Igor ICARTT GUI documentation

Contents

1 Background

The GUI was written by Donna Sueper, with revisions from Ken Aikin and most recently Kyle Zarzana. Pedro Campuzano-Jost provided useful feedback and testing for version 3.xx of the code. The code is freely available, and we encourage Igor users to make their ICARTT files with the code.

1.1 Requirements

The GUI is compatible with Igor versions 7.0 and higher, and should work on both Windows and MacOS. The main restriction you will encounter if using Igor 7.x is that object names cannot exceed 31 characters. If you are using Igor 8.0 or later, then names up to 255 characters are allowed. At some point the GUI will no longer support Igor versions with the 31 character limit, at which point the major version number should be incremented.

Older versions of the code also required the General Macros ipf. Several of the functions contained in that ipf were replaced with native Igor commands, and the remaining functions were added to ICARTT GUI ipf. Functions that start with "`GM_`" were taken from the General Macros. The assumption is that the General Macro functions used by the ICARTT GUI are stable and will not need any bug fixes or updates, and including those functions in the ICARTT ipf makes the code more portable since you only need to keep track of one file.

1.2 Resources

1.2.1 Online

The document describing ICARTT V2.0 is available on the NASA website, where you can also find the standard for the now deprecated ICARTT V1.1.

NASA Langley provides several resources for checking ICARTT files after they have been generated. There is a web-based checker that you can use to check either a single file or all the files contained in a single .zip archive. The archive can have any name as long as there are no spaces or other weird characters. There is also an executable (Windows only) that you can download to check files offline. As of 7 August 2023, the latest version of the executable (2.3.4) does not seem to be completely compliant with ICARTT V2.0, but NASA plans to release a new version soon.

The biggest change to version 3.xx of the GUI is support for ICARTT V2.0, which, among other things, mandated the inclusion of Atmospheric Composition Variable Standard Names “in an effort to improve usability, standardization, and machine-readability.” The GUI does not provide support for making the names, so you need to refer to the NASA document standard names document for guidance.

While we have made every effort to make the GUI correct, please be aware that it may not work in all instances, as different data repositories have slightly different requirements. Additionally, data managers for different campaigns may have extra requirements, such as standardizing certain fields across all instruments. Please consult with the campaign data manager for guidance.

1.2.2 Within in the GUI

The GUI has several help buttons that open a notebook with additional documentation for various sections. The constant `kvDefault_Notebook_Size` sets the default font size in the help notebook and can be adjusted as needed. Additionally, many controls have help text that can displayed by hovering your mouse over the control. The help text usually gives the section of the ICARTT standards document relevant to that particular control.

The code has a variety of error checking to ensure that the created ICARTT file is compliant with the standards. However, the error checking can be fairly harsh, and the code will usually abort rather than trying to correct the issue (e.g. if there is an illegal character in the file name, the code will not try to replace it with a valid one). Some of the abort messages are a little cryptic, but they do include the function that generated the error and the line number if the user wants to see exactly where in the code the issue arose.

There are buttons at the top right of the GUI that you can use to open the pages referenced in §1.2.1.

1.2.3 Technical support

It is unclear who exactly is in charge of the code and where the master copy resides. If you do encounter an abort error that you believe is incorrect, please include the function and line number where the error occurred when you reach out to someone.

1.2.4 Other programming languages

Software for interacting with ICARTT files is also available for several other programming languages. Prof. Jose Jimenez (CU Boulder) has a list of several on his website. These include a MATLAB package written by Dr. Glenn Wolfe (NASA GDFL), an ICARTT reader in R by Prof. Steve Wofsy (Harvard), and a Python (project page) package from Prof. Christoph Knote (University of Augsburg).

2 Using the GUI to create files

The GUI relies on a variety of global strings and variables, as well as several waves. These reside in the data browser in a folder called `ICARTTFileFormat`. These objects are created by the code before it makes the GUI, and if this folder is deleted, the GUI will not function correctly. Since the objects are global, if you just kill the GUI and then recreate it, there might be some residual memory of the previous values. If you need to start over, the best practice is to close the GUI, delete the `ICARTTFileFormat` data folder, and then make a new GUI.

There are several constants towards the top of the ipf. These are all static, so they cannot be accessed by another ipf. Generally the user should not be modifying these. There are a few constants related to loading data that a user could change. One controls what happens if a data wave already exists when you load a new ICARTT file. The default is to overwrite an already existing wave, but this can be turned off. There is a constant that sets the wave note that is added to each loaded wave. This is a string that can be customized by the user. See §3.1 for more information. The third controls the default font size in the help notebook.

2.0 GUI creation

Load the ipf into the experiment and make sure it compiles. Go to the ICARTT menu and select one of the two "Make panel" choices. If you choose "Generic panel" an empty panel will be created, and you will have to fill in all the fields manually. If you have a previously created ICARTT file that you want to use as a template, select "Load header from existing file," which will prompt you for the file location, create the GUI, and fill in the fields from the template file.

There is a popup menu at the top of the GUI to select the version of ICARTT that you want to use.

2.1 Step 1: ICARTT data table

Create a table in Igor that has the waves you want to save. The waves will need to be in the correct order in the table (time waves, optional location waves, then data waves). If

Always be wary of any helpful item that weighs less than its operating manual - Terry Pratchett, *Jingo* the interval between data points is greater than 1 second, you will need to supply start and stop time waves, as well as a mid time wave if the midpoint is not the center between the start and stop time. If you are supplying multiple time waves, the order MUST be Start, then Stop, and then Mid. Position waves should be included if your data are not on the same timestamp as the measurement platform position data.

The names of the waves will become the "variable short names" in the ICARTT file, so make sure they are correct. For gas phase measurements, you are encouraged to use trace gas CoreNames given in Table 4 of the standard names document. If there are multiple instruments measuring the same species, it is best if the instrument name is included in the parameter name to avoid confusion (e.g. `NO_LIF` and `NO_CL` for nitric oxide measured by both laser induced fluorescence and chemiluminescence). Remember that other people will be using your data, so make things as clear as possible.

Once all the waves are in the table and in the correct order, adjust the format by right clicking on a column. The Igor default "General" format for numeric waves will not work. The acceptable formats are "Date and time" (for time waves), "Decimal," "Integer," or "Scientific." After you have adjusted the format, adjust the number of digits. It is recommended that you use 5 decimal places for aircraft latitude and longitude data. For other waves use a number of digits consistent with the uncertainty of the measurement.

Time waves are stored in the ICARTT file as seconds since midnight of the first day of the measurement. The code will do this conversion for you, so you can leave time waves in Igor time (seconds since midnight UTC, 1 Jan 1904). Also make sure that time waves have enough precision that for a given point the start time is less than the mid time which is less than the stop time.

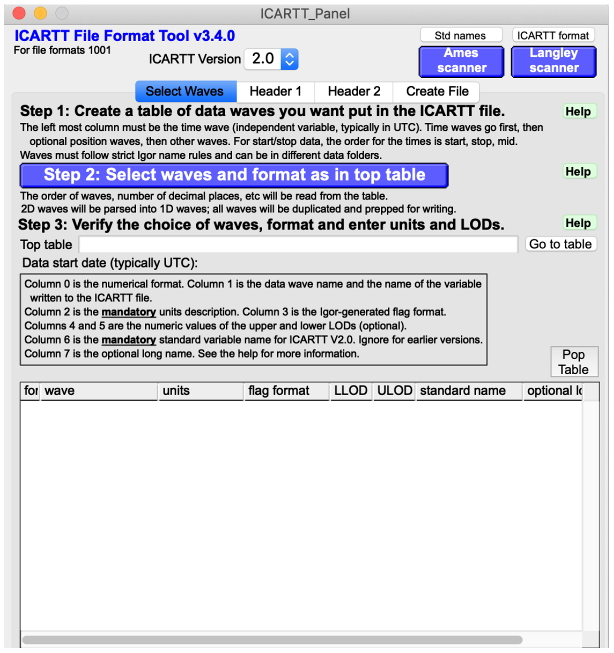
If you want to flag data points less than the lower LOD or greater than the upper LOD, you need to handle this now. Data points that are less than the lower LOD need to converted to `-Inf`, while points that are above the upper LOD need to be converted to `Inf`. The code will replace these values with the correct flag for upper and lower LODs. You will enter the values of the LOD at a later step.

2.2 Step 2: Wave selection and formatting

Click the blue "Step 2: Select waves and format as in top table" button. The ListBox at the bottom of the panel should populate with the information about the waves you are trying to save. The flag format will be determined based on the data in the waves, and should not be altered by the user.

The data start date field will be populated by the code. This must be the start date of the first time point, so it is not a user editable field.

if your mouse is over the listbox, you can use the mouse wheel to scroll both vertically or horizontally. If you hold down the command/control key when scrolling, you will scroll faster.



ICARTT Version 2.0

Step 1: Create a table of data waves you want put in the ICARTT file.
The left most column must be the time wave (independent variable, typically in UTC). Time waves go first, then optional position waves, then other waves. For start/stop data, the order for the times is start, stop, mid.
Waves must follow strict Igor name rules and can be in different data folders.

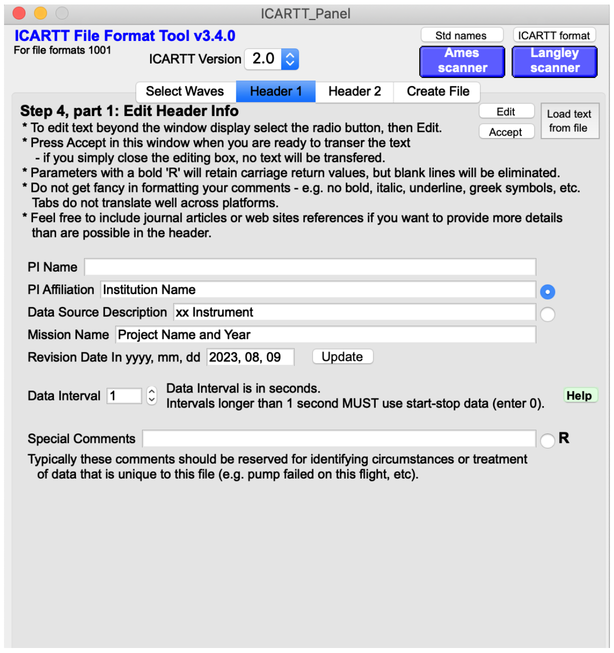
Step 2: Select waves and format as in top table
The order of waves, number of decimal places, etc will be read from the table.
2D waves will be parsed into 1D waves; all waves will be duplicated and prepped for writing.

Step 3: Verify the choice of waves, format and enter units and LODs.
Top table
Data start date (typically UTC):

Column 0 is the numerical format. Column 1 is the data wave name and the name of the variable written to the ICARTT file.
Column 2 is the **mandatory** units description. Column 3 is the Igor-generated flag format.
Columns 4 and 5 are the numeric values of the upper and lower LODs (optional).
Column 6 is the **mandatory** standard variable name for ICARTT V2.0. Ignore for earlier versions.
Column 7 is the optional long name. See the help for more information.

for	wave	units	flag format	LLOD	ULOD	standard name	optional lo

Select Waves tab. Choose the waves to include in the file, and then enter the units, LODs (if using), standard names (if using), and long descriptions.



ICARTT Version 2.0

Step 4, part 1: Edit Header Info
* To edit text beyond the window display select the radio button, then Edit.
* Press Accept in this window when you are ready to transer the text
- if you simply close the editing box, no text will be transfered.
* Parameters with a bold 'R' will retain carriage return values, but blank lines will be eliminated.
* Do not get fancy in formatting your comments - e.g. no bold, italic, underline, greek symbols, etc. Tabs do not translate well across platforms.
* Feel free to include journal articles or web sites references if you want to provide more details than are possible in the header.

PI Name
PI Affiliation: Institution Name
Data Source Description: xx Instrument
Mission Name: Project Name and Year
Revision Date In yyyy, mm, dd: 2023, 08, 09
Data Interval: 1 — Data Interval is in seconds. Intervals longer than 1 second MUST use start-stop data (enter 0).
Special Comments
Typically these comments should be reserved for identifying circumstances or treatment of data that is unique to this file (e.g. pump failed on this flight, etc).

Header 1 tab. Fill in the various fields, paying particular attention to the data interval.

2.3 Step 3: Units, LODs, and variable information

Enter the units for the waves in the units column. Enter "none" if the quantity is unitless.

If you are using LODs, enter the **values** of the lower and/or upper LOD into the appropriate columns. However, if your LOD filtering is more complicated than a simple threshold, you can leave the values as "N/A." You will need to explain how you are flagging the data, and the Langley scanner will warn you that the values are missing, but the files will still pass all the major checks.

If you are making ICARTT V2.0 files, you will also need to fill in the standard names based on the NASA rules. The standard names can only contain alphanumeric characters and underscores. The code will alert you if any illegal characters are present. Finally, you can add optional descriptions if desired, but if you add an optional description for one parameter, you have to add them for all parameters.

2.4 Step 4: Header info

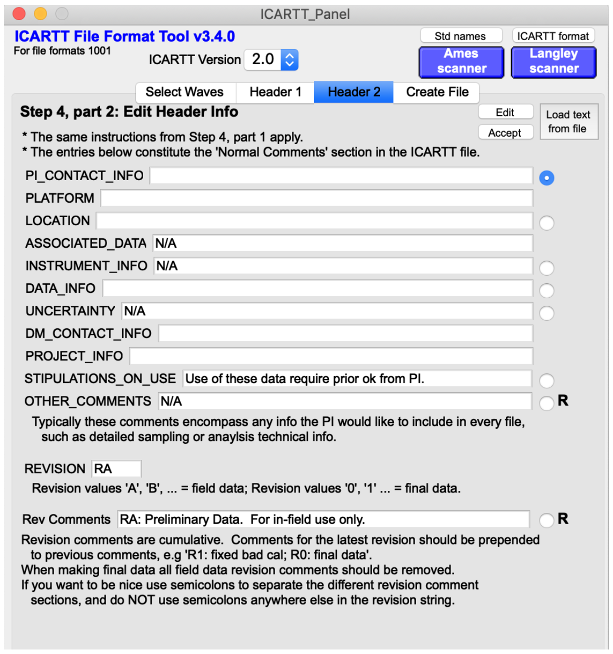
Go to the "Header 1" tab and enter values for the various fields. These hopefully are self-explanatory, and the help text for each field has some additional information. The Mission Name should be the same for all measurements from the project, so make sure to coordinate with the project data manager before making your files. Special comments are optional and are used for things that are unique for that particular file (e.g. pump failure, different ion source, etc).

For data collected continuously at a constant 1 Hz or higher frequency with a constant measurement sampling interval, measurements may be represented by a single timestamp.

Using this method, the reported timeline must be unbroken between the first and last reported measurements; missing data identifiers must be used to account for data gaps due to calibration or other periods of instrument down time. This method uses a data interval code equal to the interval between measurements, in seconds. For time series data, the data interval code is set to 1 for 1 Hz data and 0.1 for 10 Hz data. All intervals longer than 1 second must be reported with start and stop times, and the data interval code is set to 0. Mid time must be reported if it is not the exact midpoint of the interval.

Go to the "Header 2" tab and fill in the various fields. Ideally the platform name will be standardized across all the groups on the project. For the location, enter the lat/lon/elevation for fixed ground site, the location data file or dataID for the mobile platform, or description of lat/lon/altitude variables in the current file. Generally, if the measurement is on a different timestamp from the plane location data, the data file should then include location information on the same timestamp as the measurement.

At the bottom of the "Header 2" tab make sure the revision value and revision comments are correct. Letters starting from A are used for preliminary, in-field data, while integers starting from 0 are used for final data.



For file formats 1001

ICARTT Version 2.0

Step 4, part 2: Edit Header Info

* The same instructions from Step 4, part 1 apply.
* The entries below constitute the 'Normal Comments' section in the ICARTT file.

PI_CONTACT_INFO	
PLATFORM	
LOCATION	
ASSOCIATED_DATA	N/A
INSTRUMENT_INFO	N/A
DATA_INFO	
UNCERTAINTY	N/A
DM_CONTACT_INFO	
PROJECT_INFO	
STIPULATIONS_ON_USE	Use of these data require prior ok from PI.
OTHER_COMMENTS	N/A

Typically these comments encompass any info the PI would like to include in every file, such as detailed sampling or anaylsis technical info.

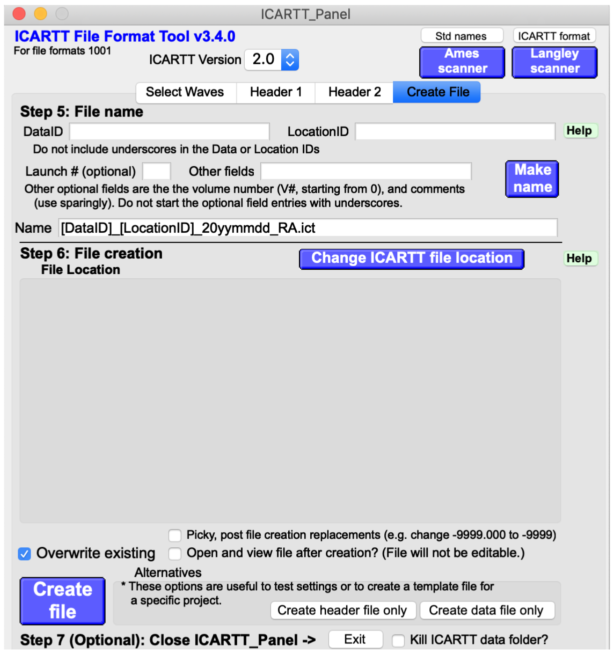
REVISION RA

Revision values 'A', 'B', ... = field data; Revision values '0', '1' ... = final data.

Rev Comments RA: Preliminary Data. For in-field use only.

Revision comments are cumulative. Comments for the latest revision should be prepended to previous comments, e.g 'R1: fixed bad cal; R0: final data'.
When making final data all field data revision comments should be removed.
If you want to be nice use semicolons to separate the different revision comment sections, and do NOT use semicolons anywhere else in the revision string.

Header 2 tab. Fill in the various fields, making sure the revision number and string are correct.



For file formats 1001

ICARTT Version 2.0

Step 5: File name

DataID　LocationID

Do not include underscores in the Data or Location IDs

Launch # (optional)　Other fields

Other optional fields are the the volume number (V#, starting from 0), and comments (use sparingly). Do not start the optional field entries with underscores.

Name [DataID]_[LocationID]_20yymmdd_RA.ict

Step 6: File creation

File Location

☐ Picky, post file creation replacements (e.g. change -9999.000 to -9999)
☑ Overwrite existing
☐ Open and view file after creation? (File will not be editable.)

Alternatives
* These options are useful to test settings or to create a template file for a specific project.

Step 7 (Optional): Close ICARTT_Panel ->
☐ Kill ICARTT data folder?

Create File tab. Make sure the name and file path are correct, and then click "Create file."

2.5 Step 5: File name

On the "Create File" tab, enter the DataID, the LocationID, and the launch number if needed. The DataID often contains the mission acronym, but this is dependent on the rules for the archive, and should be standardized across all the groups on the project. The LocationID should also be standardized across all the different instrument groups. The only allowed characters in each field are alphanumerics and dashes.

Always be wary of any helpful item that weighs less than its operating manual - Terry Pratchett, *Jingo*

Once you finish typing the DataID or LocationID, the GUI will check each entry to make sure there are no illegal characters. If there are errors, the box will turn yellow and there will be a warning popup. Once you have entered everything correctly, click the "Make name" button. Check the name to make sure it is right.

2.6 Step 6: File creation

Click the "Change ICARTT file location" button and navigate to the folder where you want to create the file. The default is to overwrite a preexisting file with the same name in the destination folder, but there is a checkbox that you can deselect to prevent this behavior. Regardless of whether this is checked, there will be a warning if a file with the same name exists in that location. Click the large "Create file" button to make the ICARTT file. If the file creation was successful, a message will print out in the command history. If there were issues, abort windows will appear with messages about what the problems were.

2.7 Step 7: File checking

Use one of the NASA file checkers to verify that everything is correct. This may seem redundant, but it is possible that the ICARTT files created by the GUI aren't 100% compatible with the NASA archive.

3 Loading ICARTT files

3.1 Wave note

When loading data from an ICARTT file, the code adds a wave note with useful information to each loaded wave. The fields that are always added to the note are the standard name (ignored for ICARTT V1.1), the long description, and the units. There is a string constant called `kstrWaveNote_Format` that lists the other fields that are included in the note. The default fields are the mission name, the revision date, the revision number, the revision string, the PI name and affiliation, and the data source description. The user can change this constant to include any of the global strings or variables. If you do change this constant, make sure to run the function `WaveNote_Checker`, which will make sure the new wave note format is valid. The only delimiters that are allowed are carriage returns (\r) and semicolons.

3.2 Overwriting existing waves

If waves with the same name already exist in the current data folder, the default behavior is to overwrite them. If you want to manually handle the duplicate names, set the constant `kvOverwrite_All_Duplicate_Waves` to 0.

3.3 Flagged data

When loading the files, all missing data (-9s) will be converted to `NaN`s. The default is to convert all LOD flagged data (-7s and -8s) to `NaN`s as well. However, if you set the constant `kvReplace_LOD_Infs_w_NaNs` to 0, the code will convert values flagged as being above the upper LOD (-7s) to `Inf` and values below the lower LOD (-8s) to `-Inf`. You generally should not need to change this.