QAMT Workflow Breakdown

Ethan Runge

March 2023

1 Introduction

This document outlines the operation of the `Airborne_DiasPro.py` function which at time of writing is the primary processing method for QAMT data. This document is also meant to provide an outline of the necessary steps in the processing chain and a preliminary proposed workflow to replace `Airborne_DiasPro.py`

2 Airborne_DiasPro.py, the legacy function

The current implementation is intended to be run from a command line, with a `*.inp` file provided as an input. It has three options:

1. create a `*.inp` file, does not require an argument
2. open the project associated with the provided `*.inp` file, will fail if none is provided or if it does not exist
3. exit, self explanatory

2.1 The *.inp file

The `*.inp` file has the following structure (using my own local file as an example):

```
<Airborne Data Processing Parameters>
<Survey Parameters>
  <Name>
    <Ethan - Leidos Flight Test - 20200209 - Reading 145509>
  <Client>
    <Dias>
  <Geophysical Method>
    <QAMT>
  <Dipole Length>
    <0.0>
    <0.0>
<Output Parameters>
  <Output Directory>
    <C:\Users\EthanRunge\OneDrive - DIAS Geophysical\Documents\Processing>
  <Output File Root>
    <testing_20230209_>
<Input Parameters>
  <Reading>
    <20230209_145509>
  <Noise Analysis>
    <Path to Stationary Time Series>
      <>
  <Pre-processing cryostat>
    <SQUID Path>
      <C:\Field_Data\20230209\>
  <Line Segmentation>
    <Gps Solution Path>
      <1>
      <C:\Field_Data\20230209\20230209_145509_DAS.gps>
    <GPX file for end points>
      <C:\Field_Data\20230209\endlines_145509.gpx>
  <Calibration Fluxgate>
    <Gps Solution Path>
      <C:\Field_Data\20230209\20230209_145509_DAS.gps>
    <Fluxgate Path>
      <C:\Field_Data\20230209\>
  <Inversion Angle between SQUID and Fluxgate>
    <SQUID Path>
```

```
    <C:\Field_Data\20230209\>
  <GPS Solution Path>
    <C:\Field_Data\20230209\20230209_145509_DAS.gps>
  <Fluxgate Path>
    <C:\Field_Data\20230209\>
  <Fluxgate Calibration Path>
    <C:\Users\EthanRunge\OneDrive - DIAS Geophysical\Documents\Processing\products\
  calibrations\20230209_145509\>
<Kalman Filter Processing>
  <DAS Path>
    <C:\Field_Data\20230209\>
  <Gps Solution Path>
    <C:\Field_Data\20230209\20230209_145509_DAS.gps>
  <Filter Type>
    <Forward_Backward>
  <Time Segmentation File>
    <C:\Users\EthanRunge\OneDrive - DIAS Geophysical\Documents\Processing/products/
  testing_20230209__segmentation.time>
  <Fluxgate Calibration Path>
    <C:\Users\EthanRunge\OneDrive - DIAS Geophysical\Documents\Processing\products\
  calibrations\20230209_145509\testing_20230209__scaling.rot>
<SQUID Data Rotation>
  <SQUID Path>
    <0>
  <Fluxgate Path>
    <0>
  <Quaternion Path>
  <0>
  <Angle Squid Fluxgate>
    <C:\Users\EthanRunge\OneDrive - DIAS Geophysical\Documents\Processing/products/
  testing_20230209_rotation.rot>
  <Fluxgate Calibration Path>
    <0>
<Grid preparation>
  <Gps Solution Path>
    <1>
    <C:\Field_Data\20230209\20230209_145509_DAS.gps>
  <Gpx file for end points>
    <C:\Field_Data\20230209\endlines_145509.gpx>
  <Station spacing>
    <100>
<Time series extraction>
  <Gps Solution Path>
    <C:\Field_Data\20230209\20230209_145509_DAS.gps>
  <Stations GPX Path>
    <C:\Users\EthanRunge\OneDrive - DIAS Geophysical\Documents\Processing/products/
  testing_20230209__stations.gpx>
  <Remote Path>
    <C:\Field_Data\20230209\_remote>
  <HDF5 lines files>
    <1>
    <02>
    <C:\Field_Data\20230209\_squid\145509_02.h5>
  <Distances>
    <HF>
      <100>
    <LF>
      <200>
  <Filtering files>
    <HF>
      <C:\Field_Data\20230209\filtering_HF_simple.txt>
    <LF>
      <C:\Field_Data\20230209\filtering_LF_simple.txt>
  <Frequency band sampling rates>
    <HF>
      <31250>
    <LF>
      <3125>
<Tipper calculation>
  <Lines To Process>
    <1>
    <02>
  <Frequency band>
    <HF>
  <Number of cores>
    <4>
```

```
    <Processing File>
      <C:\Field_Data\20230209\processing_parameters_HF_wavelets.csv>
  <Event window visualization>
    <Event window visualization file path>
      <C:\Users\EthanRunge\OneDrive - DIAS Geophysical\Documents\Processing\products\extracted
    \20230209_145509\02\events\events_021.h5>
    <Frequency band>
      <HF>
    <nfft>
      <>
    <Window number>
      <>
```

This file has many repeated instances, "GPS path" for example, and can be confusing initially to try to discover what is necessary to run each of the steps. For example, <Grid preparation> <GPS Solution Path> needs the first option to be a number rather than a path to a `*.gps` file. After the process fails it becomes apparent that the grid preparation function has a built-in ability to take multiple `*.gps` files, but this is not initially intuitive to the user running the data.

2.2 Running the Program

Assuming that one has a valid `*.inp` file used as an input, choosing to open that project gives the following in the terminal:

```
Opening input file...
Input file loaded.
Choose processing option:
   0. Creating project folder
   1. Correcting cryostat from fluxjumps and current compensation
   2. Get bias values from stationary time series.
   3. Line Segmentation.
   4. Derive rotation matrix between SQUID and Fluxgate.
   5. Calibrating fluxgate using procrusts.
   6. Kalman Filter Processing.
   7. SQUID Data Rotation.
   8. Quality control of rotation quality.
   9. Grid preparation for tipper processing.
   10. Time series extractions.
   11. Tipper processing.
   12. Writing databases ASCII.
   13. Testing.
   14. Visualize event windows.
   15. Exit.
Option: >?
```

The loading step takes the `*.inp` fields and creates a `projectParameter` python class (found in `io_lib_kalman.py`) with each of the defined paths, called as the variable `inputParameters` The user then enters the particular option they wish to perform.

Each of the options has its own specific check to see if the proper fields exist in the generated `inputParameters` will work for the currently selected option. If the check fails, the function gives a statement indicating which path is not properly defined.

Each of the options is also intended to fill the appropriate fields after the function is run, but this doesn't necessarily work, as not all inputs for a later step can be defined by an earlier step completion. An example is the Kalman filter step 6. It fills the appropriate quaternion path entries in step 7. But the GPS paths still need to manually be set.

2.2.1 Option 0: Creating project folder

This function creates a series of subfolders in the location defined by the `*.inp` field <Output Parameters> <Output Directory> if they do not already exist in that location.

It creates these folders:

- deliverables
  - rotation_databases
  - processing_databases
- products
  - extracted

  - kalman_filter
  - rotated
  - processed
  - corrected
- input_files
- raw_data

It appears that some of these folders are redundant or not in use. For example, if "raw_data" has none of your raw data, no issues arise.

2.2.2 Option 1: Correcting cryostat from fluxjumps and current compensation

This function is responsible for dejumping the SQUID files. Relevant `*.inp` entries are under <Input Parameters> <Pre-processing cryostat>. It checks that the sub-argument under <SQUID Path> is a valid folder. This folder is where the user should be placing the `*.qamt` files.

After the dejumping completes, the output files are saved as `*.npy` (one `*.npy` for each `*.qamt`) in the folder location `[output_path]/products/corrected/[reading]/`. "reading" is another user defined parameter in the `*.inp` file under <Input Parameters> <Reading>

2.2.3 Option 2: Get bias values from stationary time series

This function returns the following strings:

```
-> Automatic determination of bias values from stationary time series.
-> Not ready yet. Be patient young Padawan.
```

The function was never implemented and I am unsure what the original intention or purpose of the function is.

2.2.4 Option 3: Line Segmentation

This function does automatic line detection of valid flight lines based on given `*.gps` and `*.gpx` files. Relevant `*.inp` entries are under <Input Parameters> <Line Segmentation>. It requires two paths to be defined, one under <GPX file for end points> which points to a `*.gpx` file and another two options under <Gps Solution Path>, an integer first and a `*.gps` file. Note that the `*.gpx` file needs to come from a different program and that the first argument in the <Gps Solution Path> section needs to be an integer or the process fails.

The output of the function is a series of beginning and end times defining the lines, given in `[output_path]/products/[file_root]_segmentation.time`

This is the first option that uses the `[file_root]` parameter defined in the `*.inp` file under <Output Parameters> <Output File Root> which allows the user to define a prefix common to all files from the current processing.

2.2.5 Option 4: Derive rotation matrix between SQUID and Fluxgate

This function requires a secondary choice from the user about which option they would like to use (some of which appear to not work?):

1. Rotation/1D Gain/Offset
2. Rotation/3D Gain/Offset
3. Distortion/3D Gain/Offset

This process appears as though it were in the debugging stages as many options are commented out and leave statements about it not being finished.

Relevant entries in the `*.inp` file are under <Input Parameters> <Inversion Angle between SQUID and Fluxgate> The necessary inputs for the function are a path that leads to a folder with the `*.npy` outputs under <SQUID Path>. Note that this path is not automatically filled, so you need to go into `*.inp` and fill the folder location manually, which should be the same location where the output of Option 1: Correcting cryostat from fluxjumps and current compensation is stored.

This function also requires a path to the folder containing raw fluxgate files under <Fluxgate Path>, which are `*.adc` files. This presents a small problem as the fluxgate files are not used in every flight, but some file is still required or else the program fails. At present `*.adc` files that are just noise would be passed to this function. Note that the option under <Fluxgate Path> is to a *folder* where the `*.adc` files are, not to the file

itself. Contrasting to that, the option <GPS Solution Path> must specifically point to a `*.gps` file that is being used for the analysis (hopefully the same `*.gps` file used in all other steps!)

This option also requires a fluxgate calibration path, which points to a location where the results of the next step are saved, indicating that Option 5: Calibrating fluxgate using procrustes must be run first. The path to the folder that contains the solution file needs to be entered into the <Fluxgate Calibration Path> field.

The result of this step is a rotation matrix stored in a file: `[output_path]/products/[file_root]rotation.rot`

It should be noted that sometimes the underscore is present between the `file_root` and the suffix, and sometimes not. Leaving it as is for now to avoid potentially breaking code that looks for specific line breaking characters.

2.2.6 Option 5: Calibrating fluxgate using procrustes

This function should be run before Option 4: Derive rotation matrix between SQUID and Fluxgate, as it calibrates the fluxgate. Like the previous option, it has sub-options:

1. Rotation/1D Gain/Offset
2. Rotation/3D Gain/Offset (currently beta)

These options seem to clearly map to the options of the previous step, so it seems like the choice made here will create a calibration file that will only work with the same option in Option 4: Derive rotation matrix between SQUID and Fluxgate.

Relevant entries in the `*.inp` file are under <Input Parameters> <Calibration Fluxgate> Required are <Gps Solution Path>, pointing specifically to a `*.gps` file and <Fluxgate Path>, the folder that contains the `*.adc` file, specifically not to the `*.adc` itself.

The function places the result of the analysis as a file at `[ouptut_path]/products/calibrations/[reading]/[file_root]_scaling.rot`

2.2.7 Option 6: Kalman Filter Processing

This function in the current iteration applies the legacy Kalman filtering code (with a hefty amount of debugging options I am currently using as a temporary solution).

Relevant entries from `*.inp` are under <Input Parameters> <Kalman Filter Processing>

Input requirement <DAS Path> defines a folder location that must contain the necessary `*.adc` and `*.uimu` files. These files must be in the form of `[reading]_DAS.adc` or `[reading]_DAS.uimu` <Gps Solution Path> must be defined once again, with a specific `*.gps` file but no integer requirement. <Filter Type> need to be defined as either "Forward", "Backward" or "Forward_Backward". <Time Segmentation File> must point specifically to the location of a `*_segmentation.time` file, generated from Option 3: Line Segmentation. <Fluxgate Calibration Path> must point to the `*_scaling.rot` function output by Option 5: Calibrating fluxgate using procrustes

The output of this function is a series of rotation matrices and other assorted outputs from the Kalman filter saved in hdf5 files under `[output_path]/products/kalman_filter/`. Within this folder are subfolders for each of the identified lines, and in each of those subfolder and the `*.h5` files defining the various Kalman filter results.

2.2.8 Option 7: SQUID Data Rotation

This function takes the SQUID data in `*.npy` form and rotates the data with the Kalman filter outputs. I am not totally familiar with how this function is intended to work. Relevant entries in `*.inp` are found under <Input Parameters> <SQUID Data Rotation>.

The <SQUID Path> entry needs to point to an existing folder but does not appear to be used. <Fluxgate Path> has been previously defined, and needs to match the previous steps. <Quaternion Path> appears to point to the folder that contains the `*.h5` files, and should have been autofilled at the end of Option 6: Kalman Filter Processing. <Angle SQUID Fluxgate> needs to point at the previously generated `*_rotation.rot` file. <Fluxgate Calibration Path> needs to point at the previously generated calibration file.

The output creates a series of `*.h5` files in `[output_path]/products/rotated/[file_root]` again in a series of subfolders divided by line number. The `*.inp` has the time series extraction fields filled out after the completion of this step.

2.2.9 Option 8: Quality control of rotation quality

I have not looked at this option at all, but it needs time series extraction files to be generated by Option 7: SQUID Data Rotation. It does not appear to create an output.

2.2.10 Option 9: Grid preparation for tipper processing

This function prepares the grid on which the tipper will processing occurs.

Relevant `*.inp` files are under <Input Parameters> <Grid Preparation>. The field <Gps Solution Path> needs two definitions, an integer and the `*.gps` file that has been used in all other steps. It also requires a `*.gpx` file that indicates the endlines under <Gpx file for end points>. Finally, <Station spacing> needs an integer number defined.

The output is another `*.gpx` file that defines the station locations, saved at `[output_path]/products/[file_root]_stations.gpx`

2.2.11 Option 10: Time series extraction

This function performs the time series extraction for the corrected SQUID data and the prepared grid. Relevant entries in `*.inp` are under <Input Parameters> <Time series extraction>.

<Gps Solution Path> is the same as prior options. <Stations GPX Path> needs to point at the output of Option 9: Grid preparation for tipper processing. <Remote Path> is a path to a series of files from the base station, saved as a series of `*.raw` files. <HDF5 lines files> has three fields, the first an integer indicating how many files to process, the second indicating which of the files you wish to process (the number of entries here should match the integer in the first argument). The third field points at the `*.h5` file corresponding to the second argument. <Distances> has a <LF> and <HF> field, each of which take an integer. <Filtering files> also has <LF> and <HF> fields, and take a `*.txt` file as an input. <Frequency band sampling rates> has <LF> and <HF> fields which each need to be defined with a number indicating the sampling rate for each of the types.

The output of the function creates an "events" folder for the processed line, looking like so: `[output_path]/products/extracted/[reading]/[HDF5 lines files, arg 2]/events/events_*.h5`

2.2.12 Option 11: Tipper processing

This is the tipper processing step. Relevant entries in `*.inp` are under <Input Parameters> <Tipper calculation>.

<Lines to process> has two parameters, a number of lines to process first, and the second indicates the lines you wish to process. <Frequency band> is either "HF" or "LF" depending on user preference. <Number of cores> is a parameter for parallel processing purposes. <Processing File> is a file (I do not know where it comes from or how it is generated) of type `*.csv` needed to tell the function how to process the information.

The function output is a `*.h5` file placed at `[output_path]/products/processed/[reading]` with a file name defined by `[name]_[client]_[reading].h5` where `[name]` and `[client]` are defined in the `*.inp` file under the <Survey Parameters> subsections <Name> and <Client> respectively.

2.2.13 Option 12: Writing databases ASCII

I haven't done anything with this function, but it is meant to create a processing database using the outputs of Option 11: Tipper processing at `[output_path]/deliverables/databases/processing_databases/`

2.2.14 Option 13: Testing

This function is null/redundant. The current version of the code has a function that appears to be testing the determination of rotation angle between the SQUID and fluxgate and is incomplete.

2.2.15 Option 14: Visualize event window

This is a "Function to visualize lightning event windows found in a given station and used in the robust regression." Relevant entries in `*.inp` are under <Input Parameters> <Event window visualization>.

<Event window visualization file path> points at one of the events files generated in Option 10: Time series extraction. <Frequency band> takes as argument either "LF" or "HF" based on user preference. <nfft> takes an integer argument defining a number of points. Each window (see next argument) has a valid nfft value the user can choose. This can be left blank to trigger the function to prompt the user with valid options. <Window number> is also an integer, indicating which of the windows the user wishes to view. This can be left blank to trigger a prompt showing what available options are.

This function generates no output files.

3 New Processing Chain

David came up with this flowchart for the processing with the new `*.h5` file structure we are planning to use.

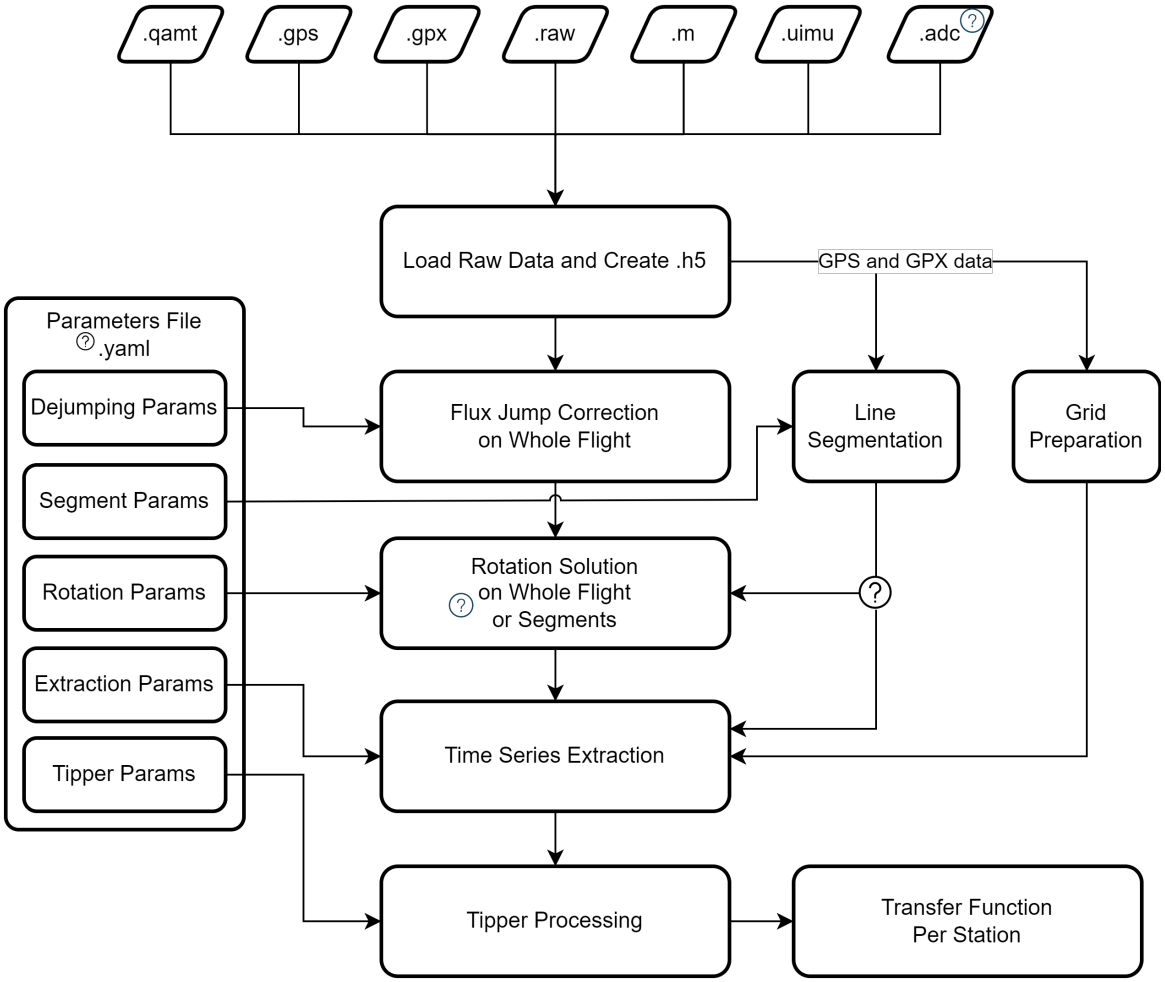

Figure 1: Flowchart for the new processing, making use of `*.h5` files. The parameter file could be `*.yaml` or another type, but values used within it can be saved as meta data to the `*.h5` file as the data is passed through that step.

There are still some questions about where the line segmentation should occur, whether or not future flights will make use of fluxgate measurements and whether they are necessary, and the methodology by which we include parameters. Ideally, anything that can be changed by the user would be included in the same location, with comments or documentation indicating what those fields do, and the parameters used in any processing step would be saved to the meta data at that step.