Inlinino Documentation

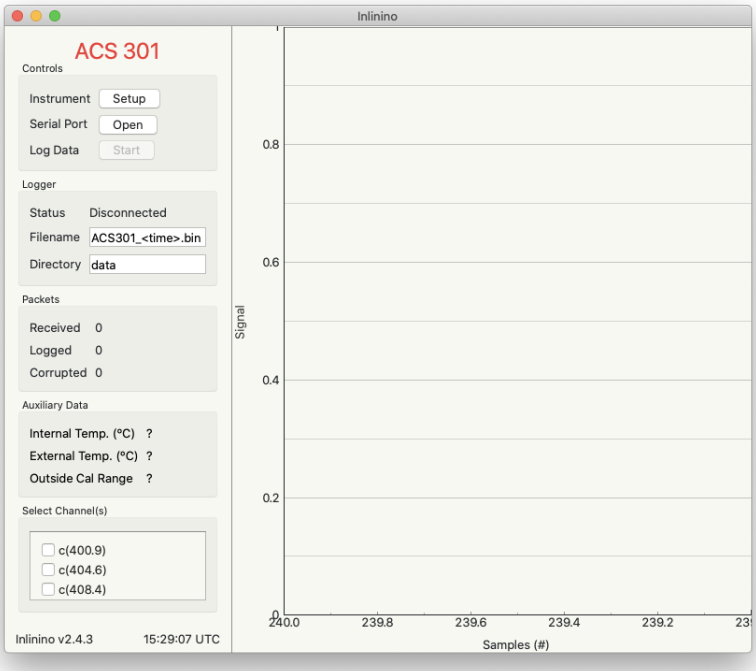
Release 2.4.3

Nils Haentjens

Sep 30, 2020

Contents

Inlinino is an open-source software data logger for oceanographers. It primarily log measurements from optical instruments deployed on research vessels during month long campaigns. Secondarily, it provides real-time visualization, which helps users troubleshoot instruments in the field and ensure collection of quality data. Inlinino is designed to interface with either serial (RS-232) or analog instruments. The data received is logged in a timestamped raw format (as communicated by the instrument) or in a comma separated file (csv) for easy importation in data analysis software. Typically, a new log file is created every hour for simplicity of post-processing and easy backups. Instruments supported are: SeaBird TSG, Satlantic PAR, WET Labs ECO sensors (e.g. ECO-BB3, ECO-FLBBCD, ECO-BBFL2, ECO-3X1M, ECO-BB9, ECO-BBRT), WET Labs ACS, Sequoia LISST, and analog sensors through a data acquisition system (DataQ DI-1100). Other instruments can be added via the user interface if they output simple ascii data frame, otherwise the code is intended to be modular to support new instruments. The use and validation of the software are documented in Haëntjens and Boss 2020 (DIY Oceanography).



CHAPTER 1

Index

1.1 Quick Start

Inlinino is available for Windows and macOS, the packaged executable can be downloaded with the links below. Windows users

- 2.4.3
- 2.4.3

Advance users or developers can setup the software directly from source (GitHub Repository).

Table of Contents

1.1.1 Startup Menu

To start Inlinino double click the executable: Inlinino-v2.4.3.exe on Windows and Inlinino-v2.4.3.app on macOS. When starting

- **Setup a new instrument:**
  1. Select the type of instrument to configure, the type of instrument available are listed in *Table 1*.
  2. Click on the setup button on the right side.
  3. A pop-up window will prompt you to configure the instrument, instructions for each type of instruments are available in the following sections.
  4. After validating the configuration of the instrument the main Inlinino window will be loaded.
- **Load a previously configured instrument:**
  1. Select the instrument from the dropdown menu.
  2. Click on the load button on the right side.
  3. The main Inlinino window will be loaded for the selected instrument.

Table 1: Table 1. Type of instruments supported

Instrument type	Instrument supported
acs	WET Labs AC-S or AC-9
dataq	DATAQ DI-1100
generic	Any sensor outputting simple ascii frame (e.g. *WET Labs ECO-Triplets*, Satlantic PAR, SeaBird TSG)
lisst	Sequoia LISST

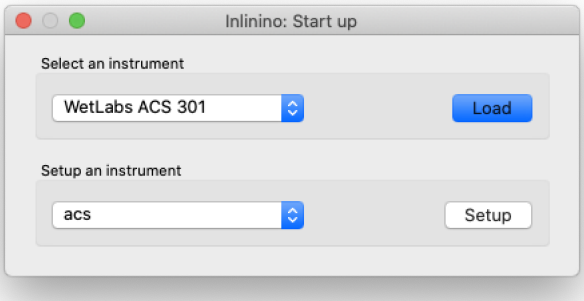


Fig. 1: Screenshot 1. Inlinino Start up window

Setup a generic instrument

Instruments are considered as generic if they send data as structured ascii frame. The frame is considered structured when each variable is separated by a separator (e.g. comma, tabulation) and the frame terminate with a terminator

(e.g. carriage return). An example on how to fill the form for a WET Labs ECO-BB3 is given in the *Screenshot 2*.

General section <top left> The manufacturer, model, and serial number of the instruments must be entered at the top left of the window.

Logger Group-Box <middle leftt>

It's possible to log the data in different format (select one or both options):

- *raw data*: exact output of the instrument to which a timestamp is appended
- *products*: variables of interest are extracted from the data frame received from the instrument and formatted into a comma separated value (csv) file with a timestamp.

The folder in which the data is logged is specified in the field *Log Directory*. The button *Browse* can be used to easily browse the computer file system and choose the adequate directory.

Parser Group-Box <left>

- *Frame Terminator*: indicate the end of the frame, hence the beginning of the next frame. Typically *rn* or *n*.
- *Frame Separator*: element separating values in frame. Typically , or *t*.
- *Variable Names*: list of variable names separated by commas.
- *Variable Units*: list of variable units separated by commas.
- *Variable Columns*: list of position of each variable in the frame.
- *Variable Types*: list of type of each variable. Can either be a floating number (*float*) or an integer (*int*).
- *Variable Precision*: list of string format used for each variables to write product log file. Typically *%d* for integers and *%.3f* for floating number with a precision of 3 decimal places.

Note: Variable Names, Variable Units, Variable Columns, Variable Types, and Variable Precision must be lists containing the same number of elements. All elements are separated by commas. Spaces are ignored. Special characters are not permitted.

Note: Note that the software record the date and time at which it received each data frame, hence no need to record the instrument timestamp which often drifts significantly during month long campaigns and is rarely set properly.

Below is an example of two data frames received from a WET Labs ECO-BB3 to illustrate this settings. Each variable is separated by a tabulation (*t*). The column 2, 4, and 6 correspond to the wavelengths of the sensor (note that the indexing start at 0 not 1). The column of interest with values corresponding to each one of the wavelength are 3, 5, and 7 (e.g. 255, 244, and 232 for the first frame). An example of variable names could be: beta470, beta532, and beta650. The variable units are: counts, counts, and counts in this case. Frames are separated by a new line character (*rn*). The date, time, and checksum (last number) would be ignored in these case. The *Screenshot 2* shows how to fill the form for this instrument and the data frames in question.:

```
11/08/16    20:33:49        470     255     532     244     650     232     524
11/08/16    20:33:50        470     263     532     251     650     237     588
```

Append prefix to log file Group-Box <left> This group-box is common to every instrument and explanations are provided in the section *Edit Instrument Configuration*.

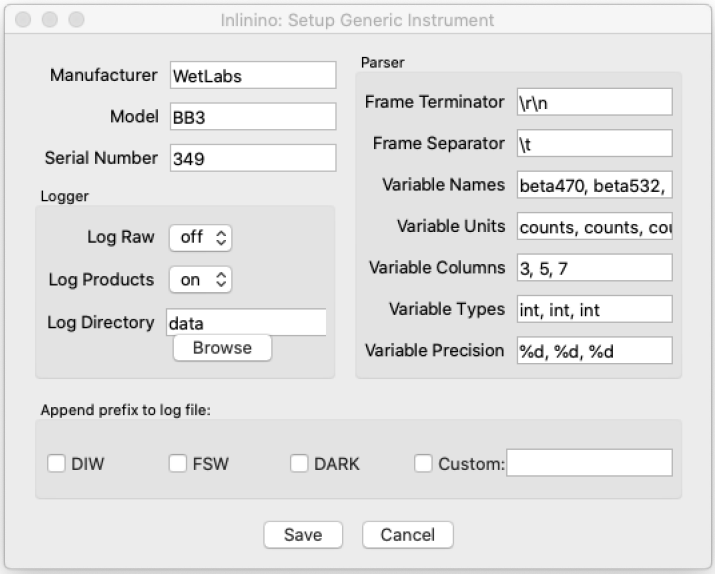


Fig. 2: Screenshot 2. Inlinino generic instrument setup window prefilled for a WET Labs ECO-BB3

Setup an analog instrument

Inlinino supports analog instruments by using a data acquisition (DAQ) module connected to a virtual serial port. The DAQ supported with the current version of Inlinino is the *DataQ DI-1100* which is commercially available. Previous version of inlinino supporting a custom made DAQ *PASC*, which is not supported anymore.

DataQ DI-1100

`General section <top left>` The manufacturer, model, and serial number of the instruments must be entered at the top left of the window. Special character are not supported and space will be ignored.

The folder in which the data is logged is specified in the field *Log Directory*. The button *Browse* can be used to easily browse the computer file system and choose the adequate directory.

`General Group-Box <top right>` Select the channels from DataQ DI-1100 from which instruments are connected and tension (in Volts) will be recorded. Unchecked channels will be disregarded (no data logged).

> **Note:** Selecting less channels increase the subsampling frequency (indirectly increasing the sampling resolution). Typically this DAQ is set to average all subsamples and log this average at 1 Hz.

`Append prefix to log file Group-Box <left>` This group-box is common to every instrument and explanations are provided in the section *Edit Instrument Configuration*.

PASC

The precision analog to serial converted (PASC) DAQ developed with the first version of Inlinino was not ported to the current version of Inlinino. For now, we recommend using the commercially available *DataQ DI-1100* DAQ supported by Inlinino.

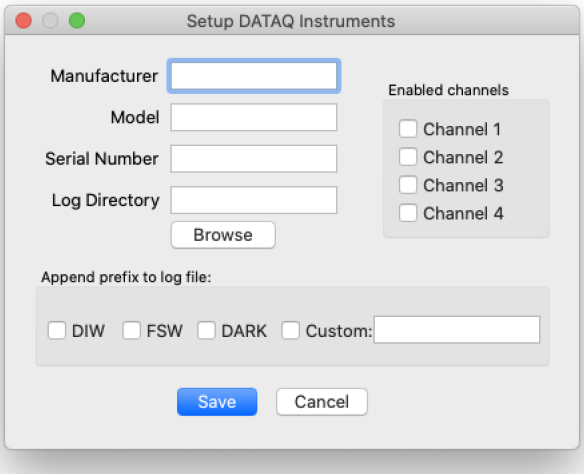


Fig. 3: Screenshot 3. Inlinino DataQ DI-1100 setup window for analog instruments

Setup a Sequoia LISST

Setting up the Sequoia LISST instrument is straightforward as all settings are contained in the manufacturer's device file (.txt) and the initialization file (.ini). The *browse* button on the right side of each field can be used to locate this files on the computer. Both of these files are required.

The folder in which the data is logged is specified in the field *Log Directory*. The button *Browse* can be used to easily browse the computer file system and choose the adequate directory.

`Append prefix to log file Group-Box <left>` This group-box is common to every instrument and explanations are provided in the section *Edit Instrument Configuration*.

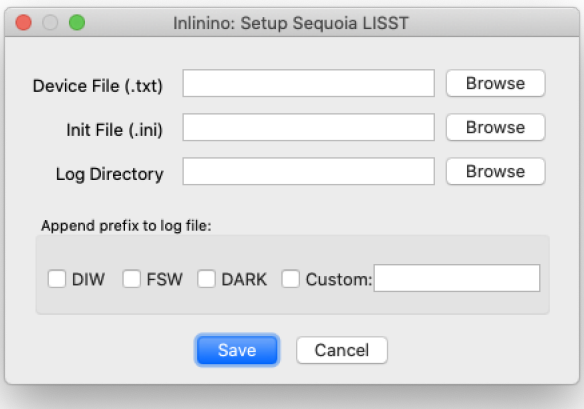


Fig. 4: Screenshot 4. Inlinino setup window for a Sequoia LISST

Setup a WET Labs AC-S

Setting up a WET Labs AC-S or AC-9 is simple as all settings needed are provided by the manufacturer in the device file (.dev). This file can be located on the computer with the *browse* button on the side of the Device File field.

The folder in which the data is logged is specified in the field *Log Directory*. The button *Browse* can be used to easily browse the computer file system and choose the adequate directory.

`Append prefix to log file Group-Box <left>` This group-box is common to every instrument and explanations are provided in the section *Edit Instrument Configuration*.

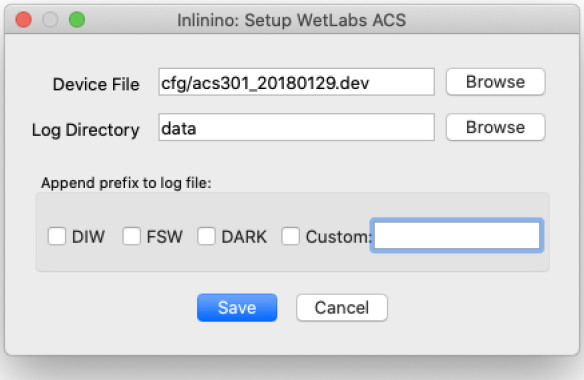


Fig. 5: Screenshot 5. Inlinino setup window for a WET Labs AC-S

1.1.2 Main Window

Once an instrument selected or setup through the startup window, the main Inlinino window will be loaded. The model and seri

- red: the instrument is not connected
- orange: the instrument is connected but not logging
- green: the instrument is connected and logging data

To log data follow the steps describe in the following section (*Log Data*).

To log data from multiple instruments simultaneously, start multiple instances of Inlinino. To do so, simply click on the Inlinino executable icon (.app on macOS or .exe on Windows) as many times as instruments to log. On the startup window select the appropriate instrument each time.

The last 120 values of selected channels are displayed in the plotting section of the main window (*Figure 7*) once the instrument is connected. On generic and analog instruments all channels are selected. On the WET Labs ACS and Sequoia LISST the user can select the channels of interest from the `Select Channel(s)` Group-Box menu at the bottom of the sidebar. By default, the latest channels selected by the users are plotted.

The `Packets` Group-Box of the sidebar displays in real-time the number of packets received, logged, and corrupted. When an instrument is turned on a few corrupted packets could be received, they are generally due to the instruments initialization message. If the number of corrupted packets keep increasing, a problem with the data format, the instrument settings, or the connections is occurring. Note that if the raw data is logged, the corrupted packets are logged but not timestamped. The raw data logging option is available in the setup menu of generic instruments. For other instrument types it's activated by defaults.

Instruments with many channels like the WET Labs AC-S and Sequoia LISST have an auxiliary window (*Figure 8*) which shows the latest value received of each channel. Instruments communicating auxiliary data (e.g. instrument temperature) have it displayed in the `Auxiliary Data` Group-Box of the sidebar.

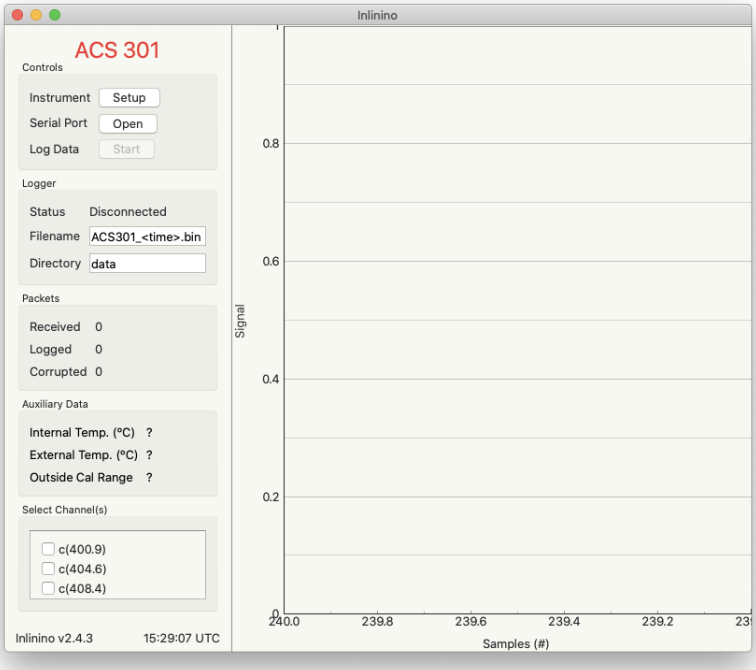


Fig. 6: Screenshot 6. Inlinino Main Window

Log data

1. Verify the configuration of the instrument by clicking on the setup button. Detailled instruction are provided in the, c.f. *next section*.
2. **Connect the instrument:**
    a. Click on `Open` button under the `Controls` section at the top of the sidebar.
    b. A pop-up window will prompt the serial port to which the instrument should be connected to.
    c. Once connected the instrument name at the top of the sidebar will change to an orange color, and data can be visualize on the figure section of the window.

> **Warning:** If no data can be visualized or the data makes no sense, the instrument is either incorrectly configured, their is a problem wih the connection, the instrument has an issue, or the instrument is not sampling properly (e.g. bubbles in the sample).

3. **Log data:**
    a. Click on `Start` button under the `Controls` section at the top of the sidebar.

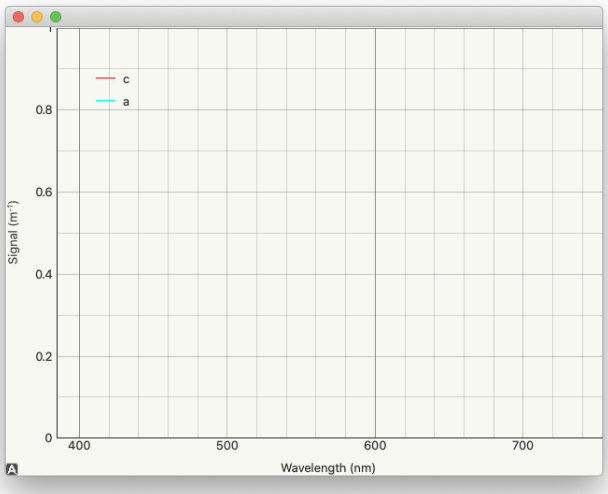


Fig. 7: Screenshot 7. Inlinino Auxiliary window specific to instruments with many channels (e.g. AC-S, LISST)

b. The name of the instrument will change to a green color. The filename under the `Logger` section will be updated to reflect the current file in which the data is logged. The filename is formatted as follow *<prefix><instrument_model><instrument_serial_number>_<YYYYMMDD>_<hhmmss>*. Prefix can be updated from the instrument *setup* (c.f. *Edit Instrument Configuration*). is Typically, a new file is created hourly, however, this can be adjusted through the configuration file.

Edit instrument configuration

Before logging data, it's important to check the configuration of the instrument, that the variables recorded are correct or that the calibration files are up to date.

To edit the configuration of an instrument, Inlinino must be started with the instrument loaded. Click the `Setup` button under the `Controls` Group-Box at the top of the sidebar. The setup window will pop-up. To update settings follow instructions specific to each instrument's type (c.f. *Table 1*).

`Append prefix to log file Group-Box` Data filenames follow the syntax `<instrument_model><instrument_serial_number>_YYYYMMDD_HHMMSS.csv`. For example a file for a WET Labs ACS serial number 301 created July 9, 2016 at 16:01:00 UTC would be named: `ACS301_20160709_160100.csv`.

One might want to append a prefix to a data file name logged with Inlinino, in the event of instrument calibration or some experiments. To consistently append a prefix to the name of a file logged with Inlinino, select the prefix desired at the bottom of the setup window (*Screenshot 8*).

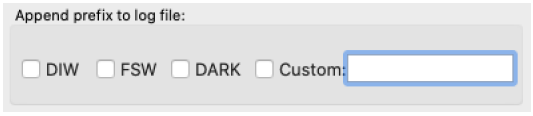


Fig. 8: Screenshot 8. Append prefix to log file name.

> **Warning:** The `Append prefix to log file` settings are only applied to the current Inlinino session and are reset when Inlinino is restarted.

Next step

Additional settings are explained in the Configuration section.

1.2 Configuration

Essential settings are explained in the *Quick Start* Section. More advanced settings are detailed below. All parameters editable through Inlinino's user interface are saved in the inlinino_cfg.json file. This file is located in Inlinino's root directory on Windows and under Inlinino-v2.4.3.app/Contents/Resources/inlinino_cfg.json on macOS. The configuration file follows json syntax. The section *instruments* contains an array with the settings of each instruments configured to date. Some parameters are *common* to every instrument type while others are *specific* to each instrument. *Example of configuration* for each instrument type are listed at the end of the chapter.

> **Warning:** Be careful when manually editing the configuration file. Breaking the json syntax might prevent Inlinino from starting. It's recommended to backup the configuration file before making any modifications.

Table of Contents

1.2.1 Common parameters

List of parameters common and required by every instrument type.

`module: <string>` The module parameter refers to which parent-class needs to be loaded to communicate with an instrument. The modules/types of instrument implemented are:

- generic: Class used for most instruments outputting simple ascii frames.

- dataq: Specific to the DataQ DI-1100 data acquisition module to log analog instruments
- lisst: Specific to Sequoia LISST instrument
- acs: Specific to WET Labs AC-S and AC-9 instruments

manufacturer: **<string>** Instrument manufacturer. This field can only contain the following characters: A-Z, a-z, 0-9.

model: **<string>** Instrument model. This field can only contain the following characters: A-Z, a-z, 0-9.

serial_number: **<string>** Instrument serial number. This field can only contain the following characters: A-Z, a-z, 0-9.

log_path: **<string>** Path to the directory in which the data from the instruments is logged.

Note: On Windows, the path need to include two backslashes as they are special characters in JSON. For example: `C:\\Data\\Inlinino"`.

---

log_raw: **<boolean>** Indicate if log the raw data coming from an instrument.

Note: For the ACS this option logs the binary data received from the instrument. It is highly recommended to set it to *True*, as it allows to reprocess the raw data in case of parsing issues with Inlinino. By defaults it is enabled when using the user interface.

---

log_products: **<boolean>** Indicate to log data received in a comma separated value file, easily read by data analysis software.

Note: For the ACS on long cruises (e.g. month, year), one might want to desable this parameter as the volume of data collected is significantly higher when enabled

---

1.2.2 Specific Parameters

List of parameters specific to an instrument type/module.

Generic Instruments

terminator: **<dict>** Indicate the end of the frame, hence the beginning of the next frame. For example:

```
{"terminator": {
  "__bytes__": "ascii",
  "content": "\r\n"
}}
```

separator: **<dict>** Element separating values in frame. For example:

```
{"separator": {
  "__bytes__": "ascii",
  "content": "\t"
}}
```

variable_names: **<list>** List of variable names separated by commas.

variable_units: **<list>** List of variable units separated by commas.

variable_columns: **<list>** List of position of each variable in the frame.

variable_types: **<list>** List of type of each variable. Can either be a floating number (*float*) or an integer (*int*).

variable_precision: **<list>** List of string format used for each variables to write product log file. Typically *%d* for integers and *%.3f* for floating number with a precision of 3 decimal places.

Note: All list must have the same number of elements.

Analog Instruments

channels_enabled: **< list >** List of analog channels to log data from.

```
{"channels_enabled": [1,2]}
```

Sequoia LISST

device_file: **< string >** Path to device file, also referred as instrument file, from the manufacturer.

```
{"device_file": "cfg/LISST1183_20180119_InstrumentData.txt"}
```

ini_file: **< string >** Path to initialization file (.ini) from the manufacturer.

```
{"ini_file": "cfg/LISST1183_20180119_Lisst.ini"}
```

WET Labs ACS

device_file: **< string >** Path to device file from the manufacturer.

```
{"device_file": "cfg/acs301_20180129.dev"}
```

1.2.3 Example of configurations

Generic Instruments

Example of configuration for a WET Labs ECO-BB3.

```
{
  "manufacturer": "WetLabs",
  "model": "BB3",
  "serial_number": "349",
  "module": "generic",
  "terminator": {
    "__bytes__": "ascii",
    "content": "\r\n"
  },
  "separator": {
    "__bytes__": "ascii",
```

(continues on next page)

(continued from previous page)

```
    "content": "\t"
  },
  "variable_names": ["beta470", "beta532", "beta660"],
  "variable_units": ["counts", "counts", "counts"],
  "variable_columns": [3, 5, 7],
  "variable_types": ["int", "int", "int"],
  "variable_precision": ["%d", "%d", "%d"],
  "variable_displayed": ["beta470", "beta532", "beta660"],
  "log_raw": false,
  "log_products": true,
  "log_path": "data"
}
```

Analog Instruments

Example of configuration for a DataQ DI-1100.

```
{
  "module": "dataq",
  "manufacturer": "WetLabs",
  "model": "WSCD",
  "serial_number": "859",
  "log_path": "data",
  "log_raw": false,
  "log_products": true,
  "channels_enabled": [2]
}
```

Sequoia LISST

Example of configuration for a Sequoia LISST.

```
{
  "manufacturer": "Sequoia",
  "model": "LISST",
  "serial_number": "1183",
  "module": "lisst",
  "ini_file": "cfg/LISST1183_20180119_Lisst.ini",
  "device_file": "cfg/LISST1183_20180119_InstrumentData.txt",
  "log_raw": true,
  "log_products": true,
  "log_path": "data"
}
```

WET Labs ACS

Example of configuration for a WET Labs ACS.

```
{
  "manufacturer": "WetLabs",
  "model": "ACS",
  "serial_number": "301",
```

(continues on next page)

(continued from previous page)

```
  "module": "acs",
  "device_file": "cfg/acs301_20180129.dev",
  "log_raw": true,
  "log_products": true,
  "log_path": "data"
}
```

1.3 PASC

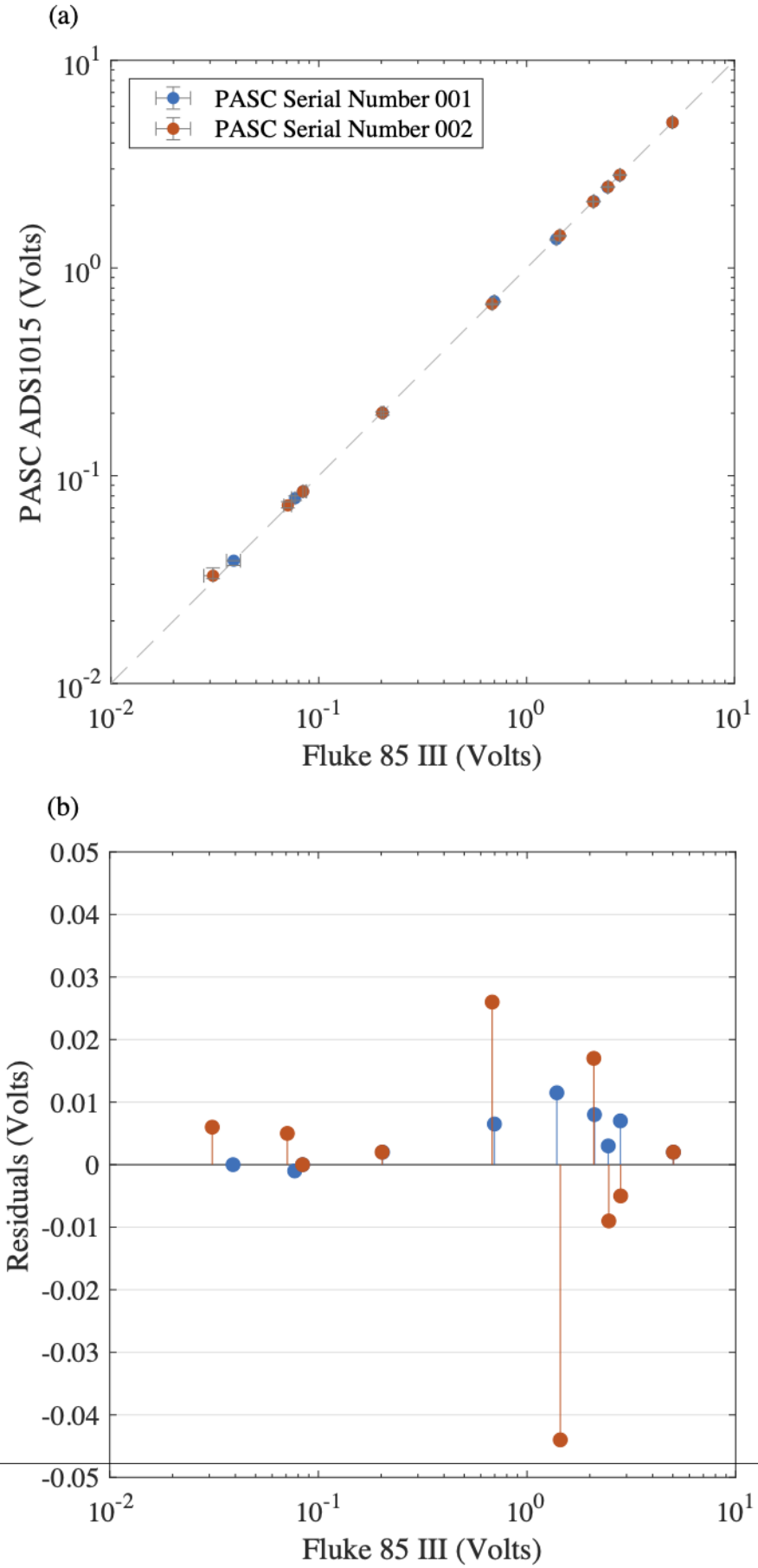
The precision analog to serial converter (PASC) is an optional data acquisition (DAQ) device. PASC is only required to log data from instruments communicating through analog channels. PASC can be built with an Arduino Uno type microcontroller and a precision analog to digital converter such as the Texas Instrument ADS1015 or ADS1115 developpement boards. The wiring instructions to build your own are available at Adafruit website.

We uploaded the firmware to the microcontroller following these instructions.

1. Load mcu_firmwares/PASC.cpp in the Arduino IDE:
    1. In ~/Documents/Arduino create a folder PASC/
    2. Copy and rename mcu_firmware/PASC.cpp to ~/Documents/Arduino/PASC/PASC.ino
    3. Load PASC.ino from Arduino Software (File > Open...)
2. Comment/uncomment appropriate lines in PASC.ino following instructions in comments of the file.
3. Compile and upload PASC to the microcontroller (button on the top left of Arduino IDE).

1.3.1 PASC Precision and Accuracy Validation

The precision and accuracy of the PASC serial number 001 and 002 were assessed with a Fluke 85 III Voltmeter. We found no significant bias and a reasonable root mean square error of (5.3 mV), the data is presented in the Figure below.


(a)
PASC Serial Number 001
PASC Serial Number 002
PASC ADS1015 (Volts)
Fluke 85 III (Volts)
(b)
Residuals (Volts)
Fluke 85 III (Volts)

1.3.2 Configuration in previous versions of Inlinino

The parameters required to setup the PASC with the previous version of Inlinino are:

- module
- name
- frequency
- gain (for ADS1X15 only)
- variables
  - pin
  - units

`frequency: < int >` Frequency (in Hertz) at which the Arduino will be reading and reporting voltage.

Note: Theoretical maximum sampling frequency are:

Uno	ADS-1015	ADS-1115
9600	3300	860

Maximum frequency taking into account conversion delay:

Number	Uno	ADS-1015		ADS-1115	
of PIN	*SE*	*SE*	*DIF*	*SE*	*DIF*
1	50	1000	500	125	62
2	25	500	250	62	31
3	16	333	•	41	•
4	12	250	•	31	•
5	10	•	•	•	•

`gain: < int >` *Available only for the ADS1X15 interface.*

Set gain of ADS-1x15.

The ADC input range (or gain) can be changed via this parameter.

Available options are:

Gain	VDD	Resolution (1 bit = x mV)		
x	(+/- V)	Uno	ADS-1015	ADS-1115
2/3	6.144	•	3	0.1875
•	5.0	4.88	•	•
1	4.096	•	2	0.125
2	2.048	•	1	0.0625
4	1.024	•	0.5	0.03125
8	0.512	•	0.25	0.015625
16	0.256	•	0.125	0.0078125

Note: A gain of two third is set with `"gain":23`.

Warning: Never exceed the VDD +0.3V ! Exceeding the upper or lower limits may damage a channel of your ADC or destroy it ! Be carefull with this setting, be conservative and start with a gain of 2/3 (`"gain":23`) for an input of +/- 6.144 V

Note: Gain is displayed on the digital display on the top left of the GUI. Gain setting is recorded in the output log file with the units.

`variables: {}` Each pin connected to the board need to be declared in this section. Each variable has a name, a pin name and units.

`pin: "< string >"` Set which pin to read measurments from.

pin single ended options are:

- SIN_A0
- SIN_A1
- SIN_A2
- SIN_A3
- SIN_A4
- SIN_A5 (available only on Arduino Uno)

pin differential connections options are (available only on ADS-1X15):

- DIF_A01
- DIF_A23

Example of configuration for logging data of an analog fluorometer, the WET Labs WSCD. The instrument output is 12 bit 0-5 Volts, as we are taking measurements in very clear water, signal should never go above 3 Volts. In order to match the resolution of the instrument, an ADS-1015 is used with a gain setting of 1x and a frequency of 1 Hz (as the instrument operates at 1 Hz). The <user_cfg.json> file look like:

```
"instruments":{
  "WSCD_859":{
    "module":"Arduino",
    "name":"ADS1015",
    "frequency":1,
    "gain":1,
    "variables":{
      "fdom":{
        "pin":"SIN_A0",
        "units":"counts"
      }
    }
  }
}
```