Python QAQC workflow

Sarah Frey

5 November 2018
Edited 20 September 2019
Edited 6 January 2019

This workflow is based largely on the readme file Matt Gregory created for running this Python QAQC program. Also, the Terminal commands differ slightly depending on whether you are using a PC (red) or a Mac (blue).

The Python code was created for the purpose of data quality assurance and quality control of HOBO temperature sensor data collected at the Andrews Forest. It reads in the raw data and compiles and formats it, flags the data based on user-specified parameters, cleans out the flagged data selected for removal, and fills in missing data with regressed data from other sites with similar temperature patterns (this is based on r^2 values; for the tree loggers, the sites available for data filling are restricted to the same tree). In the notes below, the "python code manager" would be a person yet to be designated, who is connected with the Andrews Forest field technicians and data managers; if in doubt, contact Mark.Schulze@oregonstate.edu.

Please see the document 'Python_flag_definitions.docx' for the list of flags, their definitions and parameters (which can be adjusted). All observations that are flagged are removed in the "cleaned" files. In the "filled" files, the missing observations are filled based on a regression based on data from other sites with similar temperature patterns.

A flag parameter file known as a .yaml file needs to be present in your working directory. For the 1.5m HOBOs, it is config_ground.yaml and for the tree HOBOs, it is config_tree.yaml. Make sure that in the config_ground/tree.yaml file, the paths point to your versions of the raw data files. This should create the flagged, cleaned, reference and filled files for each raw file in the 'new_bird/all_sites_raw_final' (for 1.5m HOBOs) or 'new_tree/all_sites_raw_final' (for tree HOBOs) directory.

Your file structure will look something like this:

data_clean = working directory where all of your files are located, this folder will contain the config_ground.yaml or config_tree.yaml file and the folder new_bird (or new_tree).

new_bird or new_tree = this folder should contain either the 'raw' folder or the 'all_sites_raw_final' depending on what version of raw you are starting with. It should also contain the config.yaml file. This is also where the folders 'cleaned', 'filled', 'flagged' and 'reference_sites' are created if they are not already present.

Once you have your folders and data and flag file set up, you are ready to install and run the Python QAQC program. Follow these instructions:

1. Install python 3.7 on your computer

2. Open Terminal, then change to the working directory (assuming working directory is C:\hja\data_clean)

> cd C:\hja\data_clean

$ cd /HJ_Andrews/data_clean

The exact working directory will depend on your computer and where the folders and files are located.

3. Check the version of python that you have installed is the correct one:

> C:\Python37\python.exe (not sure if this is correct anymore)

$ python3 --version

If this works then move forward, otherwise contact python code manager.

4. Set up new virtual (Python) environment for this code

> C:\Python37\Scripts\mkvirtualenv hja

$ python3 -m venv hja

5. Activate the virtual environment

$ source hja/bin/activate

This will set up a new virtual environment called 'hja' and your command prompt will now look like:

(hja) >

(hja) $

Whenever you work on this project, you should use this virtual environment rather than the default Python installation. Whenever starting up, use this command to get into the 'hja' virtualenv.

> workon hja

"workon" does not work with Mac, use this instead

$ source hja/bin/activate

This will make the virtual environment active and give you an (hja) at the beginning of your prompt. You can test that you are actually running Python 3 by typing 'python' and it should say something like Python 3.7.x right above the '>>>'

6. Install needed packages. The hja-hobo-clean.whl file actually stores all the packages it needs (or it should), so it will install these dependencies (e.g. numpy, pandas, etc.) before it installs hja-hobo-clean itself.

On a PC:

7. Install wheel file for hja-hobo-clean

Wheel file = hja_hobo_clean-0.1.0-py2-none-any.whl

(hja) > pip install -U /path/to/hja_hobo_clean-0.1.0-py3-none-any.whl

(hja) $ pip install -U /path/to/hja_hobo_clean-0.1.0-py3-none-any.whl

(*Where '/path/to/' would be replaced with the path to the location of the file on your computer. If you are already in the directory where the wheel file is located, you can leave out the /path/to/ part.*)

You can check the version of the packages on your machine with:

(hja) $ pip list --format=columns
(*not sure what the PC version is*)

It will show you something like this:

```
Package        Version
-------------- -------
Click          7.0
dotmap         1.3.8
Flask          1.1.1
hja-hobo-clean 0.1.0
```

itsdangerous 1.1.0
Jinja2 2.10.3
MarkupSafe 1.1.1
numpy 1.17.4
pandas 0.25.3
pip 19.3.1
python-dateutil 2.8.1
pytz 2019.3
PyYAML 5.1.2
scipy 1.3.2
setuptools 41.2.0
six 1.13.0
Werkzeug 0.16.0

After completing step 7, go to step 8 if this is the first time you have run this code. Go to step 9 if you already have installed everything from steps 1 through 7.

8. Execute the hja-hobo-clean QAQC

Here we provide only the Mac code for this step.

When the QAQC is executed the folders 'cleaned', 'filled', 'flagged' and 'reference_sites' are created if they are not already present.

FOR 1.5m DATASET
$ hja-hobo-clean config_ground.yaml

FOR tree DATASET
$ hja-hobo-clean config_tree.yaml

These files are in the folder 'all_sites_raw_final' which are located on box.

FOR STARTING FROM RAW FILES (TAKES A LOT LONGER)
$ hja-hobo-clean config_tree.yaml –raw
$ hja-hobo-clean config_ground.yaml –raw

9. If you have already done steps 1-7 then you would only need to run the following commands to run the QAQC:

$ cd /path/to/data_clean

(*Where '/path/to/' would be replaced with the path to the location of the folder that contains the .yaml file and the folder 'new_bird/all_sites_raw_final' on your computer*)

Activate the virtual environment

$ source hja/bin/activate

FOR 1.5m DATASET USING COMPILED RAW FILES
$ hja-hobo-clean config_ground.yaml

FOR TREE DATASET USING COMPILED RAW FILES
$ hja-hobo-clean config_tree.yaml

FOR STARTING FROM SEPARATE RAW FILES (TAKES A LOT LONGER)
$ hja-hobo-clean config_ground.yaml –raw
$ hja-hobo-clean config_tree.yaml –raw

A NOTE ON STARTING FROM RAW FILES: When you choose to start with the raw files, you have subfolders of all the files that exist for each logger/site that are placed into the 'raw' folder. Their names are all formatted the same: 'BIRD_XXX015' with XXX being the site number written out as 3 digits and 015 referring to the height of the logger. For the tree HOBO files, the names are as follows: TREE_XXXYYY with XXX referring to the site number and YYY referring to the height of the logger (065, 115, 165, etc.). Each folder also needs to contain a metadata.yaml file that contains the following information: file_name_match: BIRD_XXX_*.csv or TREE_XXXYYY_*.csv . These files already exist in each site folder so it is best to add additional files in the future to those folders. Also, this only needs to be done once for each time additional data are added, because it creates the compiled and formatted files that go into 'all_sites_raw_final' folder, which can then be used for any additional QAQC executions on those same data. Running it from the 'really' raw files takes a lot longer as well.

* To update hja-hobo-clean file:
$ pip install -U hja_hobo_clean
* To update the wheel file
$ pip install -U hja_hobo_clean-0.1.0-py3-none-any.whl

DATA AND FLAG VISUALIZATION TOOL

Run data visualization (from bitbucket)

Clone the git repository to your system

Open a command shell and navigate to a parent directory where you want to put the application

Type: git clone https://<username>@bitbucket.org/hjandrews/hja-hobo-visualize.git ha-hobo-visualize using your username without the angle brackets.

This creates a directory called hja-hobo-visualize that has all the files you need to run the application.

Populate the 'static/data' directory with flagged files

Copy all flagged files produced from hja-hobo-clean into the 'static/data' directory. The application looks for CSV files in this directory to populate the dropdown list in the application.

Make sure that flask is installed
Ideally, the user has set up a Python virtual environment for isolating this from their main Python install. For instructions, see documentation for the virtualenv package (and the virtualenvwrapper-win package if on Windows).

Once the virtual environment is set up, run the following command:
> pip install -U flask

Start the flask application

Get into the hja-hobo-visualize folder
> cd C:\hja\data_clean\hja-hobo-visualize

Open a command shell
> python app.py

On a Mac:
$ pip install -U flask
$ cd /HJ_Andrews/data_clean/hja-hobo-visualize
$ python app.py

Open a browser and navigate to http://localhost:5000/

Stop the flask application: Once finished with the visualization, close the browser tab and hit 'Ctrl-C'