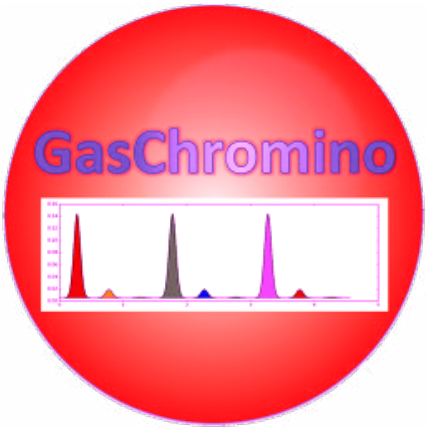


GasChromino Program

Description

The GasChromino Program is an interface program that allows a PC/Mac/Linux system to communicate with an Arduino that is connected to one or more gas chromatography instruments.

Installation instructions are included in the Installation appendix. Configuration instructions are included in the Configuration appendix.

Basic Use

This section outlines the general use of GasChromino to collect data from a single GC instrument. Use of multiple instruments is described below.

Upon opening the program, it should establish a connection to the Arduino, and it will show a successful connection in the lower right corner. If it fails to connect, see Connection to Arduino.

Prepare GC Prepare the GC by programming the temperature program, lighting the GC FID if necessary, etc.

Set time You should then set the *Length of GC experiment in minutes* to the length of time that the GC run will last. To account for differences in start times, make this slightly *longer* than the actual run-time of the GC to ensure that all data is acquired.

Set comment If you wish you could then set the *Comment for Experiment* that you want associated with this experiment.

Start experiment To prepare the Arduino for collecting data, click on the button Prepare Arduino. This stores the comment and sends the experiment time to the Arduino, but does not actually start the GC.

Make injection and Start data collection Once the GC is ready:

1. Make your injection
2. Start the GC by hitting the GC Start button
3. Start data collection by hitting the Start button on the GCArduino.
    a. There is no harm in starting the GC and Arduino in the opposite order as long as they are started close in time to each other
    b. Future implementation may involve remote start of the GC using the Arduino, but this is not yet implemented.

Observe data collection Once the Arduino start button has been pressed, the program will switch to the Live Data window and show the data as it arrives to the computer.

Stopping data collection The data collection will automatically stop when the time of the experiment entered earlier is reached. Alternatively, the stop button on the Arduino will also stop data collection.

Stopping the GC Stopping the GC is done in the normal fashion using the GC stop button. The GC is not and cannot be stopped by the Arduino.

Peak Processing

The GasChromino program can process GC traces using an automatic peak finding routine, but the user can also manually indicate peaks by marking the beginning and end of the peak. In both routines, the GC finds a baseline of those data points not indicated as peaks and uses this baseline to correct the area of the peaks. Finally, a normalized area is provided for each peak versus the total area of peaks. The user can remove all peaks or individual peaks as desired.

Automatic Processing The automatic processing routine utilizes two parameters to look for the start of peaks in comparison to a running baseline. If the slope at two points is greater than the *Peak Start Gradient Threshold* for two consecutive points (slope at point "b" and slope at point "c") and the value of the intensity at the second point (point "b") is greater than the *Peak Start Threshold*, a peak start is detected and marked at the previous point (point "a"), otherwise as long as point "b" is near in value to the running baseline, it is added to the running baseline and the routine moves to the next point. Once a peak start is detected, the routine finds to the top of the peak and detects the end of the peak in a similar fashion. For details of the peak detection routine, see the appendix on Processing details.

Thus, the two parameters to manipulate to change the automatic processing routine are the *Peak Start Threshold* and *Peak Start Gradient Threshold*. By making either of these values smaller, more peaks will be detected, and the user can use his or her own data or the Sample Data Trace provided to experiment with this routine. Changing the two parameters at any point prior to stopping an experiment will result in those values being incorporated in the routine. Alternatively, the data can be reprocessed subsequent to acquisition by hitting the Find and Integrate Peaks button.

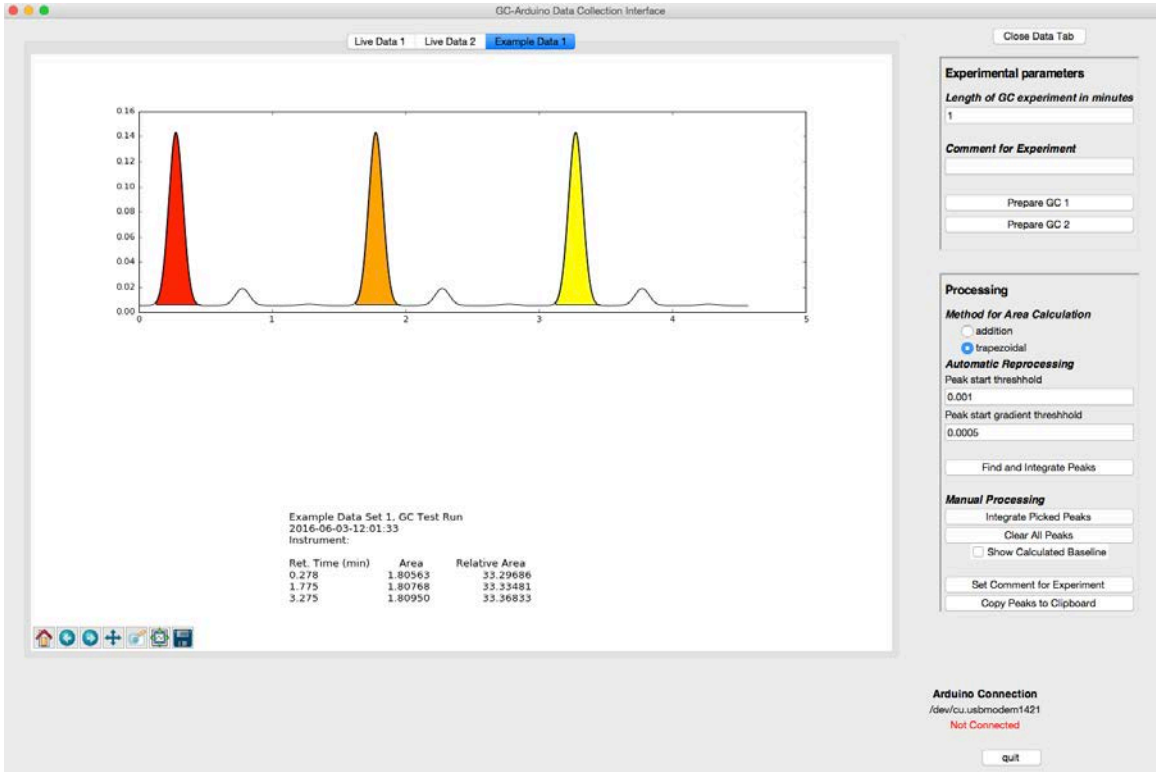
Manual Processing The manual processing routine can be done after automatic processing to either add or subtract peaks. Manual processing determines the baseline used to correct peak areas by using all points not included in peaks. Thus, it is very important that all peaks are included when using manual processing, otherwise the baseline correction can be quite faulty. This is especially true for large peaks like the solvent [see below for a method to remove solvent peaks from the listing of peaks].

Picking peaks To carry out manual processing to add peaks after most are detected using automatic processing, the user can Ctrl-Click and drag where the initial click is at the starting point and the release of the click is the ending point. Small red arrows on the trace indicate the beginning and end of indicated peaks. Processing of the picked peak is then effected by clicking on the Integrate Picked Peaks button. Additional peaks can then be added as desired either before or after clicking on the button.

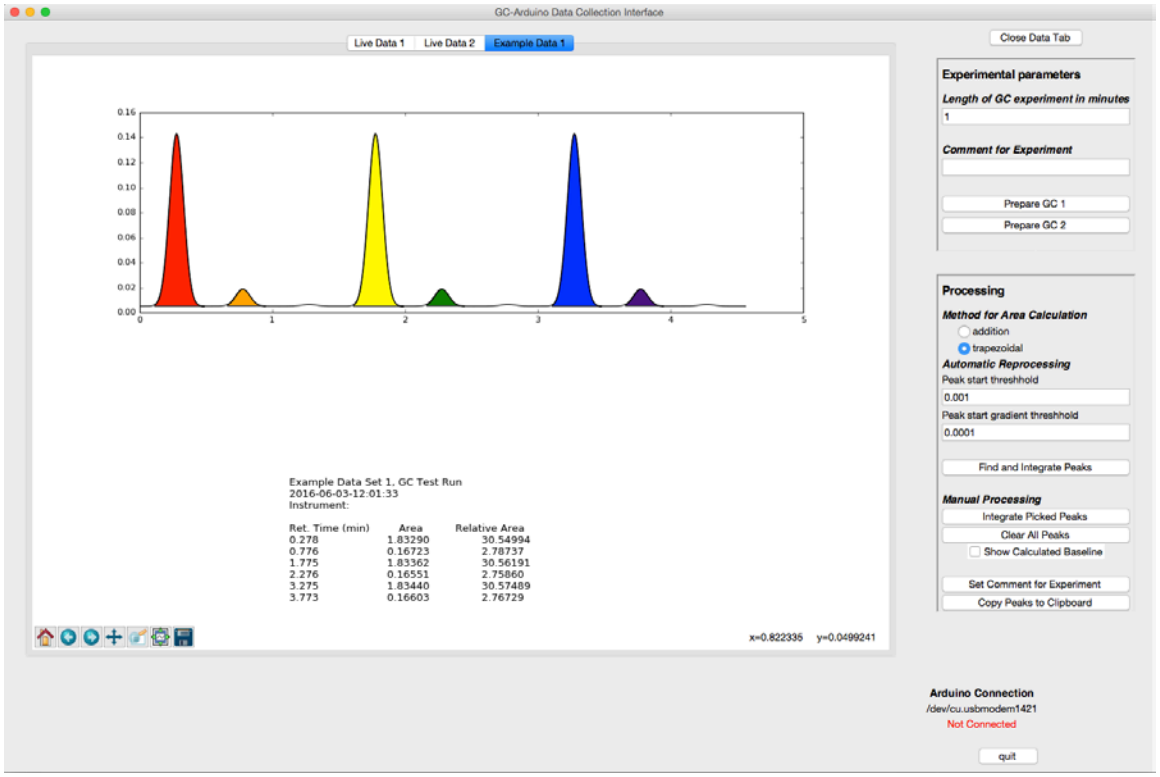
Deleting peaks Individual peaks can be removed from the peak list by shift-clicking while the cursor is within the peak area. This will remove the peak from the list and renormalize the peak areas. However, it does not add those points to the baseline calculation and thus is a way to remove large peaks from the integration and not affect the baseline.

Tutorial for Algorithm to Pick Peaks The following is a quick tutorial utilizing the provided data sets to explore the pick peaking routines.

1. Open the example data set Example Data 1.



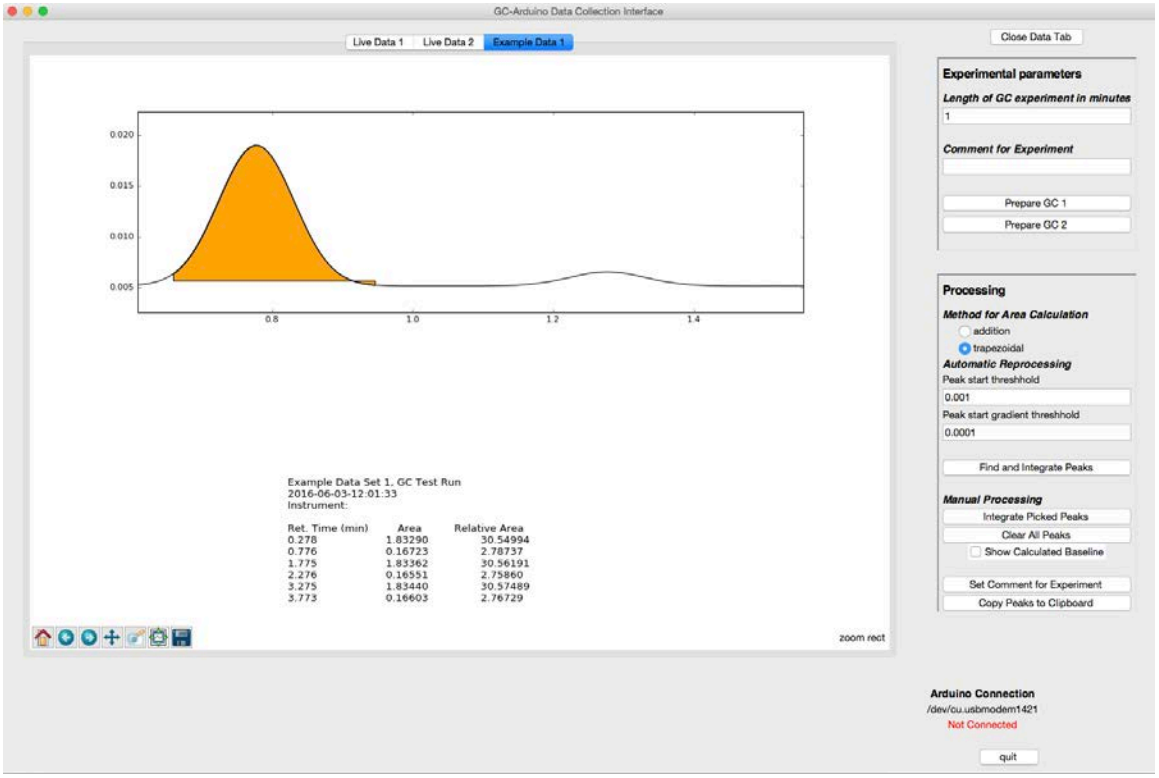
2. Change the *Peak gradient start threshold* to be 0.0001 (change 5 to 1)
3. Click on the Find and Integrate Peaks button



As you can see, the middle size peaks have been detected, but not the smallest peaks. Moreover, closer inspection of the middle size peaks will show imperfections in the baseline detection in this dataset.

4. Pick the Zoom to rectangle tool in the lower left corner of the data window

5. Draw a rectangle around the orange peak and the small peak to its right and release to zoom to that area.

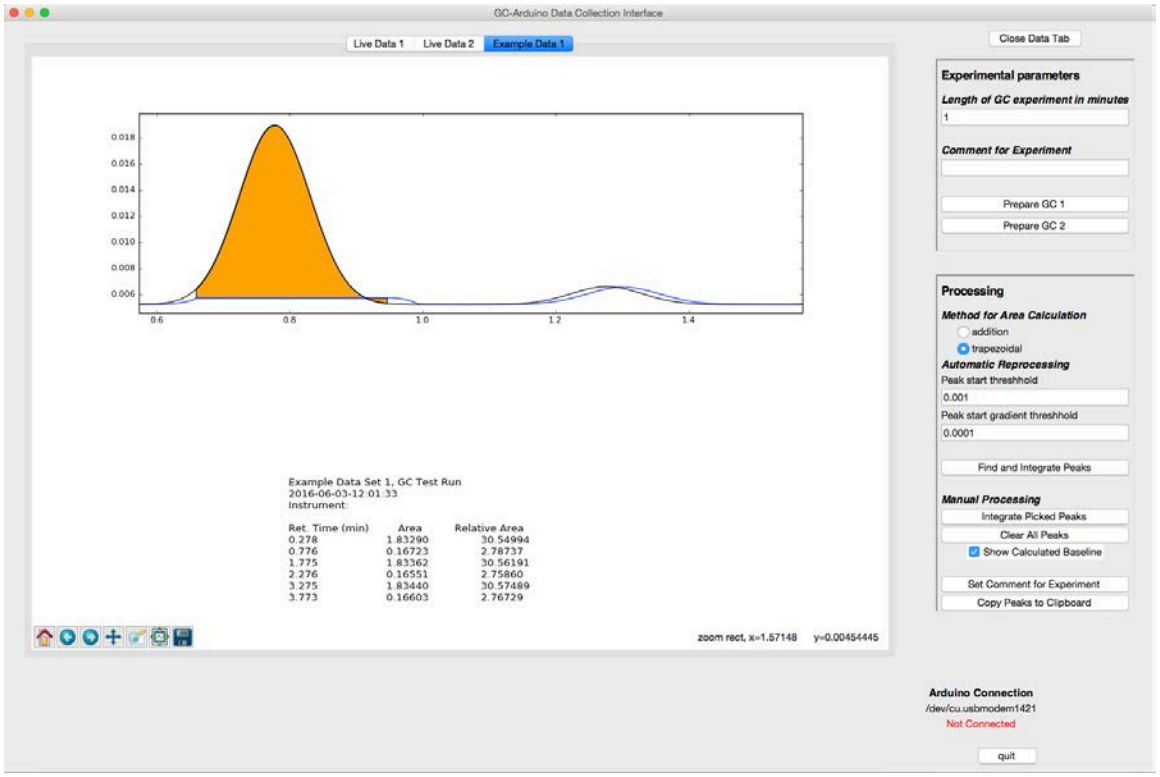


Note the small errors in the area indicated for the orange peak. The horizontal line of the bottom of the peak indicates the baseline that is being used to correct the integration of the orange peak. A small portion of the peak is missed at the beginning of the peak and because of that the baseline calculated by the automatic routine is a bit high.

One can fine-tune the peak detection further to fix small errors like this. By dropping the Peak start threshold and the Peak start gradient threshold, the definition of the baseline is refined and the shape of the peak integration improves.

A good way to understand better how the two thresholds are affecting the baseline and peak detection routine is to view the baseline (by default it is turned off). The toggle is just under the Clear All Peaks button.

6. Toggle the baseline by clicking on the checkbox Show Calculated Baseline.

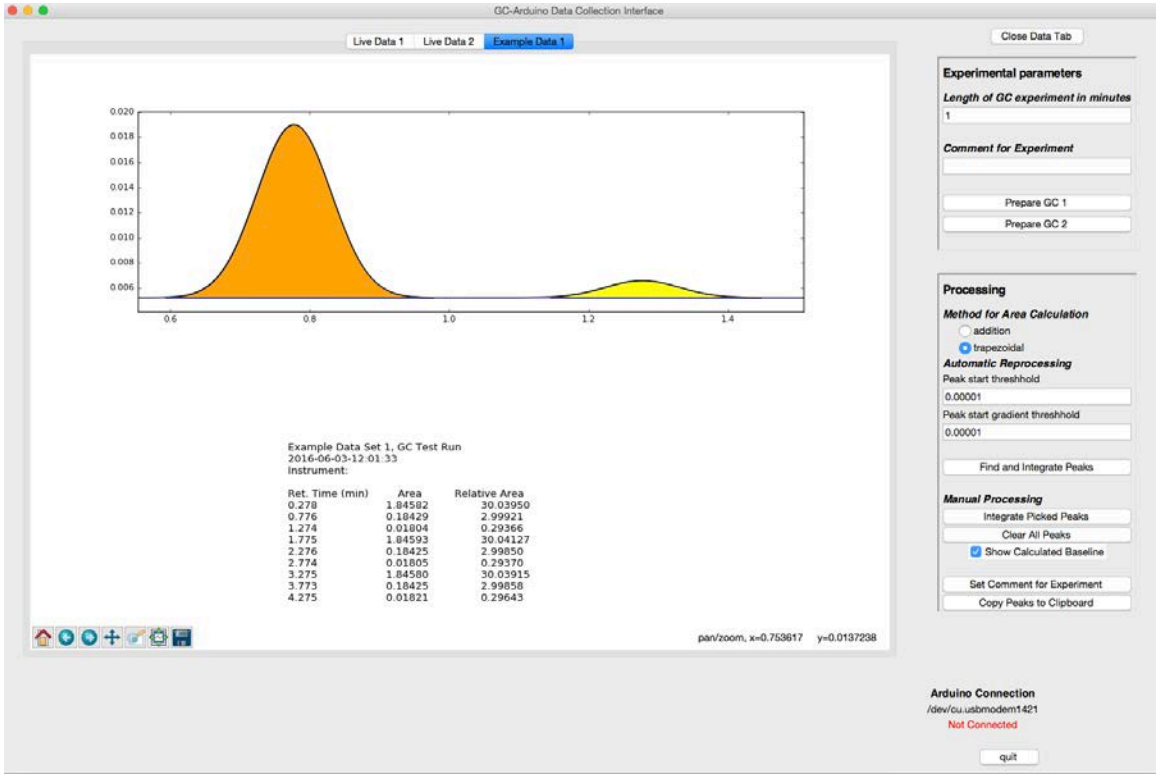


One can see how the detection of the beginning of the orange peak is late, resulting in a baseline that lags behind the GC trace at the beginning of the orange peak. Due to this it calculates a baseline that is actually higher than it should be resulting in the portion of the peak below the baseline at the tail of the orange peak.

Additionally, because the smaller peak on the right is so broad, the gradient threshold is not sufficiently low to detect it and the baseline ends up as a ghost of that peak.

The peak detection is significantly improved by lowering the two thresholds. Explore the effects of changing each of the thresholds on the peak detection routine.

7. Set both the *Peak start threshold* and the *Peak start gradient threshold* to 0.00001, hit Find and integrate peaks, and re-zoom.

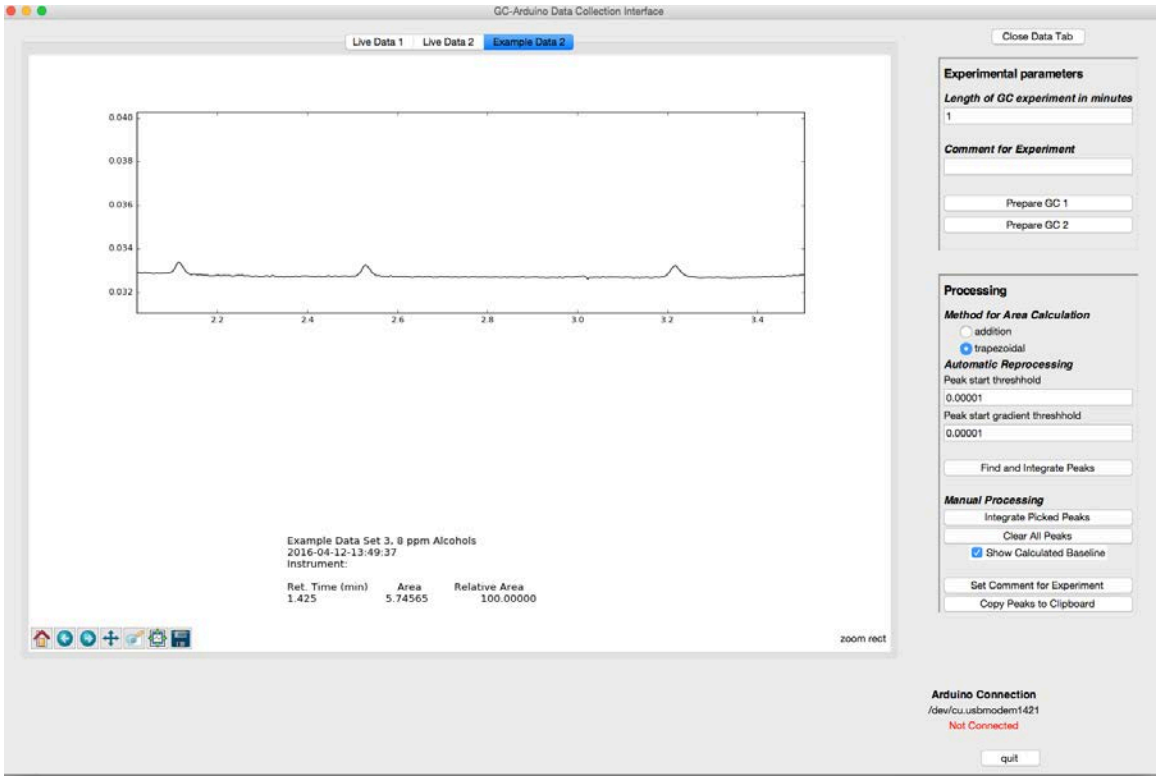


Now, the small peak is also picked and the baseline is much better defined as well.

As a measure of the accuracy of the GasChromino and this peak-picking routine, the ratio of the three peaks in a grouping should be 10:1:0.1.

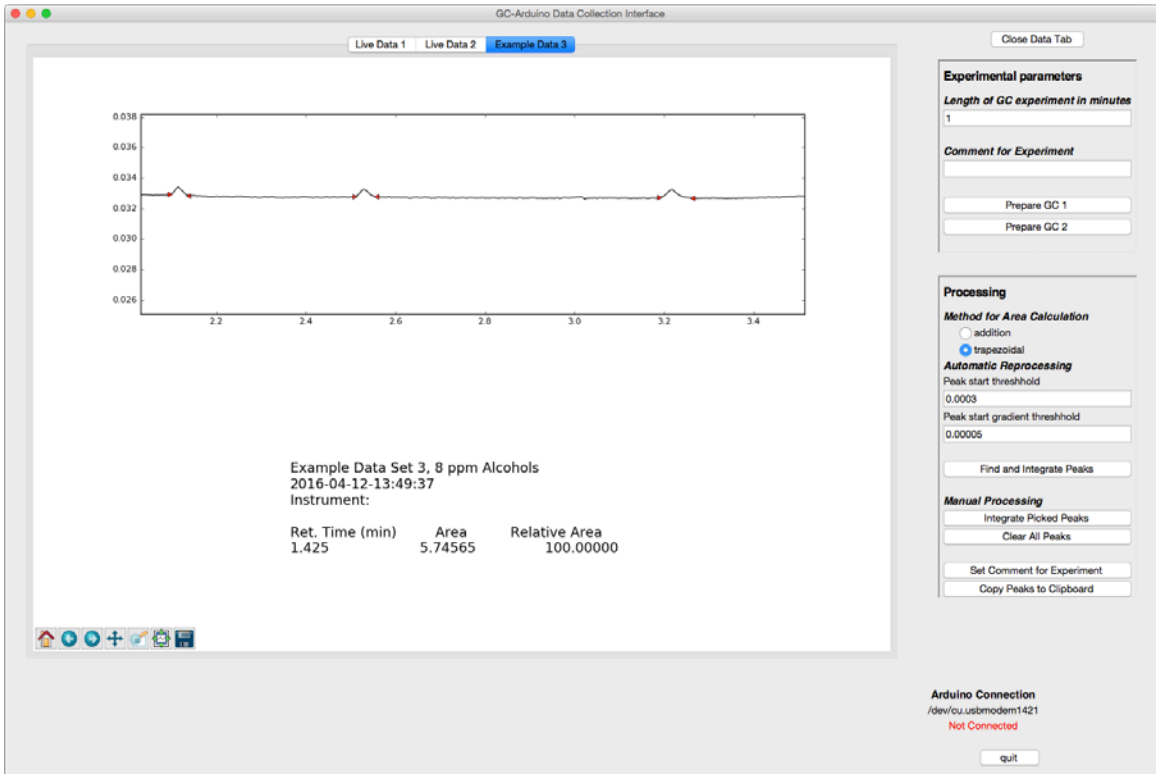
This methodology will not work particularly well with very small peaks, however. This is especially true when the baseline is sloping due to smaller peaks riding on the trailing edge of a much larger (e.g. solvent) peak. A more realistic dataset with very small peaks is provided as Example Data 3.

8. Open Example Data 3. Zoom near the baseline from 2 to 3.5 min and you will see small peaks. It may take multiple zooms to get a picture approximating that shown below.



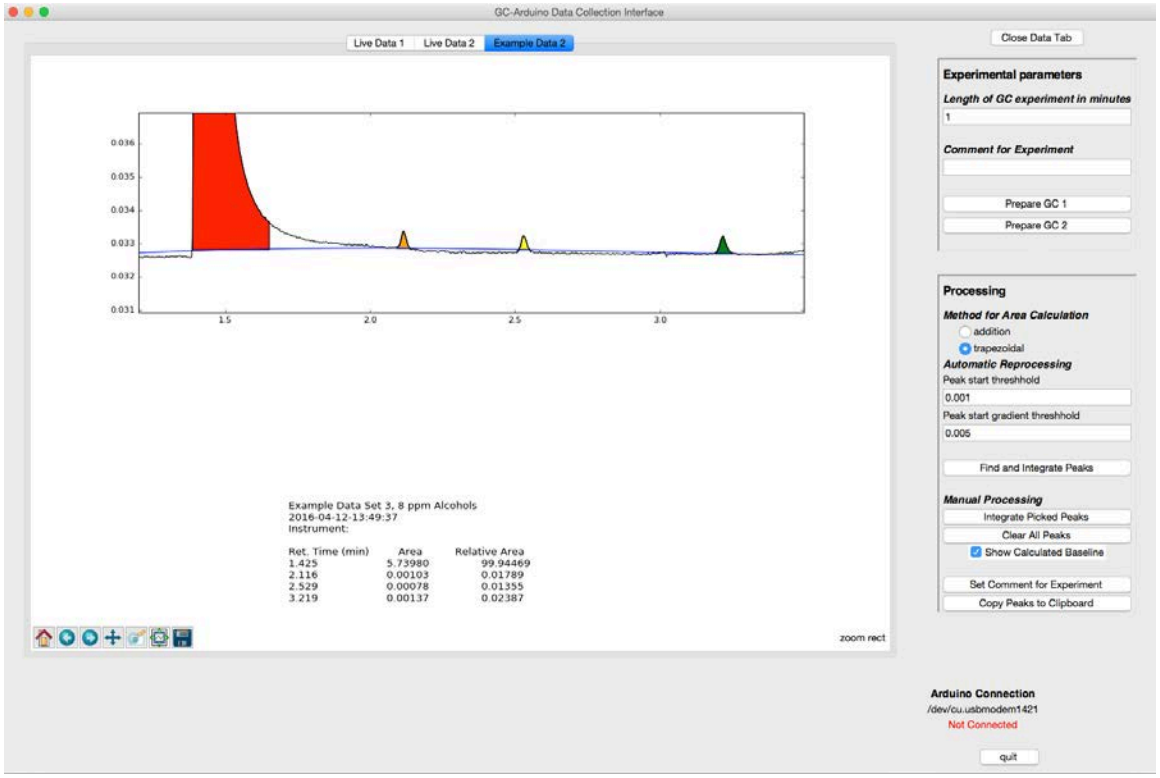
With careful choice of the automatic detection parameters, one could eventually pick these peaks in this particular dataset; however, it is much easier to pick them manually.

9. Make sure that you de-select the zoom to rectangle tool and then Ctrl-click and drag from left to right of the left-most small peak, releasing the click when you reach the point you decide is the right-most end of the peak. Repeat for the following peaks.



Note the small red triangles at the left and right of the small peaks. These mark the chosen left- and right-limits of each peak.

10. Click on Integrate Picked Peaks button to integrate the peaks. The screen will reset to see the entire GC trace.
11. After re-zooming to the region of interest (including the solvent this time) with baseline on, one can see the results of the picked peaks.

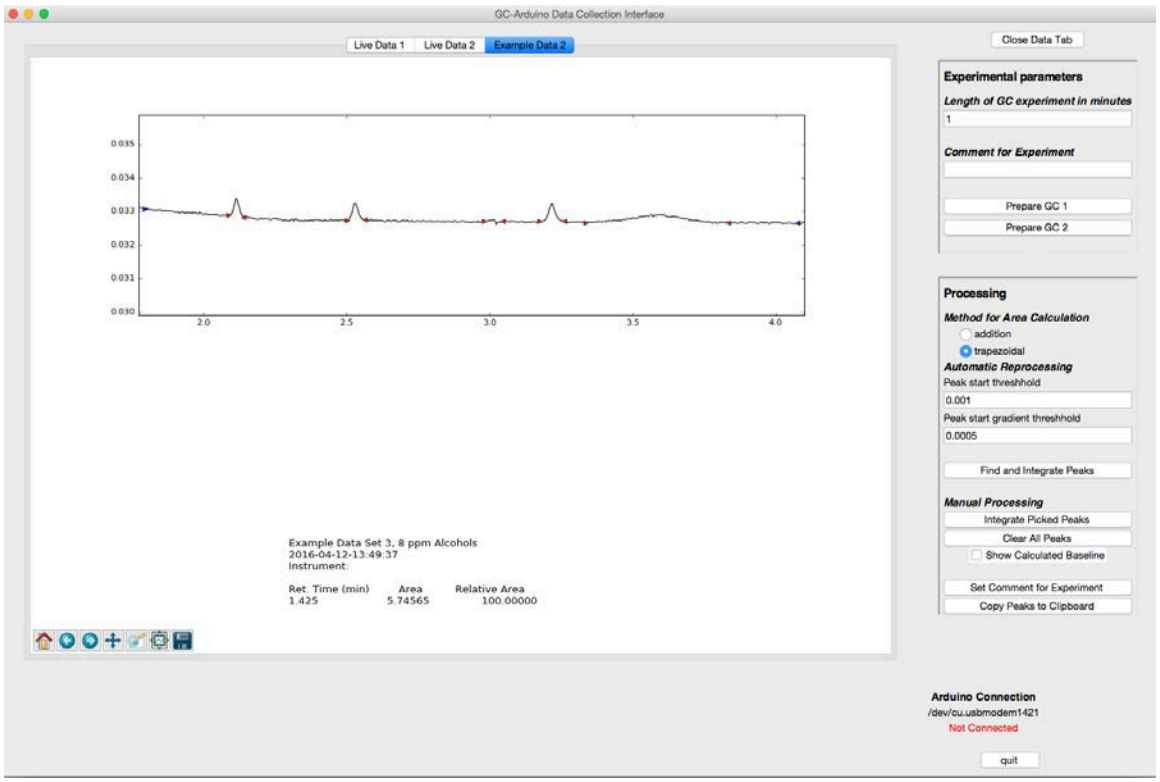


It is clear that the baseline determination is struggling due to the presence of the very large peak. As long as the large peak is not of interest, the program can follow the baseline of the trailing peak by telling it to limit the baseline calculation to near the area of interest.

To correct, complete the following steps.

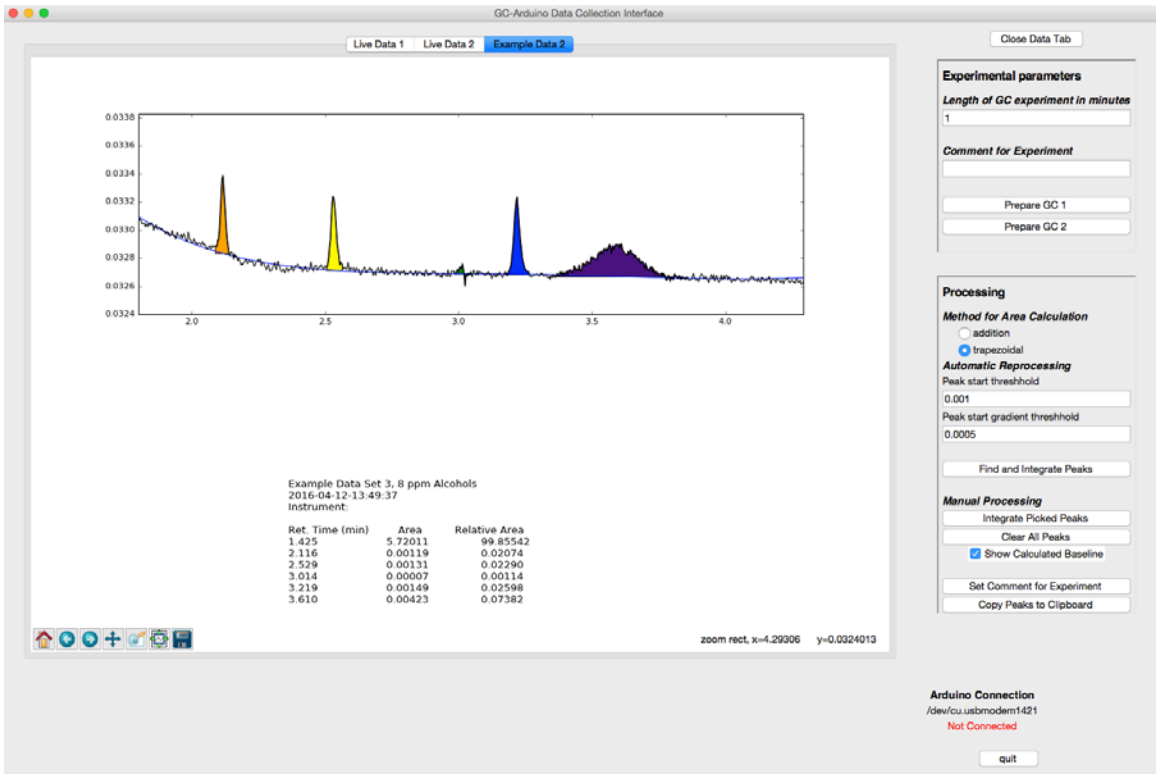
12. Hit Clear All Peaks button
13. Hit the Find and Integrate Peaks button to automatically detect the solvent (alternatively, one can simply reload the data by reopening it).
14. Re-zoom to the area of interest, this time from about 1.9 min to 4.1 min.
15. Pick the three peaks of interest, but also the little glitch at about 3.0 min and the very broad peak at about 3.5 min. It is important to identify all peaks so they do not get counted as baseline.
16. To limit the baseline calculation Ctrl-b Click and Drag (hit Ctrl+b at the same time you are clicking and dragging from the left side of the trace to the right).

    You should have a picture that looks like this.



The Blue triangles mark the beginning and end of the baseline area. Peaks will not be included in the baseline calculation.

17. Click on integrate picked peaks and turn on the baseline (if it is not already). Re-zoom to the area of interest.



One sees a much better fit of the baseline (in the area of interest), and the three peaks that nominally are 1:1:1 are much closer in area. If one looks at other parts of the GC trace, the baseline fit is very poor since the fit is based solely upon the single region that you have defined.

Presuming that the desired ratio is actually between the three small peaks and that the area of the solvent peak, the glitch and the broad peak are not of interest, one can delete the solvent peak and the other two.

18. Remove the glitch and the broad peak by Shift-Clicking while the cursor is on the peak to remove. You will need to re-zoom as the display resets after each deleted peak.
19. To remove the solvent peak, first, reset the trace by clicking on the home button in the toolbar:

20. Shift-Click on the solvent peak and the peak will be deleted, the areas updated, and the trace re-plotted. Below is the zoom of the three small peaks.

A third data set that has large (solvent), medium (peaks of interest), and small (impurities) peaks is provided as Example Data 2. Explore the peak-picking routines to practice with this data set.

Using Multiple Gas Chromatography Instruments

The GasChromino program is capable of handling two GC instruments at once, collecting data from both of them at the same time. Overall, the process is similar to that for collecting data from a single instrument, but one can start both instruments and observe data from each. Each time an instrument finishes a data collection, the data will be transferred to a new tab and automatically processed. You can switch back to the Live Data of the other instrument, other data tabs, or perform processing while the program is collecting data from the other instrument.

Hopefully, the two GCs are well labeled to make choosing the instrument in the program easier. See Configuration below for a description of how to change the labels on the Prepare Channel 1 and Prepare Channel 2 buttons to make them match your situation.

Prepare GC Prepare either (or both) GCs by programming the temperature program, lighting the GC FID if necessary, etc.

Set time You should then set the time of the experiment to the length of time that the GC run will last. To account for differences in start times, make this slightly *longer* than the actual run-time of the GC to ensure that all data is acquired.

Set comment If you wish you can then set the comment that you want associated with this experiment.

Prepare Arduino To prepare the Arduino for collecting data, click on the button Prepare GC 1 or Prepare GC 2. This stores the comment and sends the experiment time to the Arduino, but does not actually start the GC.

Differences for multiple instruments

When using two GCs, it will be necessary to set the parameters twice (if the time and comment parameters are to be different) and hit the Prepare Channel 1 or 2 for each instrument. Both instruments can be prepared prior to actually starting a run.

Thus, the order of steps is:

1. Set Time for GC 1
2. Set Comment for GC 1
3. Hit prepare button GC 1
4. Set Time for GC 2
5. Set Comment for GC 2
6. Hit prepare button GC

Make injection and Start data collection on first GC

Once the first GC is ready:

1. Make your injection on one of the two GCs
2. Start that GC by hitting its Start button
3. Start data collection by hitting the corresponding Start button on the GasChromino.

Observe data collection on first GC Once the Arduino start button has been pressed, the program will switch to the Live Data window for that GC and show the data as it arrives to the computer.

Make injection and Start data collection on second GC

Once the second GC is ready:

1. Make your injection on the second GC
2. Start that GC by hitting its Start button
3. Start data collection by hitting the corresponding Start button on the GasChromino.

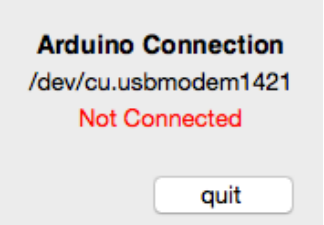
Observe data collection on first GC Once the Arduino start button has been pressed, the program will switch to the Live Data window for that GC and show the data as it arrives to the computer.

Stopping data collection for either instrument The data collection will automatically stop when the time of the experiment entered earlier is reached. Alternatively, the stop button for either of the GCs on the Arduino will stop data collection for that GC.

Stopping the GC Stopping the GC is done in the normal fashion using the GC stop button. The GC is not and cannot be stopped by the Arduino.

Connection to Arduino

If the program does not automatically recognize the Arduino, a pop-up window will appear indicating this and the bottom right corner of the main window will indicate that the Arduino is not connected. Additionally, the port that the program is attempting to connect to is provided. In the case shown below, this port is /dev/cu.usbmodem1421.



The failure to connect may mean that the Arduino USB cable is connected to the wrong USB port or not connected at all.

Troubleshooting

1. Check that the USB cable is connected to the Arduino and the computer
2. If both cables were connected, check that the port that the program is connecting to is correct.
    a. One can determine this port using the Arduino software
    b. One can also see which ports are in use by choosing the menu item Arduino Communications, submenu Set Arduino Port

This will provide a list of the ports that were in use when the program started.

3. If the Arduino is simply connected to a different port, one can use the menu item Arduino Communications/Connect to Arduino to attempt to reconnect.
4. If the cables were not connected, the Arduino port will not show up in the list of ports. [Currently the program only tests the ports upon start-up. Newly implemented: now can recheck, needs error testing.]
    a. If the port name is correct, one can still use the menu item Arduino Communications/Set Arduino Port even though the port name does not show up.
    b. If the port name is not correct, use the Recheck Port List submenu item. The port name should now show up and can be chosen.
    c. If the port still does not show up, there may be a problem with the Arduino, or possibly a bad cable. One could try resetting the Arduino and/or changing out the cable.

Installation

For general use, a zip file that includes everything necessary to run GasChromino is available from XXX.XXX. After downloading the zip file appropriate for your OS (currently only available for Mac OS X Yosemite), unzip and run the included gcSetup batch file. The gcSetup batch file will ask several questions regarding file placement and copy the files to the correct locations. After installation, the zip file and the initial directory it was unzipped into can be deleted. Details about individual OS's follow.

Prior Installations The gcSetup program recognizes when the application has been already installed, and checks to make sure that you want to reinstall. It will then overwrite files for the application and support files. Data files should not be affected, although the environment variables pointing to data file directories may be changed depending upon user choices during installation.

Mac OS X Installation

App Installation Directory The default location for installation of GasChromino is in the /Applications directory. This allows the program to be utilized by all users on the computer. If this is not desired, the user can type in another directory location. There is little error-checking at this point for this directory, so be careful.

Data Files Directory The GasChromino program can be set up to have either individualized data directories or a global data directory, depending upon the needs of the laboratory environment.

Global Data Location The default global data location is in /Users/Shared/Documents/GasChrominoData but this could be anywhere, even a networked location. The gcSetup program recognizes pre-set locations based upon

prior installations and suggests default locations. The user can accept either of these or specify another directory.

Local Data Location By utilizing the prefix $HOME followed by a directory path that would (will) appear within an individual's directory, the program will set the data path to each user's home directory. Upon initial running of the program, if this directory is not detected, it is assumed that the program has not yet been run and the directory is created and the Example Data Sets are placed in it.

Support Files Directory A directory is specified by the program that contains files utilized by the program. These support files include the Instructions (this document) and a configuration file (GasChromino.cfg). If on the first time that GasChromino runs it does not find these files (most likely gcSetup has not been run), then it will default to the version supplied with the program, and ask the user to edit the file to personalize the installation. This behavior can be repeated by deleting the GasChromino.cfg file.

Global Support Files Location The default global support files location is in /Library/Application Support/GasChromino but this could be any valid path. The gcSetup program recognizes pre-set locations based upon prior installations and suggests default locations. The user can accept either of these or specify another directory. In general, when the program is going to be used to control an instrument, a single GasChromino.cfg file should be utilized.

Local Support Files Location If the program is going to be used only as a data station on the installed computer, then individuals may want to customize their individual GasChromino.cfg. By utilizing the value $HOME in place of the directory path for the personalized support location, gcSetup will create the directory and place the Instructions.pdf file in it. GasChromino.cfg is then copied by the GasChromino app into the individual user's support directory the first time the program is run (or whenever it does not find a valid GasChromino.cfg file).

Environment Variables The environment variables that control GasChromino are set within the program's Info.plist file (with default installation found in /Applications/GasChromino.app/Contents). This file is modified by the gcSetup routine upon initial installation.

Advanced Environment Setup The gcSetup routine also sets several bash environment variables using the initial user's $HOME/.bash_profile file. Should the program need to be run from Terminal, these environment variables provide the necessary information. These variables are only set for the initial user, should all users need to run the program utilizing Terminal, these lines should be modified and added to each individual's $HOME/.bash_profile or in /etc/profile or /etc/bashrc file. This more advanced setup is left to the end-user, as it will vary from installation to

installation (and if you are fiddling with using it in Terminal, you probably know what to do anyway).

Configuration

The GasChromino program utilizes a configuration file, GasChromino.cfg, to hold important information about the program that the end user may want to modify. The typical global location for this file is /Library/Application Support/GasChromino, although installations may differ (see Installation).

The first time that GasChromino is run, the GasChromino.cfg file is opened for the user to edit. These instructions are also opened. Follow the instructions below for this initial editing. Further information about personalizing the installation and the structure of the GasChromino.cfg file follows.

Initial Configuration for an Instrument The following is a list of the typical changes that will be necessary upon initial installation of GasChromino for an Instrument. These changes are made by editing the GasChromino.cfg file during the setup process. Capitalization is important in the configuration file.

Table 1 Initial Configuration for an Instrument

Configuration option	Explanation
Instrument description	
dataStation = False	This installation of GasChromino will run an instrument.
noChannels = 2	[can be either 1 or 2] Number of Gas chromatography instruments that are hooked up to Arduino box.
instrName = [‘GC 1’, ‘GC 2’]	A list of names for the GC’s. These should be set to something that will make sense to users. The names must be quoted, and the number of names in the list should match the noChannels set above. Names should not be long.
Arduino serial communications	
arduinoCom = ‘/dev/cu.usbmodem1421’	This is the name of the serial usb port on the computer that the Arduino is connected to. This value will most likely change and needs to be determined with Arduino Software. Alternatively, GasChromino has the ability to detect ports that can then be selected within the software, but to set as a default, the GasChromino.cfg file must be edited. See Connection to Arduino.

Configuration option	Explanation
Defaults for basic experimental variables	
timeExper = “1”	This is the time (in minutes) of a GC run. This can be modified to allow for a different default length of run.
comment = “”	This is the default comment that will be added to the data after acquisition.
Processing variables	
thresh = 0.001	This is the default value for the threshold parameter for processing (see Peak Processing tutorial).
gradThresh = 0.0005	This is the default value for the gradient threshold parameter for processing (see Peak Processing tutorial).
Window visual appearance objects	
traceColor = ‘black’	Color of GC trace
baselineColor = ‘blue’	Color of calculated baseline
showBaseline = False	Baseline is not shown by default
colorList = ['red', 'orange', 'yellow', 'green', 'blue', 'indigo', 'violet']	Colors of peak areas highlighted by program. List cycles through colors. Colors can be added or removed or order changed. A single color would be repeated for all peaks.

Initial Configuration for a Data Station The following is a list of the typical changes that will be necessary upon initial installation of GasChromino for an Instrument. These changes are made by editing the GasChromino.cfg file during the setup process. Capitalization is important in the configuration file.

Table 2 Initial Configuration for a Data Station

Configuration option	Explanation
Instrument description	
dataStation = True	This installation of GasChromino will not run an instrument, but is only intended to process data.
noChannels = 1	Since no GC's are hooked up, there are no channels; however, the Data Station itself counts as one for naming purposes.
instrName = ['Data Station']	A name for the Data Station.
Processing variables	
thresh = 0.001	This is the default value for the threshold parameter for processing (see Peak Processing tutorial).
gradThresh = 0.0005	This is the default value for the gradient threshold parameter for processing (see Peak Processing tutorial).
Window visual appearance objects	
traceColor = 'black'	Color of GC trace
baselineColor = 'blue'	Color of calculated baseline
showBaseline = False	Baseline is not shown by default
colorList = ['red', 'orange', 'yellow', 'green', 'blue', 'indigo', 'violet']	Colors of peak areas highlighted by program. List cycles through colors. Colors can be added or removed or order changed. A single color would be repeated for all peaks.

Further Configuration Several other options, which most users will not be interested in changing, are available. For completeness, all user modifiable variables in GasChromino.cfg file are included here with commentary. The values in the section DO NOT CHANGE, should not be modified by the user.

Table 3 Complete Configuration Options

Configuration option	Explanation
Instrument description	
dataStation = True	This installation of GasChromino will not run an instrument, but is only intended to process data.
dataStation = False	This installation of GasChromino will run an instrument.
noChannels = 2	[can be either 1 or 2] Number of Gas chromatography instruments that are hooked up to Arduino box.
instrName = ['GC 1', 'GC 2']	A list of names for the GC's. These should be set to something that will make sense to users. The names must be quoted, and the number of names in the list should match the noChannels set above. Names should not be long.
Arduino serial communications	
arduinoCom = '/dev/cu.usbmodem1421'	This is the name of the serial usb port on the computer that the Arduino is connected to. This value will most likely change and needs to be determined by using the Arduino Software. Alternatively, GasChromino has the ability to detect ports that can then be selected within the software, but to set as a default, the GasChromino.cfg file must be edited. See Connection to Arduino.
timeout = 0.1	This is the length of time (in seconds) that GasChromino will wait for serial communications with the Arduino.
baudrate = 57600	Baudrate for serial communication with Arduino. If changed, it must also be changed in Arduino programming.
adcChoice = "ads1115"	Choice for analog to digital converter on the Arduino. The current circuitry only supports this option. If alternate

Configuration option	Explanation
	circuitry is provided, this value could be changed (concomitant with changes in the Arduino software) to allow the Arduino multiple options. A previous option of utilizing the Arduino's native adc is preserved, but not implemented in Arduino software.
adcChoices = {"arduino": "arduino", "ads1115": "ads1115"}	These are the list of choices for adc. Should modifications to the program be attempted, this is where the list of choices would be modified. For further information see source of software.
Defaults for basic experimental variables	
timeExper = "1"	This is the time (in minutes) of a GC run. This can be modified to allow for a different default length of run.
comment = ""	This is the default comment that will be added to the data after acquisition.
Processing Variables	
thresh = 0.001	This is the default value for the threshold parameter for processing (see Peak Processing tutorial).
gradThresh = 0.0005	This is the default value for the gradient threshold parameter for processing (see Peak Processing tutorial).
areaChoice = "trapezoidal"	This is related to the methodology to calculate the area under the peaks. Each peak is treated as a series of small trapezoids. The alternate choice here is "addition" which treats each point in the peak as a simple intensity and the intensities are added up. Trapezoidal integration provides more accurate areas typically.

Configuration option	Explanation
inBaseCt = 15	This is the number of points that are included in the running baseline for the automatic peak detection. The baseline value for any point is the average of the previous 15 baseline points. Thus, the program assumes that at the beginning of the trace, the first 15 points do not have a peak in them. Deviations from this assumption will result in poor peak detection.
Window visual appearance objects	
traceColor = 'black'	Color of GC trace
baselineColor = 'blue'	Color of calculated baseline
showBaseline = False	Baseline is not shown by default
colorList = ['red', 'orange', 'yellow', 'green', 'blue', 'indigo', 'violet']	Colors of peak areas highlighted by program. List cycles through colors. Colors can be added or removed or order changed. A single color would be repeated for all peaks.
mainwindSize = "1500x1200"	Size of main program window "WIDTHxHEIGHT" in pixels
mainwindPos = "50+50"	Position of main program window on initial running from top-left corner of screen.
liveframeHeight = 500	Initial height of frame for Live data from GC
liveframeWidth = 900	Initial width of frame for Live data from GC
dataframeHeight = 300	Initial height of frame for Data
dataframeWidth = 500	Initial width of frame for Data
mainwindTit = "GasChromino Data Collection Interface"	Title on top of the main program window