Standard Operating Procedure
Infrared Gas Analyzer (IRGA) CO_2 measurement by integration method

Brief description:

This method uses continuous measurements (~ every second) from an Infrared Gas Analyzer (IRGA) unit to measure the CO_2 concentration of a gas sample. The protocol described here only details the use of a custom software script to convert the raw continuous data from the IRGA unit software into discrete sample-by-sample integrations. When used in conjunction with the appropriate standards these integrals can give you the concentration of CO_2 in the sample that you injected. The output is an excel spreadsheet with each sample's integrated CO_2 value and the time at which it was taken. This output can then be copied into your sample spreadsheet.

Materials:

A spreadsheet must be created during the gas sampling that notes the time that each sample was pushed through the IRGA. Excel has a timestamp function that you can automate with a 'Macro' to output the date and time down to the second at the press of an assigned keystroke. A computer that has the capability of running a Python script is required. This protocol assumes that you are using the Robertson lab computer in the gas lab.

Procedure:

Create a spreadsheet where you mark each of your samples with a timestamp that corresponds to the time when each gas sample was taken and injected into the IRGA.

Open the IRGA software on the lab computer and create a new file in a designated folder (this is important for the next steps) by clicking on the red 'record' circle. **Make sure that your filename has no spaces (ex. '5_20_20_data.txt' not '5/20/20 data.txt').** Make note of the 'background' CO_2 level registered on the IRGA when just N_2 gas is flowing through the IRGA. This is the threshold for splitting samples that you will input into the custom software script call. Keep track as this concentration may fluctuate during sampling. You can 'Zero' the IRGA after the background CO_2 reading stabilizes. If the baseline during sampling is 0.0 ppm CO_2, and your injections are correctly spaced (no overlapping output), then a threshold of 1.0 ppm CO_2 will work fine. Overlapping samples run the risk of not allowing the reading to drop below the specified threshold and thus the samples will be lumped together by the script.

It is important that you run the N_2 gas into the IRGA at the same flow rate during a sampling run (including when your standards are run) and between sampling runs. You can measure this with a flow meter. If your flow rates are different between sampling runs, they can be corrected if the rates are known.

Begin injecting samples, marking their timestamp on your spreadsheet. Only inject the next sample after the CO_2 reading is consistently below the chosen threshold. Once all samples have been injected, click the 'stop' icon on the IRGA software and shut down the software.

The file and protocol are located online with a website called Github for you to download a copy of. You can go to this address to locate the file and protocol:

https://github.com/grantfalvo/Robertson_Lab_SOP_scripts

Locate the green button labeled 'Code', select it and choose 'Download Zip'. This will download all of the files on this site. Find the file called 'IRGA_Raw_data_processing_LI_820.py' or 'IRGA_Raw_data_processing_LI_830.py' and drag it to the folder that you just created. The two scripts handle the output from the LI-820 and the LI-830, respectively. You will need to choose the correct one based on the model of the IRGA you are using.

Open the Python running program of your choice. For the Robertson lab computer, locate and open Spyder (Anaconda3). Within Spyder (Anaconda3) click 'File – Open' and open the file in your newly created folder titled 'IRGA_Raw_data_processing_LI-8X0.py' (Figure 3). The code should populate one of the panels in the program (Figure 4). You do not have to write or even read any of this code. Do not change any of the code in the original file if you do not know what you're doing.

To run the script in Spyder (Anaconda3), click 'Run – Configuration per file' (Figure 5). You must do this each time you run the script in order to enter the proper arguments. Next a window with the Run Options will appear (Figure 6). Leave the defaults but select the box called 'Command line options' and type the following into the text box next to this option:

filename.txt 1.0 1.0

where filename is the name of your raw IRGA output file followed by the extension '.txt' (remember that there can be no spaces in your filename), and the first '1.0' is your threshold value discussed above in ppm of CO_2 (defaults to 1.0 ppm CO2). This value can be changed to what is best for your sampling run. The second '1.0' value is the frequency of measurements that the IRGA is running at (defaults to 1.0 second for LI-820 and 0.5 seconds for LI-830). After this is filled out click 'Run' in the Run Options window and the script will run in a few seconds (some trivial errors may appear even during successful runs). A new Excel file will be automatically created in the folder that you just created with the same filename as the raw IRGA output file but with the '.xlsx' extension.

This is your output file and should have the following structure:

group	**area**	**max_height**	**Time(H:M:S)**
0	103.693	49.343	19:02:53

Where:
there are 4 columns for group, area, max_height, and time. Groups are random ascending numbers (not important what the numbers are) areas and max_heights are large positive numbers (this is your data), time is an ascending clock times corresponding to when you sampled (this is when you sampled according to the lab computer clock).

You can then copy these ‘areas’ and ‘max_heights’ into your spreadsheet with the timestamps. If your spreadsheet has each sample entered in the same order that you sampled, then you can just copy and paste directly because the python script outputs the samples in the order that they were injected into the IRGA. ‘Areas’ are preferred as they are independent of injection speed, whereas ‘max_heights’ are not.

This is a highly flexible script if you know how to use Python. Any questions can be directed to Grant Falvo (falvogra@msu.edu).

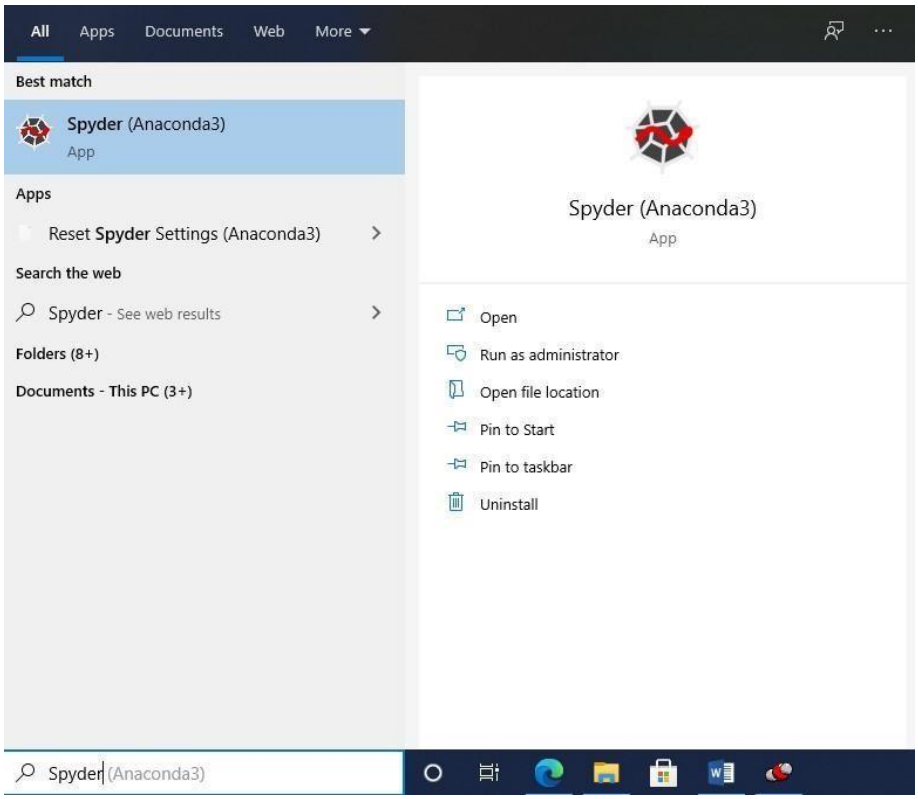

Figure 1.

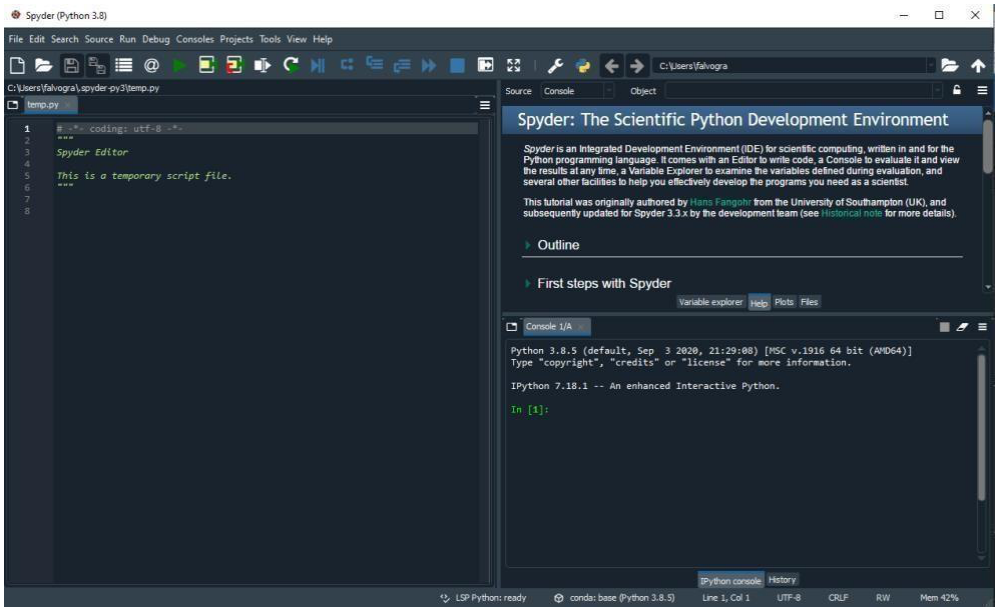

Figure 2.

Figure 3.

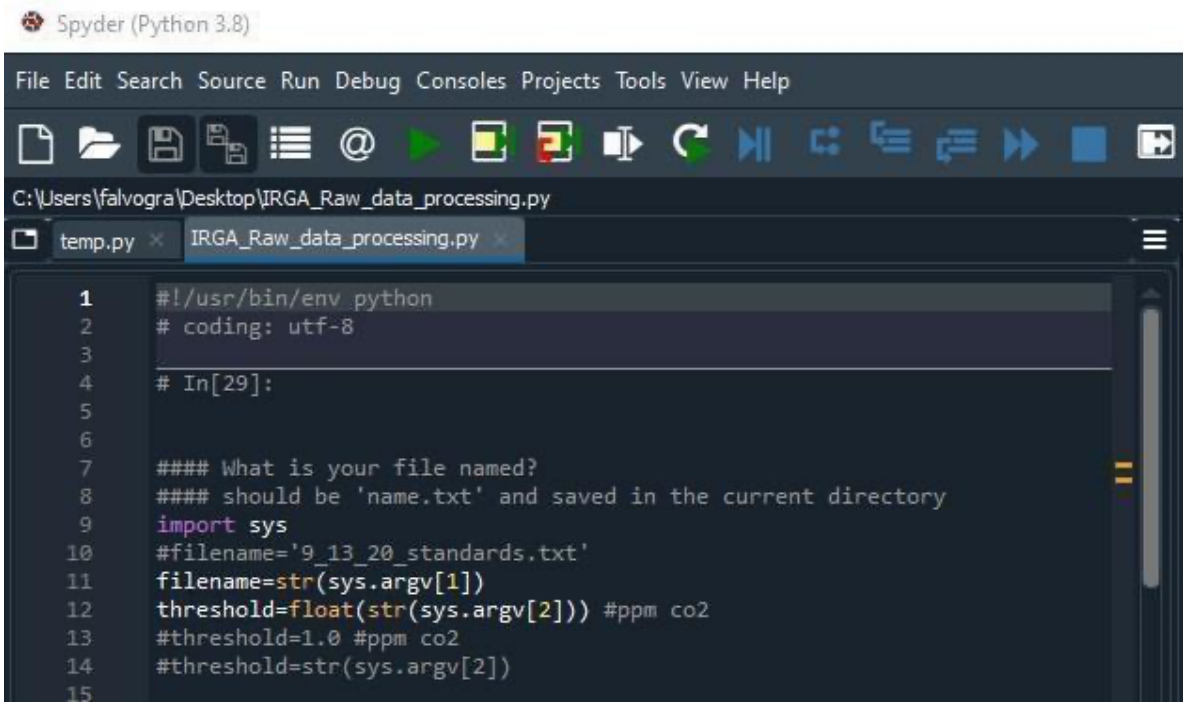

Figure 4

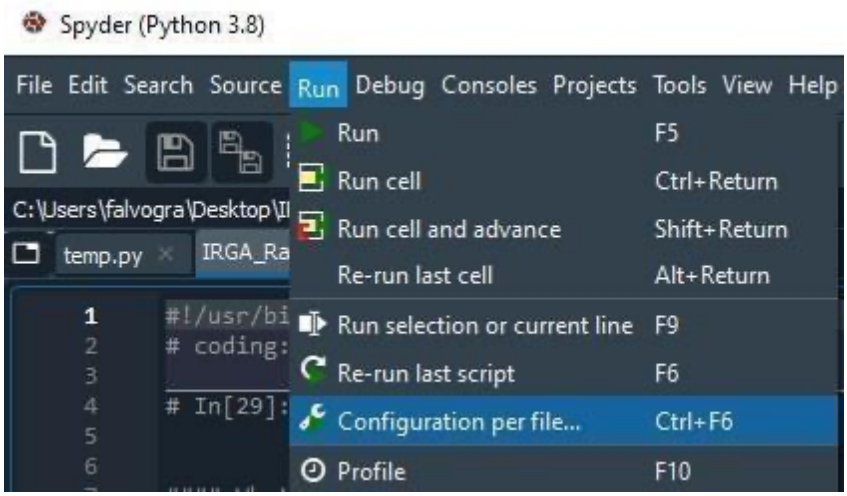

Figure 5

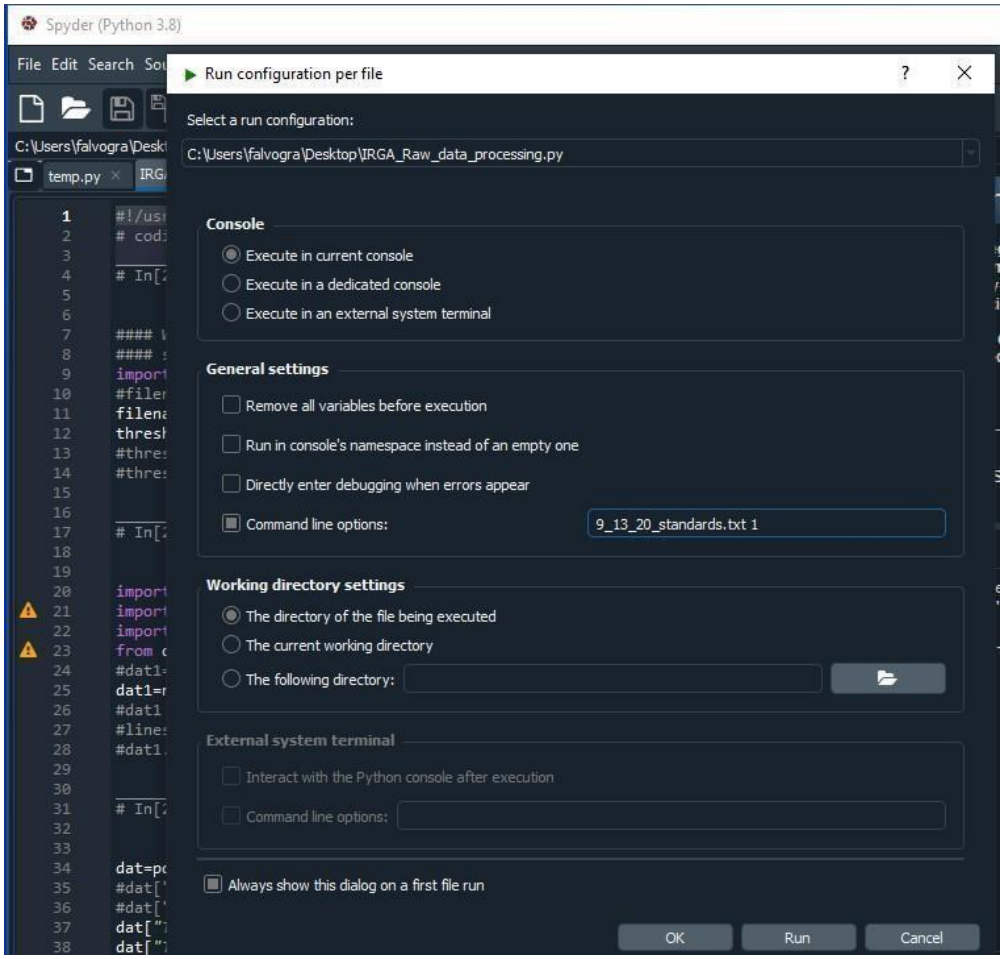
Spyder (Python 3.8)
File Edit Search
Run configuration per file
Select a run configuration:
C:\Users\falvogra\Desktop\IRGA_Raw_data_processing.py
Console
Execute in current console
Execute in a dedicated console
Execute in an external system terminal
General settings
Remove all variables before execution
Run in console's namespace instead of an empty one
Directly enter debugging when errors appear
Command line options:
9_13_20_standards.txt 1
Working directory settings
The directory of the file being executed
The current working directory
The following directory:
External system terminal
Interact with the Python console after execution
Command line options:
Always show this dialog on a first file run
OK
Run
Cancel

Figure 6

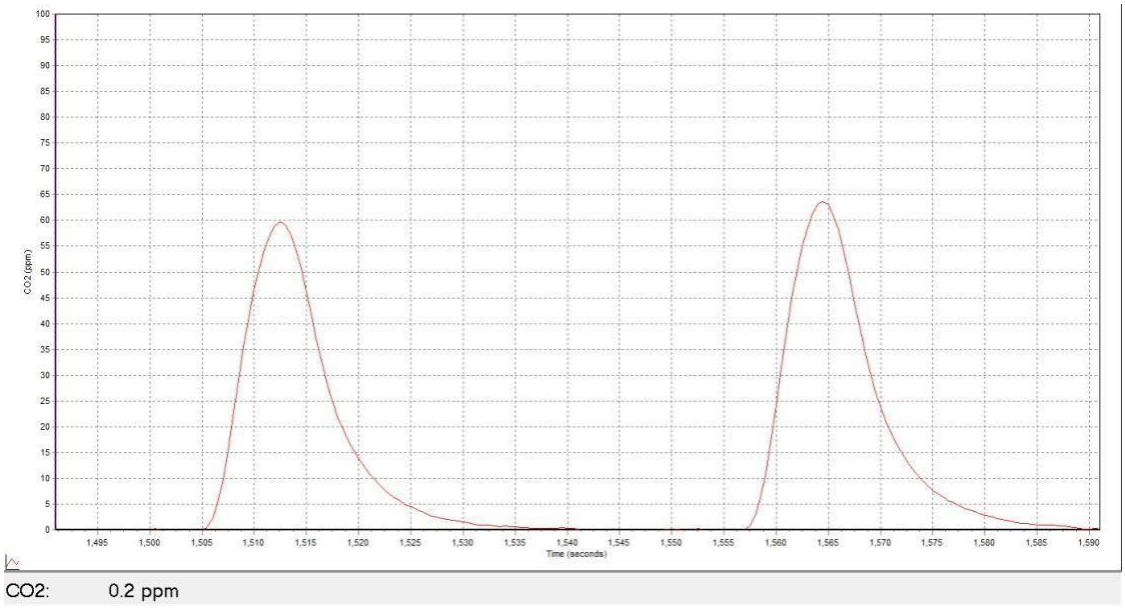
CO2 (ppm)
Time (seconds)
CO2: 0.2 ppm

Figure 7