
                 nutrient=list(nutritional_db['Nutrient'].unique()))
```

food: eggplant

nutrient: Zinc, Zn

	Food	Nutritional Value	Units	Nutrient
142404	Eggplant, raw	0.16	mg	Zinc, Zn
142405	Eggplant, cooked, boiled, drained, without salt	0.12	mg	Zinc, Zn
142836	Eggplant, cooked, boiled, drained, with salt	0.12	mg	Zinc, Zn
147446	Eggplant, pickled	0.23	mg	Zinc, Zn

Figure 8. Interactive table example in Jupyter Notebook (Martinelli, 2017)

Since the Jupyter Notebook was found to have all the Excel spreadsheet capabilities, plus extra capabilities, it made most sense to continue the project using the Jupyter Notebook.

3.3. *Co-Design*

The Jupyter Notebook went through the process of co-design. A definition of co-design by Kleinsmann and Valkenburg is "the process in which actors from different disciplines share their knowledge about both the design process and the design content … in order to create shared understanding on both aspects … and to achieve the larger common objective: the new product to be designed" (Steen, 2013). The definition explains that co-design involves everyone involved sharing their knowledge on the process of creation as well as what is being created, and using everyone's knowledge for the final product.

The Aprovecho team provided knowledge on how to approach the creation of the Jupyter Notebook, how testing worked, and all the python scripts that run in the background of the Jupyter Notebook. This was combined with the knowledge the author has of how to create a Jupyter Notebook. Throughout the process, there were biweekly meetings to share new knowledge on the process and product, whether that be testing results, new code, or further explanations of how the current code is running. Through the process of sharing ideas and knowledge, this Jupyter Notebook was able to be created.

4. Results

The resulting LEMS processing notebook is split into 8 different sections. The first section is the setup, which includes defining the number of tests and entering the data entry form for each test. The next 7 sections are the different calculations that can be run. Before each of the calculation sections, the notebook specifies the values this section will calculate, what files the user must have to run this section, other calculation selections that must be run, what the expected output is, and instructions or special notes for running that section. At the start of each calculation section, the user is prompted if they want to run that section with the line "Would you like to calculate (calculation type)? Press 0-No 1-Yes". An empty text box appears under this line for the user to enter 0 or 1 to determine if the code for that section will be run. All the sections will be described in detail below. It is important to note that for testing, if the notebook is run with more than one test, the same test file is used for each trial. This means that the same test data is being used for each trial. This is due to availability of test data, but produces the same result as using two different test files.

4.1. Setup

After opening the Notebook, the first thing the user will see is instructions on how to install Jupyter Notebook and python on Windows. The user will also see instructions on how to run the Jupyter Notebook. The instructions for installing Jupyter Notebook and python are within the notebook to keep all relevant information together. The installation instructions can be emailed by someone at Aprovecho if the Jupyter Notebook can not be opened.

How to install Jupyter Notebook on Windows

1. Go to https://www.python.org/downloads/windows/
2. Select Download Windows installer (32-bit) or Download Windows installer (64-bit) depending on your system
3. Open the file that downloaded
4. Hit install now a) make sure that use admin privileges when installing py.exe checked b) make sure that add python.exe to PATH is checked
5. Should get a window that says Set up was Successful (Use this site for additional help https://www.geeksforgeeks.org/how-to-install-python-on-windows/)
6. Go to Command Prompt
7. Use the following command "python get-pip.py"
8. Should see Successfully installed pip in the command window
9. Use the following command "python -m pip install jupyter" (don't include the quotation marks " ")
10. Jupyter notebook is now installed (use this site for additional help https://www.geeksforgeeks.org/how-to-install-jupyter-notebook-in-windows/).
11. Use the following command to open jupyter notebook "jupyter notebook" (don't include the quotation marks " ")

How to Run the Jupyter Notebook

To run this Jupyter Note book

1. Go to kernel → Restart & Run All
2. Scoll back to top of the page and hit here where it says "Click here to show/hide codes in this notebook" (optional) (can be found near the bottom of this page)
3. Jupyer Notebook will run

Figure 9. Initial instructions found in Jupyter Notebook.

If the user follows the instructions to run the notebook, they will see an output that says "click here to show/hide codes in this notebook". This can be used to switch between hiding and showing the code to allow for an experienced user to see the code and beginner users to hide the code. When the notebook is started, the code is always showing.

[1]: Click here to show/hide codes in this notebook.

Figure 10. Code output that allows the user to toggle between showing the code and hiding it.

Next, an output that reads "Code is running" appears. The following steps take a few seconds to process, so this output is to reassure the user that the code is running correctly, even if no outputs are seen. The user can also see a description of the setup to prepare the user to answer some questions in order to understand the system the code is running on.

Setup

Be prepared to answer questions:

```
1) Type of Computer System
2) Test Type
3) Number of tests run
```

Figure 11. Setup directions found in the Jupyter Notebook.

The following steps of code that take a while is the installation and importing of necessary python packages. These packages include, but are not limited to, openpyxl, uncertainties, tabulate, easygui, pandas, and matplotlib. These python packages allow for a variety of tasks that occur in the notebook, such as plotting and creating tables. It takes approximately 15 seconds for the installation and importing to occur. The user is then prompted to specify their computer system with the line “Please specify computer system. Put L for Linux and W for windows”. Underneath that line is an empty text box with a flashing cursor to signify to the user that they should enter their computer system. After entering either L or W (l and w are also accepted), the system will respond by printing “You have a Linux system” or “You have a Windows System”. If neither an L nor W is inputted, the system will print, “You have not entered a possible computer system. Please Try Again”. The user is then prompted to enter L or W again. This cycle will continue until the user inputs an L or W.

The user is then prompted to enter the specific test type with the line “Please specify test type. Put 19867 for iso 19867 and 19869 for iso 19869”. As with the prompt for the computer system, there is an empty text box for the user to input the test type. The system will continue to ask for a test type until a valid type is entered. Finally, the user is prompted for how many tests were performed with the line “How many tests did you perform? (How many test files do you have to analyze)”. An empty text box is available below for the user to enter a number.

The next step in the setup is entering the path for data entry. This is a very important step because this will create the naming convention for all the data, and tell the computer where the base data for the test is located. There are instructions within the notebook for the user to do this. The user is prompted to enter the path name with the line “Input path of data entry form (spreadsheet):”. There will be an empty text box under that line for the user to input the path. If the user is running with more than one test, this process will occur for the number of tests that the user had set.

Path for Data Entry form

The path to the data entry form is the location of the data entry form on your computer An example of how to get location of a file is

1. Open File Explorer
2. Double Click on file (will open a menu of options)
3. Click copy path
4. Enter path when prompted but DO NOT INCLUDE QUOATION MARKS (")

The file input must be a spreadsheet (ends in .xlsx)

ALL FILES MUST BE CLOSED IN ORDER TO RUN THE NOTEBOOK

Figure 12. Instructions for entering the path for data entry form found in the Jupyter Notebook

After entering the path names, the user will see the files the system loaded and the files the system created for each test.

```
Input path of data entry form (spreadsheet):
 C:\Users\ryana\Documents\Thesis\LEMS-Data-Processing-main\Data\yatzo alcohol\yatzo_test1\yatzo_test1_TE_DataEntryForm.xlsx
LEMS_MakeInputFile_EnergyCalcs v0.3   20231105 12:36:24
loaded: C:\Users\ryana\Documents\Thesis\LEMS-Data-Processing-main\Data\yatzo alcohol\yatzo_test1\yatzo_test1_TE_DataEntryForm.xlsx
created: C:\Users\ryana\Documents\Thesis\LEMS-Data-Processing-main\Data\yatzo alcohol\yatzo_test1\yatzo_test1_EnergyInputs.csv
Input path of data entry form (spreadsheet):
 C:\Users\ryana\Documents\Thesis\LEMS-Data-Processing-main\Data\yatzo alcohol\yatzo_test1\yatzo_test1_TE_DataEntryForm.xlsx
LEMS_MakeInputFile_EnergyCalcs v0.3   20231105 12:36:31
loaded: C:\Users\ryana\Documents\Thesis\LEMS-Data-Processing-main\Data\yatzo alcohol\yatzo_test1\yatzo_test1_TE_DataEntryForm.xlsx
created: C:\Users\ryana\Documents\Thesis\LEMS-Data-Processing-main\Data\yatzo alcohol\yatzo_test1\yatzo_test1_EnergyInputs.csv
```

Figure 13. An example output after entering the path of the data entry form with 2 test files

Finally, there is a note on data files. This informs users how the files have to be organized for the notebook to access necessary files and how the test name is decided.

Note on Data Files

All data files must be in the same file directory (same folder) as the excel file given above. From the excel file, the notebook will determine a test name. For future reference, the testname will be considered test in the descriptions below.

Figure 14. Notes on Data files found in the Jupyter Notebook

4.2. Energy Metrics

This section will calculate the different energy metrics such as cooking power, efficiency, and firepower. First, the user will see the introduction for the energy metrics section.

Energy Metrics

The Energy Metrics section will calculate values such as initial fuel temperature, initial water temperature, initial mass, final fuel temperature, max water temperature, final mass,useful energy delivered, cooking power, efficiency without char, efficiency with char, burn rate and fire power. These are some of the values calulated at high, medium, and low power. If you conduct multiple tests, there results will be compared in the comparison section.

Must Have

1. file in the same folder as the initial excel file with the name "test_EnergyInputs.csv" (in replace of test will be your test name) (will come from Data Entry Section)

Must Run

1. SetUp Section
2. Path for Data Entry Form Section

Will Output

1. a file in the same folder as the initial excel file with the name "test_EnergyOutputs.csv" (in replace of test will be your test name)

Figure 15. Introduction to energy metrics section found in the Jupyter Notebook

There are 3 tables that may be output to the Jupyter notebook. There is a constants table, an energy calculation table, and a comparison table. The constants table is only output if there is one test that is run. The entry calculation table is output if one or multiple tests are run. The table will look very similar if it is one test or two tests. The comparison table will only be output if there are multiple tests run.

variable_name	value
fuel_type	alcohol
fuel_source	
fuel_dimensions	
fuel_mc	
fuel_heating_value	19852.0+/-0
char_heating_value	
boil_temp	99.3+/-0
pot1_dry_mass	2.271+/-0
pot2_dry_mass	
pot3_dry_mass	
pot4_dry_mass	
initial_air_temp	25.0+/-0
initial_RH	60.4+/-0
initial_pressure	987.0+/-0
initial_wind_velocity	
start_time_hp	11:15:30
initial_fuel_mass_hp	3.72+/-0
initial_water_temp_pot1_hp	20.7+/-0
initial_water_temp_pot2_hp	

Figure 16. Example constants table (only seen with one test run)

The energy metrics table is very long because of all the values that are in the table. In figure 17, only a few rows of the table are captured to show the format of the table.

Variable	High Power Value	Medium Power Value	Low Power Value
start_time	11:15:30	1.0+/-0	1.0+/-0
initial_fuel_mass	3.72+/-0	11:56:10	12:40:40
initial_water_temp_pot1	20.7+/-0	3.5785+/-0	3.499+/-0
initial_water_temp_pot2		20.6+/-0	21.4+/-0
initial_water_temp_pot3			
initial_water_temp_pot4			

Figure 17. Example energy metrics table (seen with one or multiple test runs)

Variable	Averages	N	Stadev	Interval	High Tier	Low Tier	COV
thermal_efficiency_w_char	nan	0	nan	nan	nan	nan	nan
thermal_efficiency_wo_char	nan	0	nan	nan	nan	nan	nan
char_mass_productivity	nan	0	nan	nan	nan	nan	nan
char_energy_productivity	nan	0	nan	nan	nan	nan	nan
avg_cooking_power	nan	0	nan	nan	nan	nan	nan
burning_rate	nan	0	nan	nan	nan	nan	nan

Figure 18. Example comparison table (seen only with multiple test runs)

4.3. Adjust Calculations

This section allows for adjustments to be made to the data due to sensor calibrations. First, the user will see the introduction for the adjust calculation section.

Adjust Calculations

This section will adjust the data due to the sensor calibration.

A popup window will appear

1. It will ask if you want to save and close the header file
2. Press ok
3. It will ask to enter the sensorbox firmware version
4. Enter the firmware version (SB4003.16 is the default)
5. Press ok
6. It will tell you "the following plots show the effect of the recalulation. Close the plots to continue"
7. Press ok
8. 2 plots per test will appear in the notebook

Must Have

1. file in the same folder as the initial excel file with the name "test_RawData.csv" (in replace of test will be your test name)

Must Run

1. SetUp Section
2. Path for Data Entry Form Section

Will Output

1. a file in the same folder as the initial excel file with the name "test_RawData_Recalibrated.csv" (in replace of test will be your test name)
2. a file in the same folder as the initial excel file with the name "test_Header.csv" (in replace of test will be your test name)

Figure 19. Introduction to adjust calculations section found in the Jupyter Notebook

There is a series of popup windows that will appear. The first window is to confirm the header path, the second is to enter the sensor box firmware version, and the third is to announce that plots were created

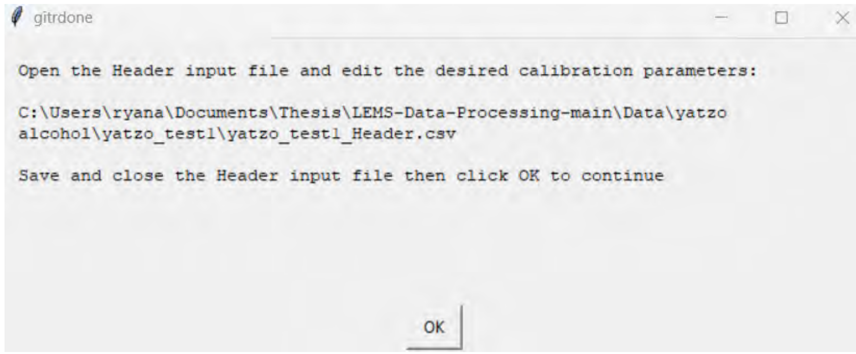


Figure 20. First popup window that appears after running the adjust calculation section

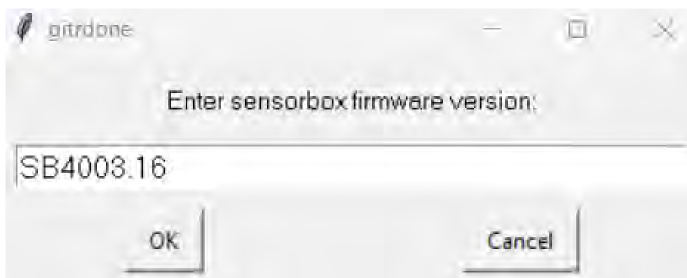


Figure 21. Second popup window that appears after running the adjust calculation section

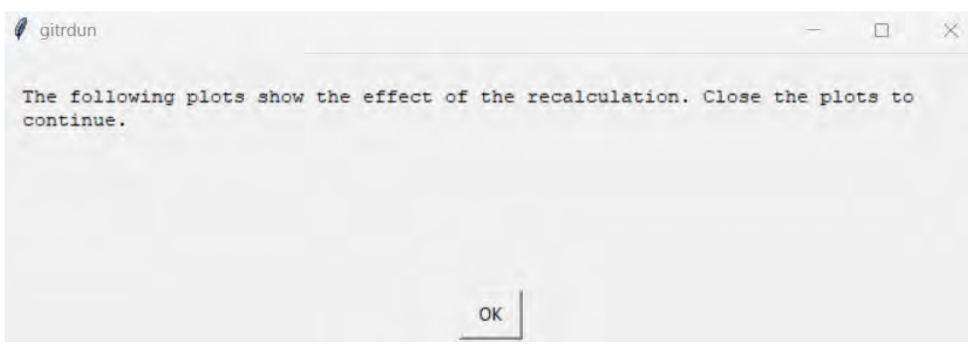


Figure 22. Third popup window that appears after running the adjust calculation section

After the popup windows are closed, there is a blue line indicating the test that was run. Under this line, a log of what the program did and the files that were created is printed. There are also two graphs that appear. One graph shows the Gravflow 1 old and new and the other graph shows Gravflow 2 old and new.

```
---------------------- This is for yatzo_test1 ----------------------

LEMS_Adjust_Calibrations v0.1   20231105 13:12:10
Header file already exists:

C:\Users\ryana\Documents\Thesis\LEMS-Data-Processing-main\Data\yatzo alcohol\yatzo_test1\yatzo_test1_Header.csv

firmware_version=SB4003.16
GravFlo1  updated
GravFlo2  updated
GravFlo1 A_old = 1.0 , A_new = 0.000502758
GravFlo1 B_old = 0.0 , B_new = -11901.2
GravFlo2 A_old = 1.0 , A_new = 0.0004818
GravFlo2 B_old = 0.0 , B_new = -12141.1
recalculated GravFlo1 data series
recalculated GravFlo2 data series
created: C:\Users\ryana\Documents\Thesis\LEMS-Data-Processing-main\Data\yatzo alcohol\yatzo_test1\yatzo_test1_RawData
_Recalibrated.csv
```

Figure 23. Example log printed after the adjust calculation section is run

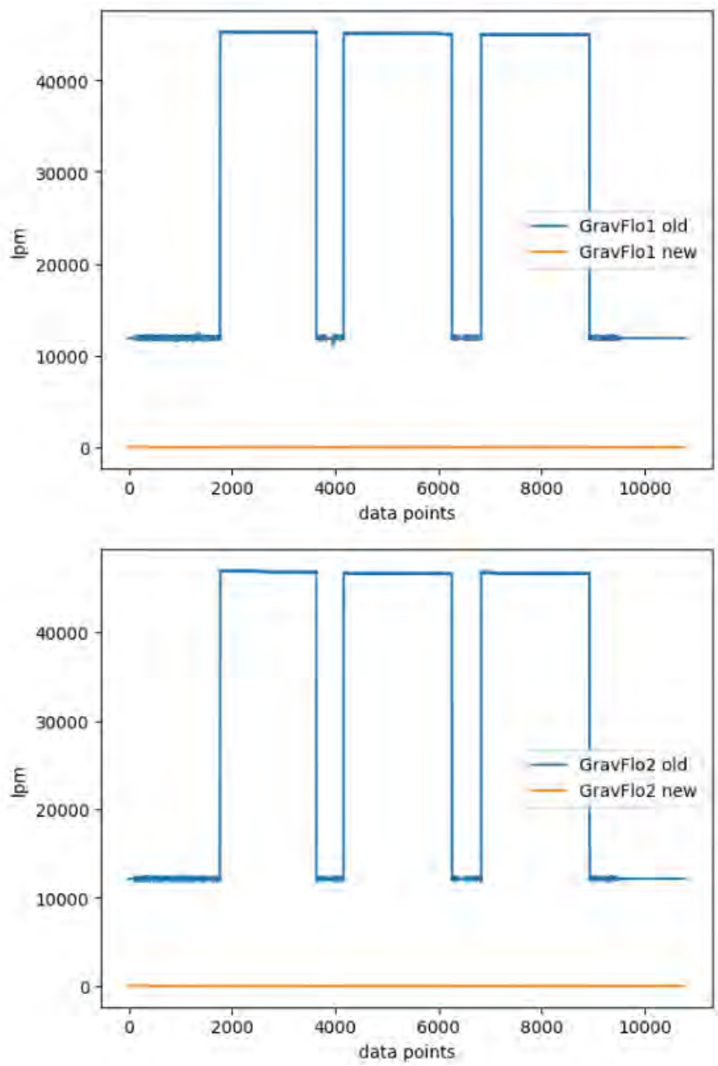


Figure 24. Example graphs output after running the adjust calculation section is run. GravFlo1 and GravFlo2 were very similar for this example and there are very little differences between the two graph.

If this section is run with multiple tests, the above process will continue for each test. Each test will have three pop up windows, a blue line, a log, and two graphs. The blue line contains the test name and will separate the log and graphs for each test.

4.4. *Time Shift Calculations*

The purpose of this section is to complete a time shift to the data. First, the user will see the introduction to the time shift calculations.

Time Shift Calculations

This section will complete a time shift to the given data.

A pop up window will appear

1. It will ask you to enter the seconds to shift each data series
2. Enter the seconds to shift each data series for all the given variables
3. Press ok (if you can not see ok, grab the top of the popup window and drag until you can see the whole window)
4. Nothing else will happen and Nothing will appear on the notebook (that is what is supposed to happen)

Must Have

1. file in the same folder as the initial excel file with the name "test_RawData_Recalibrated.csv" (in replace of test will be your test name). This will come from running the Adjust Calculation Section

Must Run

1. SetUp Section
2. Path for Data Entry Form Section
3. Adjust Calculations Section

Will Output

1. a file in the same folder as the initial excel file with the name "test_TimeShifts.csv" (in replace of test will be your test name)
2. a file in the same folder as the initial excel file with the name "test_RawData_Shifted.csv" (in replace of test will be your test name)

Figure 25. Introduction to time shift calculations section found in the Jupyter Notebook

After accepting the calculations for this section, a popup will appear. This popup allows the user to decide the time shift for a variety of variables.

Figure 26. Popup for the Time shift Calculation section

Next, a blue line with the test name will be printed. Under the blue line is a log of the files that were used and created.

```
--------------------- This is for yatzo_test1 ----------------------

LEMS_ShiftTimeSeries v0.1   20231106 09:57:10
Loaded time series data file:C:\Users\ryana\Documents\Thesis\LEMS-Data-Processing-main\Data\yatzo alcohol\yatzo_test1
\yatzo_test1_RawData_Recalibrated.csv

TimeShifts input file already exists:
C:\Users\ryana\Documents\Thesis\LEMS-Data-Processing-main\Data\yatzo alcohol\yatzo_test1\yatzo_test1_TimeShifts.csv

TimeShifts input file edited:
C:\Users\ryana\Documents\Thesis\LEMS-Data-Processing-main\Data\yatzo alcohol\yatzo_test1\yatzo_test1_TimeShifts.csv
created new shifted time series data file:
C:\Users\ryana\Documents\Thesis\LEMS-Data-Processing-main\Data\yatzo alcohol\yatzo_test1\yatzo_test1_RawData_Shifted.
csv
```

Figure 27. Example log printed after running the time shift calculation section

If this section is run with multiple tests, the above process will occur for that many tests. For each test, there will be one popup, a blue line printed, and a log printed of the files used and created.

4.5. Subtracting Background

The purpose of this section is to account for the background, starting with the introduction to the subtracting background section.

Subtracting background

This section will take into account the background.

A popup window will appear

1. It will ask you to edit phase times in the format hour(hh):minutes(mm):second(ss) where two numbers are given for each category
2. If the phase times you want are different than what is there (Can skip to step 3)

   2a) edit the phase times

   2b) press ok

   2c) continue this process until all the phase times you want to consider and entered and follow on to step 3

3. If the phase times you want to subtract are in the box already (the default values)

   3a) Press Cancel

   3b) two graphs and two tables will appear in the notebook

Must Have

1. file in the same folder as the initial excel file with the name "test_RawData_Shifted.csv" (in replace of test will be your test name) (comes from Time Shift Calculation Section)
2. file in the same folder as the initial excel file with the name "test_EnergyInputs.csv" (in replace of test will be your test name)

Must Run

1. SetUp Section
2. Path for Data Entry Form Section
3. Adjust Calculations Section
4. Time Shift Calculations Section

Will Output

1. a file in the same folder as the initial excel file with the name "test_TimeSeries.csv" (in replace of test will be your test name)
2. a file in the same folder as the initial excel file with the name "test_Averages.csv" (in replace of test will be your test name)
3. a file in the same folder as the initial excel file with the name "test_PhaseTimes.csv" (in replace of test will be your test name)
4. Graphs and Tables that come after the popup window

Figure 28. Introduction to subtracting background section found in the Jupyter Notebook

The user will then see a popup. This popup allows the user to edit the phase times for many variables. For this popup, if the values the user wants is in the box, they need to hit cancel to see the graph. Otherwise, they need to enter there values and hit ok.

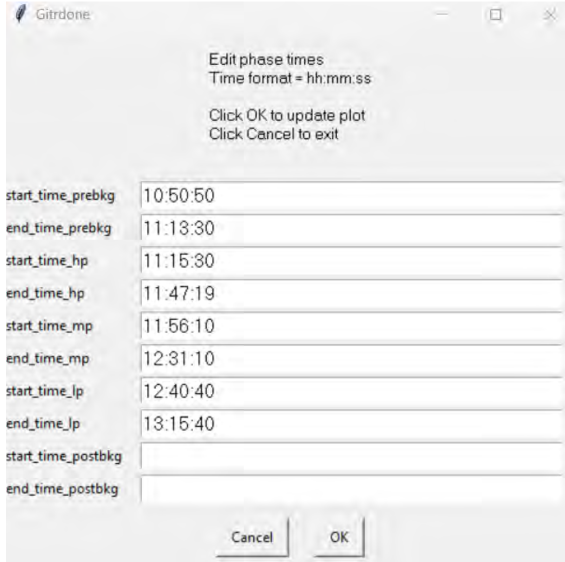


Figure 29. Popup window for the subtracting background section

Each set of values that the user tries gets a set of three graphs. The output is structured as follows: printed blue line with test name, small printed log, set of three graphs, background subtraction report, and a larger printed log.

```
--------------------- This is for yatzo_test1 ----------------------

LEMS_SubtractBkg v0.3   20231105 12:37:42

PhaseTimes input file already exists:
C:\Users\ryana\Documents\Thesis\LEMS-Data-Processing-main\Data\yatzo alcohol\yatzo_test1\yatzo_test1_PhaseTimes.csv
```

Figure 30. Example printed blue line and small printed log

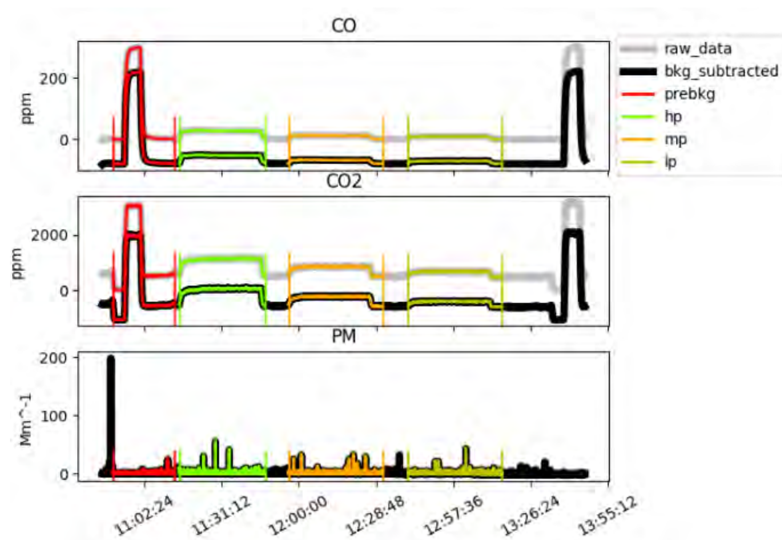


Figure 31. Example set of three graphs output by the subtracting background section

```
background subtraction report:

phase averages before background subtraction:
channel   units     prebkg    hp        mp        lp        bkgValue
-------   -----     ------    ------    ------    ------    ------
CO        ppm       78.9      26.3      10.7      8.1       78.9
CO2       ppm       1082.3    1075.7    781.2     630.5     1082.3
PM        Mm^-1     1.9       2.0       1.8       1.5       1.9
VOC       ppm       -30.0     -29.6     -30.1     -30.2     -30.0
CH4       ppm       473.8     610.6     575.9     551.8     473.8

phase averages after background subtraction:
channel   units     prebkg    hp        mp        lp        bkgValue
-------   -----     ------    ------    ------    ------    ------
CO        ppm       0.0       -52.6     -68.1     -70.7     0.0
CO2       ppm       0.0       -6.6      -301.1    -451.8    0.0
PM        Mm^-1     0.0       0.1       -0.1      -0.4      0.0
VOC       ppm       0.0       0.4       -0.1      -0.2      0.0
CH4       ppm       0.0       136.9     102.1     78.0      0.0
created background-corrected time series data file:
C:\Users\ryana\Documents\Thesis\LEMS-Data-Processing-main\Data\yatzo alcohol\yatzo_test1\yatzo_test1_TimeSeries.csv
created background-corrected time series data file:
C:\Users\ryana\Documents\Thesis\LEMS-Data-Processing-main\Data\yatzo alcohol\yatzo_test1\yatzo_test1_TimeSeries_prebk
g.csv
created background-corrected time series data file:
C:\Users\ryana\Documents\Thesis\LEMS-Data-Processing-main\Data\yatzo alcohol\yatzo_test1\yatzo_test1_TimeSeries_hp.cs
v
created background-corrected time series data file:
C:\Users\ryana\Documents\Thesis\LEMS-Data-Processing-main\Data\yatzo alcohol\yatzo_test1\yatzo_test1_TimeSeries_mp.cs
v
created background-corrected time series data file:
C:\Users\ryana\Documents\Thesis\LEMS-Data-Processing-main\Data\yatzo alcohol\yatzo_test1\yatzo_test1_TimeSeries_lp.cs
v
created phase averages data file:
C:\Users\ryana\Documents\Thesis\LEMS-Data-Processing-main\Data\yatzo alcohol\yatzo_test1\yatzo_test1_Averages.csv
```

Figure 32. Example background subtraction report and large log printed in the subtracting background section

If this section is run with multiple tests, the above process will occur for that many tests. For each test, there will be one popup, a printed blue line, a small printed log, a set of three graphs, a background subtraction report, and a larger printed log.

4.6. Gravimetric Calculations

The purpose of this section is to calculate gravimetric data. First, the user will see the introduction to the gravimetric calculations section.

Gravimetric Calculations

This section will calculate Gravimetric Data

Must Have

1. file in the same folder as the initial excel file with the name "test_GravInputs.csv" (in replace of test will be your test name)
2. file in the same folder as the initial excel file with the name "test_Averages.csv" (in replace of test will be your test name) (comes from Subtracting Background Section)

Must Run

1. SetUp Section
2. Path for Data Entry Form Section
3. Adjust Calculations Section
4. Time Shifts Calculations Section
5. Subtracting Background Section

Will Output

1. a file in the same folder as the initial excel file with the name "test_PhaseTimes.csv" (in replace of test will be your test name)
2. a file in the same folder as the initial excel file with the name "test_GravOutputs.csv" (in replace of test will be your test name)
3. a table with the gavimetric data

Figure 33. Introduction to gravimetric calculations section found in the Jupyter Notebook

After running this section, a blue line is printed with the test name. Under the line, a log and gravimetric PM mass concentration report is printed.

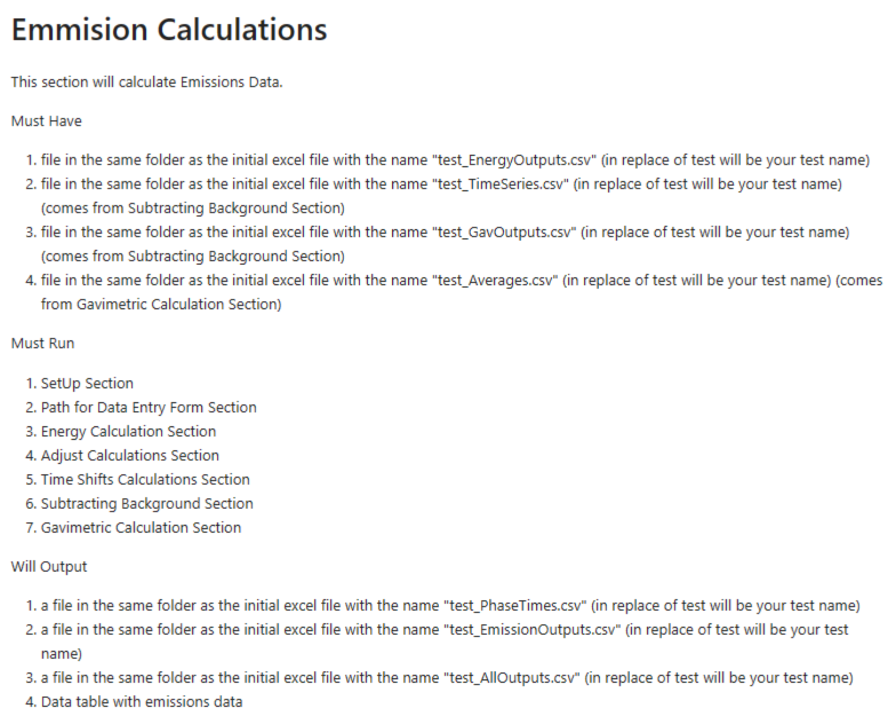
```
--------------------- This is for yatzo_test1 ----------------------

LEMS_GravCalcs v0.2   20231106 10:27:48
Loaded input file of gravimetric filter weights:C:\Users\ryana\Documents\Thesis\LEMS-Data-Processing-main\Data\yatzo
alcohol\yatzo_test1\yatzo_test1_GravInputs.csv
Loaded phase averages:C:\Users\ryana\Documents\Thesis\LEMS-Data-Processing-main\Data\yatzo alcohol\yatzo_test1\yatzo_
test1_Averages.csv
Loaded input file of phase start and end times:C:\Users\ryana\Documents\Thesis\LEMS-Data-Processing-main\Data\yatzo a
lcohol\yatzo_test1\yatzo_test1_PhaseTimes.csv

Gravimetric PM mass concentration report:

Phase:hp
Grav train  channel     net mass (g)        flow (lpm)          phase time (min)  PM conc (ug/m^3)
..........  .......     ............        ..........          ................  ................
A:          GravFlo1    3.7e-05+/-0.0       16.35+/-0.0         31.82             71.1+/-0.0
B:          GravFlo2    3.2e-05+/-0.0       16.326+/-0.0        31.82             61.6+/-0.0
total:      both        6.9e-05+/-0.0       32.675+/-0.0        31.82             66.4+/-0.0

Phase:mp
Grav train  channel     net mass (g)        flow (lpm)          phase time (min)  PM conc (ug/m^3)
..........  .......     ............        ..........          ................  ................
A:          GravFlo1    3e-05+/-0.0         16.633+/-0.0        35.0              51.5+/-0.0
B:          GravFlo2    3.5e-05+/-0.0       16.584+/-0.0        35.0              60.3+/-0.0
total:      both        6.5e-05+/-0.0       33.217+/-0.0        35.0              55.9+/-0.0

Phase:lp
Grav train  channel     net mass (g)        flow (lpm)          phase time (min)  PM conc (ug/m^3)
..........  .......     ............        ..........          ................  ................
A:          GravFlo1    2e-05+/-0.0         16.59+/-0.0         35.0              34.4+/-0.0
B:          GravFlo2    2.8e-05+/-0.0       16.61+/-0.0         35.0              48.2+/-0.0
total:      both        4.8e-05+/-0.0       33.201+/-0.0        35.0              41.3+/-0.0

created gravimetric PM output file:
C:\Users\ryana\Documents\Thesis\LEMS-Data-Processing-main\Data\yatzo alcohol\yatzo_test1\yatzo_test1_GravOutputs.csv
```

Figure 34. Example output printed in the gravimetric calculation section with the log and gravimetric concentration report

If this section is run with multiple tests, the above process will occur for that many tests. For each test, there will be a printed blue line, a printed log, and a gravimetric PM mass concentration report.

4.7. *Emissions Calculations*

The purpose of this section is to calculate emissions data. First, the user will see the introduction to the emissions calculations section.



Emmision Calculations

This section will calculate Emissions Data.

Must Have

1. file in the same folder as the initial excel file with the name "test_EnergyOutputs.csv" (in replace of test will be your test name)
2. file in the same folder as the initial excel file with the name "test_TimeSeries.csv" (in replace of test will be your test name) (comes from Subtracting Background Section)
3. file in the same folder as the initial excel file with the name "test_GavOutputs.csv" (in replace of test will be your test name) (comes from Subtracting Background Section)
4. file in the same folder as the initial excel file with the name "test_Averages.csv" (in replace of test will be your test name) (comes from Gavimetric Calculation Section)

Must Run

1. SetUp Section
2. Path for Data Entry Form Section
3. Energy Calculation Section
4. Adjust Calculations Section
5. Time Shifts Calculations Section
6. Subtracting Background Section
7. Gavimetric Calculation Section

Will Output

1. a file in the same folder as the initial excel file with the name "test_PhaseTimes.csv" (in replace of test will be your test name)
2. a file in the same folder as the initial excel file with the name "test_EmissionOutputs.csv" (in replace of test will be your test name)
3. a file in the same folder as the initial excel file with the name "test_AllOutputs.csv" (in replace of test will be your test name)
4. Data table with emissions data

Figure 35. Introduction to emissions calculations section found in the Jupyter Notebook

After running this section, a blue line is printed with the test name. Under the line, a log is printed and an emission metrics table. The emission metrics table is very long and shows a wide variety of variables. For this example, only a small section of the table is shown to show the organization and data found in this table.

```
---------------------- This is for yatzo_test1 ----------------------

LEMS_EmissionCalcs v0.0   20231106 10:33:58
Loaded phase averages:C:\Users\ryana\Documents\Thesis\LEMS-Data-Processing-main\Data\yatzo alcohol\yatzo_test1\yatzo_
test1_Averages.csv
Loaded energy metrics:C:\Users\ryana\Documents\Thesis\LEMS-Data-Processing-main\Data\yatzo alcohol\yatzo_test1\yatzo_
test1_EnergyOutputs.csv
Loaded gravimetric PM metrics:C:\Users\ryana\Documents\Thesis\LEMS-Data-Processing-main\Data\yatzo alcohol\yatzo_test
1\yatzo_test1_GravOutputs.csv
Loaded phase time series data:C:\Users\ryana\Documents\Thesis\LEMS-Data-Processing-main\Data\yatzo alcohol\yatzo_test
1\yatzo_test1_TimeSeries_hp.csv
created phase time series data file with processed emissions:
C:\Users\ryana\Documents\Thesis\LEMS-Data-Processing-main\Data\yatzo alcohol\yatzo_test1\yatzo_test1_TimeSeriesMetric
s_hp.csv

created emission metrics output file:
C:\Users\ryana\Documents\Thesis\LEMS-Data-Processing-main\Data\yatzo alcohol\yatzo_test1\yatzo_test1_EmissionOutputs.
csv

created all metrics output file:
C:\Users\ryana\Documents\Thesis\LEMS-Data-Processing-main\Data\yatzo alcohol\yatzo_test1\yatzo_test1_AllOutputs.csv
+----------------------------------+--------------------+-------------------------+-------------+
| variable_name                    | units              | value                   | uncertainty |
+----------------------------------+--------------------+-------------------------+-------------+
| fuel_type                        | --                 | alcohol                 |             |
+----------------------------------+--------------------+-------------------------+-------------+
| fuel_source                      | --                 |                         |             |
+----------------------------------+--------------------+-------------------------+-------------+
| fuel_dimensions                  | cmxcmxcm           |                         |             |
```

Figure 36. Example printed log and emission metric table

If this section is run with multiple tests, the above process will occur for that many tests. For each test, there will be a printed blue line, a printed log, and an emission metrics table.

4.8. *Real Time Data Plotting*

The purpose of this section is to graph real time data. First, the user will see the introduction to the real time data plotting section. This introduction has a section explaining how to rerun just the real time data plotting section. This is because the user can manually adjust the real time data file. If the data is adjusted, to get an updated graph the section has to be rerun. To save time, the user can just run the real time data plotting section, instead of running the whole notebook again.

Real Time Data Plotting

This section will plot the real time data.

Must Have

1. file in the same folder as the initial excel file with the name "test_TimeSeries.csv" (in replace of test will be your test name) (comes from Subtracting Background Section)

Must Run

1. SetUp Section
2. Path for Data Entry Form Section
3. Energy Calculation Section
4. Adjust Calculations Section
5. Time Shifts Calculations Section
6. Subtracting Background Section

Will Output

1. a file in the same folder as the initial excel file with the name "test_plots.csv" (in replace of test will be your test name)
2. a graph

You can update the test_plots.csv file to change your plot. If you make a change to the tests_plots.csv file you will need to rerun the plotting code

To rerun the plotting code

1. Scoll to the top of the notebook and click "here" to show the code
2. Scoll back down to the bottom of the code and select the box below this (it will say import PEMS_Plotter1 as plotter at the top
3. Hit run at the top of the screen (there is a sideways triangle before the word run)
4. Scoll back to the top of the notebook and click "here" to hide the code

Figure 37. Introduction to the real time data plotting section found in the Jupyter Notebook

After running this section, a graph of the real time data will be output.

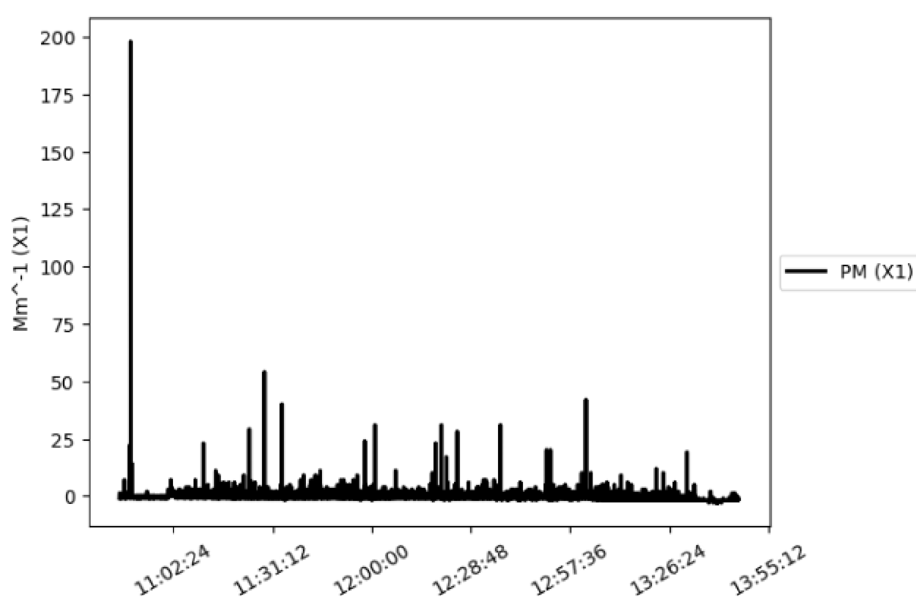


Figure 38. Example real time graph

If this section is run with multiple tests, there will be a graph output for each test.

5. *Conclusions and Next Steps*

A notebook was created to replace the Level 1 and Level 2 Excel sheets currently in place for LEMS data analysis. This notebook proved to Aprovecho that a Jupyter Notebook has potential to be an effective interface for data analysis. Aprovecho has built upon this notebook, and it will be used in Mozambique for its first LEMS real world trial in the near future. Aprovecho is also using this notebook to structure other related projects.

5.1. *Error Handling and Documentation*

The Jupyter Notebook has some error handling within it, but additional work is needed for a more seamless experience. Error handling can be seen in the setup section with the specification of the computer system and test performed. If the user enters an invalid computer system or test, they are asked to re-enter their answer until it is valid. A spot where error handling can be introduced is in the file path entry. Currently, if the user enters an incorrect file path in the path for data entry form, the notebook will break and the user will have to restart the whole notebook. Instead, if an incorrect file path is input into the notebook, the user should be able to enter a new file path without having to restart the notebook. Another spot for error handling would be within each section. Currently, if there is an error in a calculation section, the notebook has to be completely restarted. Instead, if there is a way to just rerun a section at a time, that would provide a smoother user experience.

On top of the error handling, documentation on handling the different kinds of errors should also be developed that includes a spreadsheet with common errors and how to fix the error. This way, if the user runs into a known error, they can easily fix the error on their own without having to contact Aprovecho.

5.2. *Level 3 Notebook*

There are three original Excel sheets: a Level 1, a Level 2, and a Level 3. The notebook is currently meant to work for levels 1 and 2. A Level 3 notebook still has to be created. The difference between a level three notebook and the current notebook, would be the ability to compare multiple different stoves to each other. Comparing different stoves requires different code than comparing different tests.

5.3. *Sharing Files*

An added bonus to this project would be the ability to share files. When running the current notebook, it creates data files and saves them to the local hard drive. Most of the time, the data needs to be shared among different people who use different computers. If the notebook provided an automated way to share these files, that would save the user time and not require that files be emailed every time the notebook is run.

The functions rsync and xcopy were looked at to provide this service. Unfortunately, with these functions, there is no way to verify the data. This means that if someone creates corrupted data, they would share and likely delete previous uncorrupted data. There is more research needed regarding how to share files with a way to verify the data.

5.4. *Interactive Components*

Interactive components were one of the main reasons that a Jupyter notebook was chosen. When talking about interactive components, that is referring to a table where the user can select categories of data to look at, selecting lines on a graph to look at, selecting the x and y-axis on graphs, or any other function where the user can manipulate the table or graphs without having to restart the notebook.

As previously stated, Jupyter Notebook has the ability for these functions, but more research is needed to further understand how to implement these functions. A test was run with a way for the user to manipulate the axis of a graph, but the way that was found is not the most efficient. Currently, if the axis of the graph is wanted to change, the user can manually enter the x and y-axis limits and run the graph again. However, there is a way within Jupyter to slide the graph around or set the axis limits and have the graph change without having to restart the code.

5.5. *Automated Data Entry*

Currently, the notebook still requires spreadsheets with the data. These spreadsheets have to be manually updated to run the code. These spreadsheets also have to be in a specific format and follow certain naming conventions to work with the notebook. If there was a way to automate the creation of these data sheets and the data entry, that would save a lot of time and errors. This would likely have to be done with the sensors. The sensors would have to send the data back and create spreadsheets in the required format. More research has to be done on sensors that have this ability and coding the required formats.

6. Appendix

7. Acknowledgements

The author would like to thank Aprovecho for their assistance in developing the Jupyter notebook and their development of python scripts running in the background of the notebook.

8. *References*

Alem, Yanas. “Mitigating Climate Change through Sustainable Technology Adoption:

Insights from Cookstove Intervention.” *African Economic Research Consortium*, Sept. 2021.

Allam, A H, et al. “User Experience: Challenges and Opportunities.” *JISRI*, 2013.

Aprovecho. “Aprovecho Research Center- Testing and Improving Biomass Cookstoves.” *Aprovecho*, 2021, aprovecho.org/.

Bailis, Robert, et al. “The Carbon Footprint of Traditional Woodfuels.” *Nature Climate Change*, vol. 5, no. 3, 19 Jan. 2015, pp. 266–272, https://doi.org/10.1038/nclimate2491.

Brugha, R., and Zsuzsa Varvasovszky. “Stakeholder Analysis: A Review.” *Health Policy and Planning*, vol. 15, no. 3, 2000, pp. 239–246, https://doi.org/10.1093/heapol/15.3.239.

CCA. “Our Mission & Impact.” *Clean Cooking Alliance*, 3 Sept. 2021, cleancooking.org/mission-impact/.

CCA. “Sector Directory.” *Clean Cooking Alliance* , n.d, cleancooking.org/sector-directory/.

Coursera. “Data Analytics: Definition, Uses, Examples, and More.” *Coursera*, 2023, www.coursera.org/articles/data-analytics.

Goldemberg, Jose, et al. “Household Air Pollution, Health, and Climate Change: Cleaning the Air.” *Environmental Research Letters*, vol. 13, no. 3, 23 Feb. 2018, p. 030201, https://doi.org/10.1088/1748-9326/aaa49d.

Jacob, Robert J. K. “User Interface.” *Encyclopedia of Computer Science*, 1 Jan. 2003, pp. 1821–1826, https://doi.org/1074100.1074893.

Jetter, James, et al. 2016, “Cookstove Laboratory Research Fiscal Year 2016 Report”

“Jupyter Hub.” *Jupyter Hub - DesignSafe User Guide*, www.designsafe-ci.org/user-guide/tools/jupyterhub/#example-notebooks. Accessed 16 Nov. 2023.

Kshirsagar, Milind P., and Vilas R. Kalamkar. “A comprehensive review on biomass cookstoves and a systematic approach for modern Cookstove Design.” Renewable and

Sustainable Energy Reviews, vol. 30, 2014, pp. 580–603, https://doi.org/10.1016/j.rser.2013.10.039.

MacCarty, Nordica, Damon Ogle, et al. "A Laboratory Comparison of the Global Warming Impact of Five Major Types of Biomass Cooking Stoves." *Energy for Sustainable Development*, vol. 12, no. 2, June 2008, pp. 56–65, https://doi.org/10.1016/s0973-0826(08)60429-9.

MacCarty, Nordica, Dean Still, et al. "Fuel Use and Emissions Performance of Fifty Cooking Stoves in the Laboratory and Related Benchmarks of Performance." *Energy for Sustainable Development*, vol. 14, no. 3, 29 June 2010, pp. 161–171, https://doi.org/10.1016/j.esd.2010.06.002.

Malla, Sunil, and Govinda R Timilsina. "Household Cooking Fuel Choice and Adoption of Improved Cookstoves in Developing Countries: A Review." *Policy Research Working Papers*, 1 June 2014, https://doi.org/10.1596/1813-9450-6903.

Martinelli, Alex. "Interactive Visualizations in Jupyter Notebook." *Medium*, Towards Data Science, 27 Sept. 2017, towardsdatascience.com/interactive-visualizations-in-jupyter-notebook-3be02ab2b8cd.

Nielsen, Jakob. "10 Usability Heuristics for User Interface Design." *Nielsen Norman Group*, 15 Nov. 2020, www.nngroup.com/articles/ten-usability-heuristics/.

Płoński, Piotr, and Aleksandra Płońska. "How to Hide Code in Jupyter Notebook?" *MLJAR*, Aleksandra Płońska, Piotr Płoński, 30 May 2022, mljar.com/blog/jupyter-notebook-hide-code/.

Saygili, Gokhan, et al. *Application of Probabilistic Framework For Rigid Sliding Displacements Vector (PGA, PGV) Approximation*, 7 Feb. 2018.

Steen, Marc. "Co-Design as a process of joint inquiry and Imagination." *Design Issues*, vol. 29, no. 2, 2013, pp. 16–28, https://doi.org/10.1162/desi_a_00207.

Sutar, Kailasnath B., et al. "Biomass Cookstoves: A Review of Technical Aspects." *Renewable and Sustainable Energy Reviews*, vol. 41, 3 Sept. 2014, pp. 1128–1166, https://doi.org/10.1016/j.rser.2014.09.003.

Urmee, Tania, and Samuel Gyamfi. “A review of improved Cookstove Technologies and programs.” *Renewable and Sustainable Energy Reviews*, vol. 33, 2014, pp. 625–635, https://doi.org/10.1016/j.rser.2014.02.019.

WHO. “Household Air Pollution.” *World Health Organization*, 28 Nov. 2022, www.who.int/news-room/fact-sheets/detail/household-air-pollution-and-health.