sensortoolkit beta-testing instructions

Samuel Frederick, ORAU Contractor for U.S. EPA ORD

Office: 919-541-4086 | Email: frederick.samuel@epa.gov

Last updated 12/10/2021, sensortoolkit v0.6.3 beta 2

Thank you for participating in the internal beta-testing of sensortoolkit! Your feedback in running through the library features will be valuable in improving sensortoolkit for public release.

Technical Reviewers:

Once you have completed reading through this documentation, please review the `component_testing_and_validation.docx` for details regarding the testing of sensortoolkit methods and validation of computed statistical quantities.

What is sensortoolkit?

sensortoolkit is a Python library for evaluating air sensor data. The library is intended for use with sensors collocated at ambient air monitoring sites alongside FRM/FEM monitors for comparison and analysis of sensor data against reference-grade data.

How can sensortoolkit help users evaluate sensor data?

- Import sensor data via a standardized ingestion process and interactive setup module.
- Average to 1-hour and/or 24-hour averaging intervals.
- Import FRM/FEM reference data from a variety of sources, including ingestion modules for importing data from AirNowTech, and modules for querying either the AQS or AirNow API services.
- Submit queries for single or multiple parameters, parse datasets into a consistent reference data format and save unmodified and processed datasets to a data directory.
- Conduct analysis with the `SensorEvaluation` module
  - Compute U.S. EPA's recommended performance metrics for evaluating $PM_{2.5}$ and O_3 sensors.
  - Visualize sensor performance with various figures and save to file location.
    - Sensor vs. FRM/FEM scatter plots.
    - Timeseries indicating both sensor and FRM/FEM concentrations.
    - Performance metric results and comparison against target values/ranges.
    - Save performance evaluation results, statistics, and supplemental information detailing the deployment conditions to a deployment JSON file.
- Create testing reports using U.S. EPA's base-testing report template (PowerPoint file) with the `PerformanceReport` module.
- Additional modules are included for calculating quantities (AQI, $PM_{2.5}$ NowCast, application of sensor correction equations, the U.S. Wide correction equation for PurpleAir sensors via Barkjohn et al. 2021 [1], etc.) and conducting additional analysis (quality control modules for identifying outliers, invalidation of data points, A&B channel cleaning for PurpleAir data via Barkjohn et al. 2021 [1], etc.).
- Access to modules utilized by the `SensorEvaluation` and `PerformanceReport` for greater customization in conducting analysis.

Footnotes

[1]: Barkjohn, K. K., Gantt, B., and Clements, A. L.: Development and application of a United States-wide correction for PM2.5 data collected with the PurpleAir sensor, Atmos. Meas. Tech., 14, 4617–4637, https://doi.org/10.5194/amt-14-4617-2021, 2021.

Who is the intended audience for this library?

sensortoolkit is most suitable for individuals who have some prior coding experience in python. The library is equipped with an API (application programming interface) that allows for ease of navigation and customization, making sensortoolkit accessible to individuals with a wide range of skillsets (e.g., individuals interested in monitoring their own sensor data, students and academic researchers, and industry professionals).

Part 1: Installing and Updating sensortoolkit

Prerequisites

- Prior to the steps listed below, individuals interested in beta-testing will be granted access to the BitBucket repository where the library code is maintained. BitBucket is used by EPA for hosting private, internal projects and for maintaining version control for projects intended for public release that have not undergone clearance yet.

> **Please make sure you are able to view the repository at the following link: https://bitbucket.epa.gov/users/sfrede01/repos/sensortoolkit/browse**

If you are not able to view the repository, please reach out to me (frederick.samuel@epa.gov) and I will grant you access to the repository.

- Beta-testers will need the following software installed to run sensortoolkit:
  - git (on the list of preapproved software, email EISD directly to have git installed on your system). git is a version control system that streamlines the process of downloading (cloning) online repositories, updating local copies, and suggesting changes to the online repository based on modifications to your local repository.
  - A python (>=v3.7) distribution (I highly recommend Anaconda since it comes with the Spyder integrated development environment (IDE), and is what I have used to develop sensortoolkit. Users will need to submit a software request to have Anaconda installed).
  - A command line interface (CLI) for submitting various git commands and installing/updating the sensortoolkit library. I recommend using Anaconda Prompt, which comes alongside an Anaconda installation. I recommend this prompt over the Windows Command Prompt, mainly because users may occasionally need to install or update python packages, and for Anaconda installations, this is handled by the conda package manager. Conda is built in to the Anaconda Prompt, so using this CLI is preferred.

Cloning the repository

1. Open up Anaconda Prompt and navigate to a directory where you would like to install the repository. This directory will be the parent directory of the repository. For example, if you navigate to the `/Documents` directory via the CLI prompt:

```
cd C:/Users/.../Profile/Documents
```

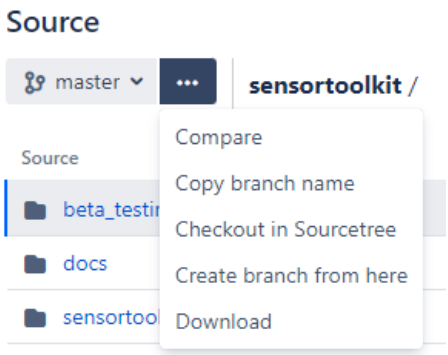
you should expect the repository to be located at the following path: `C:/Users/.../Profile/Documents/sensortoolkit`

2. Clone the repository with git using the following CLI prompt:

```
git clone https://bitbucket.epa.gov/scm/~sfrede01/sensortoolkit.git
```

This will download the most recent version of the repository on BitBucket to a new folder `/sensortoolkit` within the user's current working directory.

> Note: If you're having trouble using git to download the repository due to authentication issues or other roadblocks, you can manually download the repository contents as a zip folder from the BitBucket repository webpage. Under the "Source" header at the top left of the webpage, click on the "…" button and select "Download" from the dropdown menu:



Installing the library with pip

1. Once you've cloned or manually downloaded the repository, navigate to the folder location for the local repository. If you're continuing from the instructions above for cloning the repository, you should be able to use the following CLI prompt to change directories:

```
cd sensortoolkit
```

2. Now that you're in the library folder, the sensortoolkit needs to be installed to a target directory where python looks for packages whenever the user tells python to import a package name. By default, this is the `/site-packages` directory, and should be located at a path that looks something like `C:\Users\your_LAN_ID\Anaconda3\Lib\site-packages` (if you have Anaconda installed). The location of this package may be a little different depending on how your python installation was configured, although this shouldn't matter too much.

   Type the following CLI prompt to install sensortoolkit (don't forget the period!):

```
pip install .
```

> Note: By default, pip attempts to install packages for all users accounts. Depending on the permissions granted to the user's account, pip may not be able to install all packages. If this occurs, users should add `--user` to the above command (e.g., `pip install . --user`) to indicate that the installation should take place only on the user account.

The installation process checks for a number of packages sensortoolkit needs to run (dependencies).
If you have Anaconda installed, you'll notice that the installation process may indicate that a lot of the required libraries are already installed as those packages come with the base installation of Anaconda. Pip may need to install `python-pptx`, which is used by sensortoolkit to create testing reports as .pptx files.

Importing the sensortoolkit Library

1. Once you've downloaded and installed the library, you may want to ensure that sensortoolkit has been installed correctly. Still within the Anaconda Prompt, type the following CLI prompt:

```
python
```

This will launch python within the Anaconda Prompt in interactive mode. Next, type the following python commands (*Note: If you're copying and pasting commands in, you will need to copy one line at a time*):

```
import sensortoolkit
sensortoolkit.__version__
```

This should print the version of the library that you downloaded from BitBucket and indicates that the library has been properly installed.

> Tip: You can exit the interactive python console by typing `exit()`.

2. Up till now, we've been using the Anaconda Prompt (or an equivalent CLI) for download, installing, and importing the sensortoolkit library. For making the best use of the library and modules, I highly recommend beta-testers use an IDE such as Spyder that comes installed with an Anaconda distribution. Spyder (much like R-Studio) provides users with a platform for scripting and executing code as well as useful utilities such as the variable explorer that can be a valuable tool when working with sensor datasets.

   In order to open Spyder, you can either type into the Windows Search Menu 'spyder' (you should see an icon come up that says 'Spyder (Anaconda3)'), or if you still have the Anaconda Prompt open, you can enter the prompt `spyder`.

Updating sensortoolkit for beta-testers

1. Open the Anaconda Prompt and navigate to the location where you downloaded the repository:

```
cd path/to/sensortoolkit
```

2. Check for updates via the following git command. If your version of the library is outdated, relevant updates will be downloaded, but won't be incorporated into your local library yet.

```
git fetch
```

The next command incorporates the changes retrieved via the `fetch` command:

```
git merge
```

> Note: If you know your local library is out of date and agree to the changes to update the library, these two git commands can be combined into one `git pull` command.

3. Next, you will need to update the installation of sensortoolkit in your target directory. This is done via the following CLI prompt:

```
pip install --upgrade .
```

> Note: By default, pip attempts to install packages for all users accounts. Depending on the permissions granted to the user's account, pip may not be able to install all packages. If this occurs, users should add `--user` to the above command (e.g., `pip install --upgrade . --user`) to indicate that the installation should take place only on the user account.

Resources

Below is the file structure for the sensortoolkit library (this assumes the library has been downloaded in the Documents folder):

```
Documents
├──sensortoolkit                  <- Your cloned copy of the library
|   ├──docs                       <- HTML documentation
|   ├──beta_testing               <- Items for beta testers
|   |   |──data                   <- Example sensor dataset for testing
|   |   └──example_scripts        <- Example scripts for setting up an analysis, conducting analysis, and creating
|   └──sensortoolkit              <- Library source code
```

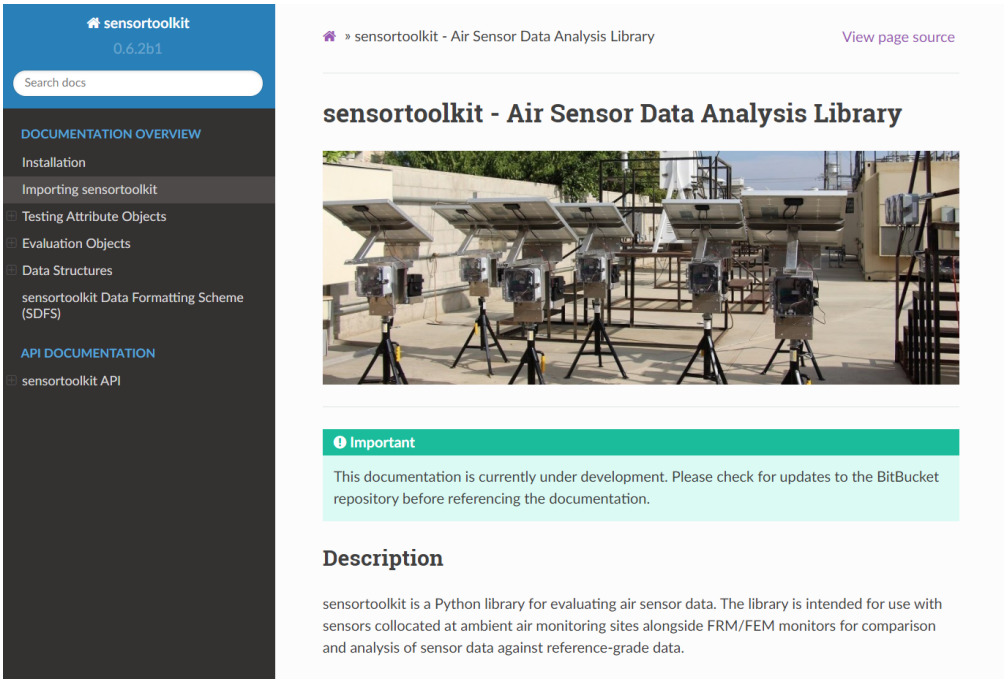
Documentation

The `/sensortoolkit/docs` folder contains the HTML documentation for the library. This documentation is intended for end users once the library is released to the public, so some of the installation discussion may differ from what is covered in this beta-testing guide. However, the HTML documentation is intended to provide a comprehensive overview of the sensortoolkit library and its use, so I would recommend beta testers review the documentation.

- To open the included HTML documentation, open a command line interface and ensure that the current working directory is the folder location where the repository was downloaded. Next, type the following command and hit enter:

```
python opendocs.py
```

The documentation should open in your default browser, and you should see a landing page that looks something like this:



Beta-testing: Example Datasets

The folder `/sensortoolkit/beta_testing` contains a number of example datasets that we will use later on during component testing.

The example sensor datasets included in `/sensortoolkit/beta_testing/data/sensor` are for a device given the name *Example Make Model*. Three files in comma-separated value (csv) format are included, and each file corresponds to a separate sensor unit.

Reference data for the evaluation are included in the `/sensortoolkit/beta_testing/data/reference` directory. The folder includes data downloaded from AirNow-Tech for collocated instrument measurements made at the sensor testing location, the Ambient Air

Innovation Research Site (AIRS) at EPA's RTP campus. The data file contains particulate matter (PM) measurements, gaseous pollutant data, and meteorological parameter data as well.

Part 2: Component Testing

1. Within your `/Documents` directory, create a new folder named `sensortoolkit_testing`.

2. Open Spyder and create a new file (*File → New File*).

3. Save this file as `component_testing.py` (*File → Save As*) and save the file within the `Documents/sensortoolkit_testing` folder. This is the script you will use to test out the features of the sensortoolkit library.

   The directory structure in your Documents folder should now look something like this:

```
Documents
├──sensortoolkit                    <- Your cloned copy of the library
|   ├──docs
|   ├──beta_testing
|   |   |──data
|   |   └──example_scripts
|   └──sensortoolkit
├──sensortoolkit_testing            <- Your testing directory
|   └──component_testing.py
```

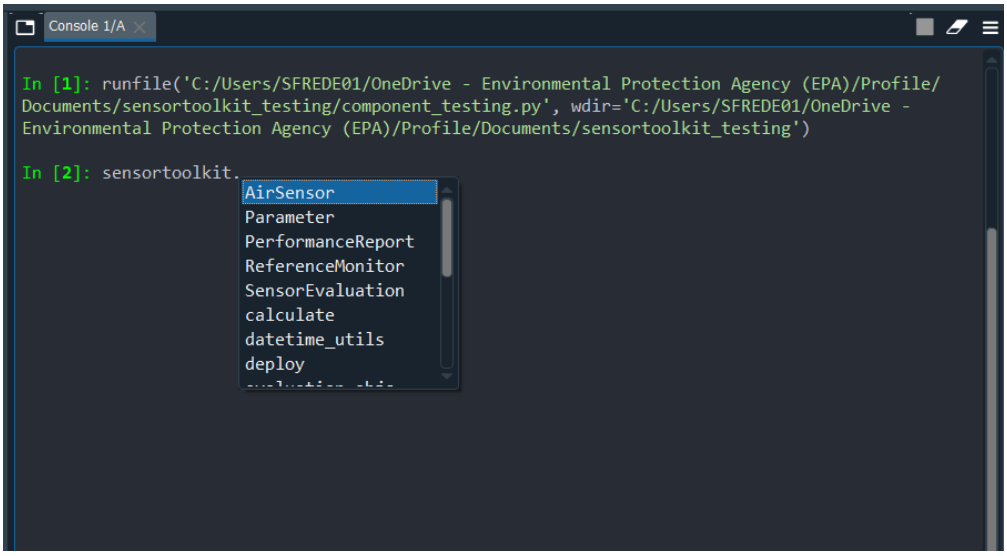
4. Within `component_testing.py`, import the library by typing the following command:

```
import sensortoolkit
```

5. Add a variable indicating the full path to your project directory. We will refer to this as the `work_path`, which will hold sensor and reference data, figures, statistics, and report files created by the sensortoolkit library. Below, I've shortened the path somewhat to fit the document easier, but be sure that the `work_path` is an absolute path and not relative.

```
work_path = (r'C:\Users\...\Profile\Documents\sensortoolkit_testing')
```

5. Run the file by pressing the green arrow in the upper left corner (or press F5). In the console, you should see a runfile statement that indicates the `component_testing.py` is being executed.
6. Once the file has run, type into the console `sensortoolkit` followed by a period "`.`". Press Tab and you will be able to view the various components in the sensortoolkit library (see below). This dropdown includes objects, sub-packages, and methods for working with sensor data. We'll focus on the objects first (items in upper-case within the drop down menu, which coincidentally are also listed at the top).



Using the provided documentation

If you still have a command line interface open such as Anaconda Prompt, refer to the above section on Resources/Documentation to open the HTML documentation included alongside the sensortoolkit library.

Once the documentation has opened in a web browser, use the side menu to navigate to the 'Testing Attribute Objects' section. Reading through this page, you should come to a snippet of code with the following statements:

```
# This python script is located at "../my_evaluation/evaluation.py"
import sensortoolkit

# Point to the full directory path for this scripts' parent directory
# This is the project folder path
work_path = ('C:/Users/.../Documents/my_evaluation')
```

For illustrative purposes, the HTML documentation assumes that users will be working in a script named 'evaluation.py' and located within a project directory named `/my_evaluation` (as opposed to `/sensortoolkit_testing` that we will use for beta-testing).

This is a sematic difference and does not impact how the sensortoolkit library is used (users are free to assign whichever name suits best for scripts using sensortoolkit and are free to access the library via scripts located anywhere on the user's system). *However, it may help useful to testers to note that, for the purposes of beta-testing, any reference in the documentation to the* `/my_evaluation` *directory will be replaced by* `/sensortoolkit_testing`*, and references to* `evaluation.py` *should be replaced by* `component_testing.py`*.*

Important - Using the beta-testing datasets:

Before diving into component testing with the documentation, I'd like to make a few notes about the beta-testing datasets and how to use these with the interactive setup module included alongside the sensortoolkit library.

The first few sections of the HTML documentation walk users through setting up their work environment for evaluating air sensor data. This includes loading in sensor and reference datasets. As mentioned in the section labeled "Beta-testing: Example Datasets", testers should use the datasets provided in `/sensortoolkit/beta_testing/data/`.

The process of importing sensor and reference data involve users interactively providing details about the sensor or reference datasets to a setup module. First, users are prompted to indicate the file data type. **For the datasets included alongside the beta-testing distribution, both sensor and reference data are provided as** `.csv` **files.**

Also during the interactive setup process, users are prompted to select data files, and a file explorer window will open for choosing files. **For the datasets included alongside the beta-testing distribution, please use the file explorer to navigate to the `/sensortoolkit/beta_testing/data/` directory and select files from within either the `/sensor` or `/reference` subdirectory**.

> Note: The reference dataset included alongside the beta-testing distribution was downloaded from AirNow-Tech. When testers are walking through the ReferenceMonitor Setup process, you will be asked to specify the reference data service or source. Please insert "airnowtech" to properly import the dataset located at `/sensortoolkit/beta_testing/data/reference`

Objectives for working through the HTML documentation

From this point, please walk through the documentation (preferably working in order down the list of items displayed in the menu of the left-hand side of the screen, beginning with 'The Air Sensor Object' in the 'Testing Attribute Objects' section). Beta testers should step through each section, using the `component_testing.py` module to run code as you go along.

Testers should focus on working through the 'Testing Attribute Objects' and 'Evaluation Objects' sections, as these sections contain details on how to use the library to set up and conduct an evaluation, and concludes in creating a testing report.

Additional documentation is provided for describing the data structures and data formatting within sensortoolkit as well detailed documentation about modules and functions within the library source code under 'API Documentation'. Please feel free to walk through these sections and provide feedback as you see fit.

Please ensure that you are able to walk through the following components:

1. Create an `sensortoolkit.AirSensor` instance (`sensor_object`)
2. Run the Setup module for specifying the ingestion configuration for importing sensor data in the `/sensortoolkit/beta_testing/data/sensor` directory.
3. Load sensor data to the `sensor_object` .
4. Create a `sensortoolkit.ReferenceMonitor` instance (`reference_object`).
5. Run the Setup module for specifying the ingestion configuration for importing AirNow-Tech reference data in the `/sensortoolkit/beta_testing/data/reference` directory.
6. Load reference data to the `reference_object` .
7. Create a `sensortoolkit.Parameter` instance (`pollutant`) for either fine particulate matter `PM25` or ozone `O3` .
8. Create a `sensortoolkit.PerformanceReport` instance and create a report. Ensure that the report was constructed properly. Successful execution of the `sensortoolkit.PerformanceReport` module will indicate broad integration testing (transfer of information between sensortoolkit methods and modules) as a result of the high level of subsystem complexity required to run the module.

Part 3: Charge Questions

Following component testing, please respond to the following questions regarding the sensortoolkit library.

1. Did you run into any issues during component testing? If so, please record all errors at time of inspection, including a description of what actions
   prompted the error. Testers are encouraged to provide the stack traceback printed to the console when the error occurred.
2. Did you find the workflow of using the sensortoolkit library to be easy to navigate (i.e., setting up testing attribute objects, importing datasets, creating performance reports, and accessing data structures, modules, and methods within the sensortoolkit library). If not, do you have suggestions for how to improve the structure or organization of the library?
3. Do you feel that there are any areas or features not currently addressed by the sensortoolkit library that may add value to an open-source public release?
4. In using the provided HTML documentation, did you find that it was easy to follow and provided an adequate overview of how to use the library? If not, please detail areas that you believe may need to be addressed more thoroughly in the documentation.

5. The planned delivery approach for this library is for it to be released as a freely downloadable, open source software package via GitHub. The code would be posted in a public repository managed by U.S. EPA. Please feel free to comment on whether this approach may be the most appropriate avenue for delivery.
6. Any additional comments or suggestions you would like to share?