Using PurpleAir data

PurpleAir sensors employ a dual laser counter to provide some level of data integrity. This is intended to provide a way of determining sensor health and fault detection. Some examples of what can go wrong with a laser counter are a fan failure, insects or other debris inside the device or just a layer of dust from long term exposure.

If both laser counters (channels) are in agreement, the data can be seen as excellent quality. If there are different readings from the two channels, there may be a fault with one or both.

In the case of a fault, the channel may be marked as flagged or downgraded (suspect or known faulty).

PurpleAir provides ways to get direct access to the data and there are a few different ways to do this.

The simplest way to download the data is using the download page available at https://www.purpleair.com/sensorlist. This page provides an easy to use interface to download data based on a date range. **You access this page by zooming into the map, then using the download button in the bottom right of the screen. Alternatively a download link is available per sensor in the "Get this widget" section after clicking a map icon.**

- Select the sensor/s in the list you want to download.
- At the top of the page, enter the desired date range, then click Download Selected.

JSON data available from PurpleAir

PurpleAir provides access to our real time data in a JSON format. This format allows developers to access current data for all sensors or a subset of sensors.

There are three ways to access the JSON data from the legacy API:
All sensors: https://www.purpleair.com/json
One entry: https://www.purpleair.com/json?show=<ID> where ID is the "ID" of the sensor you want (in the case of dual laser where ParentID is "null").
Multiple entries: https://www.purpleair.com/json?show=<ID1>|<ID2>|<ID…>

The "experimental" JSON api:
All sensors: https://www.purpleair.com/data.json
One entry: https://www.purpleair.com/data.json?show=<ID> where ID is the "ID" of the sensor you want (in the case of dual laser where ParentID is "null").
Multiple entries: https://www.purpleair.com/data.json?show=<ID1>|<ID2>|<ID…>

Accessing private sensors via the API:
All the above methods have the ability to access private sensors. You add a “key” switch consisting of the THINGSPEAK_PRIMARY_ID_READ_KEY for the primary A channel of a sensor, for example:
https://www.purpleair.com/data.json?show=<ID>&key=<KEYVALUE>
Multiple keys can be separated by “|” and if you omit the “show” parameter, you get all sensors including the private ones with the provided key(s).

NOTE: Please limit multiple threaded applications to a few threads at any time to reduce load on purpleair.com or thingspeak.com servers. If you do not limit the amount of threads, you may be blocked or rate limited.

PurpleAir JSON fields description and example values:

The following is a list of the fields and a description of their values contained in the JSON data:

```
"ID":1234, // PurpleAir sensor ID
"ParentID":null, // The PurpleAir sensor ID of the "parent" entry in the case of Channel B
"THINGSPEAK_PRIMARY_ID":"1234", // The Thingspeak channel ID for primary data of this sensor
"THINGSPEAK_PRIMARY_ID_READ_KEY":"XXXX", // The Thingspeak read key for primary data of this sensor
"Label":"name", // The "name" that appears on the map for this sensor
"Lat":null, // Latitude position info
"Lon":null, // Longitude position info
"PM2_5Value":"1.07", // Current PM2.5 value
"State":null,  // Unused variable
"Type":"TYPE",  // Sensor type (PMS5003, PMS1003, BME280 etc)
"Hidden":"true", // Hide from public view on map: true/false
"Flag":null, // Data flagged for unusually high readings
"DEVICE_BRIGHTNESS":"1", // LED brightness (if hardware is present)
"isOwner":1, // Currently logged in user is the sensor owner
"A_H":null, // true if the sensor output has been downgraded or marked for attention due to suspected hardware issues
"temp_f":"xx",  // Current temperature in F
"humidity":"xx", // Current humidity in %
"pressure":"xx", // Current pressure in Millibars
"AGE":29831, // Sensor data age (when data was last received) in minutes
"THINGSPEAK_SECONDARY_ID":"1234", // The Thingspeak channel ID for secondary data of this sensor
"THINGSPEAK_SECONDARY_ID_READ_KEY":"XXXX", // The Thingspeak read key for secondary data of this sensor
"LastSeen":1490309930, // Last seen data time stamp in UTC
"Version":"2.47c", // Current version of sensor firmware
"LastUpdateCheck":1490308331, // Last update checked at time stamp in UTC
"Uptime":"5210", // Sensor uptime in seconds
"RSSI":"-68", // Sensor's WiFi signal strength in dBm

"Stats": // Statistics for PM2.5
"{
\"v\":1.07, // Real time or current PM2.5 Value
\"v1\":1.3988595758168765, // Short term (10 minute average)
\"v2\":10.938131480857114, // 30 minute average
\"v3\":15.028685608345926, // 1 hour average
\"v4\":6.290537580116773, // 6 hour average
\"v5\":1.8393146177050788, // 24 hour average
\"v6\":0.27522764912064507, // One week average
\"pm\":1.07, // Real time or current PM2.5 Value
```

\"lastModified\":1490309930933, // Last modified time stamp for calculated average statistics
\"timeSinceModified\":69290 // Time between last two readings in milliseconds
}"
}

Accessing data directly from Thingspeak:

Another way to get the data is to use ThingSpeak.com and to do this, you will need the API Key and channelID. These two pieces of data are available from PurpleAir's JSON.

NOTE: PurpleAir implements a rate limiting mechanism to prevent users from overloading the API servers. If you request data from the API in quick succession, you will get an error message.

More info on using ThingSpeak's API is here:
https://www.mathworks.com/help/thingspeak/rest-api.html

ThingSpeak Data Field descriptions:

Channel A and B, primary and secondary ThingSpeak channels together provide 32 fields for each sensor.
There are six ug/m3 values and six particle counts for each channel (laser) as well as temperature, humidity, WiFi signal (RSSI), sensor uptime, free memory and analog input amongst others.

NOTE: Due to a bug in the firmware, data in thingspeak was incorrectly labeled, swapping ATM and CF=1 labels. In the latest version of the download tool, the labels have been corrected. The red highlights the old (incorrect) labels. In a future firmware, the JSON and interface labels will also be fixed.

Channel A

PrimaryData
field1: PM1.0 (CF=1) ug/m3 (was labeled CF=ATM)
field2: PM2.5 (CF=1) ug/m3 (was labeled CF=ATM)
field3: PM10.0 (CF=1) ug/m3 (was labeled CF=ATM)
field4: Uptime (Minutes)
field5: RSSI (WiFi Signal Strength)
field6: Temperature (F)
field7: Humidity (%)
field8: PM2.5 (CF=ATM) ug/m3 This is the field used on the map for PM2.5 (was labeled CF=1)

SecondaryData
field1: 0.3um particles/deciliter
field2: 0.5um particles/deciliter
field3: 1.0um particles/deciliter
field4: 2.5um particles/deciliter
field5: 5.0um particles/deciliter
field6: 10.0um particles/deciliter
field7: PM1.0 (CF=ATM) ug/m3 This is the field to use for PM1.0 (was labeled CF=1)
field8: PM10 (CF=ATM) ug/m3 This is the field to use for PM10 (was labeled CF=1)

Channel B

PrimaryData
field1: PM1.0 (CF=1) ug/m3 (was labeled CF=ATM)
field2: PM2.5 (CF=1) ug/m3 (was labeled CF=ATM)
field3: PM10.0 (CF=1) ug/m3 (was labeled CF=ATM)
field4: Free HEAP memory
field5: ADC0 (analog input) voltage
field6: FIRMWARE 2.5 and up: Atmospheric Pressure
field7: FIRMWARE 4.10 and up: Bosch BSEC IAQ when BME680 gas sensor is present
field8: PM2.5 (CF=ATM) ug/m3 This is the field used on the map for PM2.5 (was labeled CF=1)

SecondaryData
field1: 0.3um particles/deciliter
field2: 0.5um particles/deciliter
field3: 1.0um particles/deciliter
field4: 2.5um particles/deciliter
field5: 5.0um particles/deciliter
field6: 10.0um particles/deciliter
field7: PM1.0 (CF=ATM) ug/m3 This is the field to use for PM1.0 (was labeled CF=1)
field8: PM10 (CF=ATM) ug/m3 This is the field to use for PM10 (was labeled CF=1)

* All time stamps are UTC.
PurpleAir sensors attempt to send both primary and secondary data every two minutes or so.

PA-II-SD data format

The SD Card version of the PA-II (PA-II-SD) has a built in real time clock and OPENLOG serial logger. The SD card contains data in CSV format with the following headers:
NOTE: These fields have not yet been corrected for the incorrect labels for CF=1 and CF=ATM.

UTCDateTime,mac_address,firmware_ver,hardware,current_temp_f,current_humidity,current_dewpoint_f,pressure,adc,mem,rssi,uptime,pm1_0_atm,pm2_5_atm,pm10_0_atm,pm1_0_cf_1,pm2_5_cf_1,pm10_0_cf_1,p_0_3_um,p_0_5_um,p_1_0_um,p_2_5_um,p_5_0_um,p_10_0_um,pm1_0_atm_b,pm2_5_atm_b,pm10_0_atm_b,pm1_0_cf_1_b,pm2_5_cf_1_b,pm10_0_cf_1_b,p_0_3_um_b,p_0_5_um_b,p_1_0_um_b,p_2_5_um_b,p_5_0_um_b,p_10_0_um_b

Header Descriptions

UTCDateTime: The Date and time derived from the Real Time Clock and synced with NTP where possible (in UTC).
Mac_address: The MAC address of the WiFi module on the sensor (used as an ID for the unit).
Firmware_ver: Firmware version of the control board.
Hardware: Hardware the control board has detected.
current_temp_f: Current temperature in F.
Current_humidity: Current Humidity in %.
Current_dewpoint_f: Calculated dew point in F.
Pressure: Current pressure in millibars.
Adc: The voltage reading on the analog input of the control board.
Mem: Free HEAP memory on the control board.
Rssi: WiFi signal strength in dBm
Uptime: Firmware uptime in seconds.

Pm1_0_atm: Channel A ATM PM1.0 particulate mass in ug/m3 (is actually CF=1)
Pm2_5_atm: Channel A ATM PM2.5 particulate mass in ug/m3 (is actually CF=1)
Pm10_0_atm: Channel A ATM PM10.0 particulate mass in ug/m3 (is actually CF=1)
Pm1_0_cf_1: Channel A CF=1 PM1.0 particulate mass in ug/m3 (is actually CF=ATM)
Pm2_5_cf_1: Channel A CF=1 PM2.5 particulate mass in ug/m3 (is actually CF=ATM)
Pm10_0_cf_1: Channel A CF=1 PM10.0 particulate mass in ug/m3 (is actually CF=ATM)
P_0_3_um: Channel A 0.3 micrometer particle counts per deciliter of air
P_0_5_um: Channel A 0.5 micrometer particle counts per deciliter of air
P_1_0_um: Channel A 1.0 micrometer particle counts per deciliter of air
P_2_5_um: Channel A 2.5 micrometer particle counts per deciliter of air
P_5_0_um: Channel A 5.0 micrometer particle counts per deciliter of air
P_10_0_um: Channel A 10.0 micrometer particle counts per deciliter of air

Pm1_0_atm_b: Channel B ATM PM1.0 particulate mass in ug/m3.
Pm2_5_atm_b: Channel B ATM PM2.5 particulate mass in ug/m3
Pm10_0_atm_b: Channel B ATM PM10.0 particulate mass in ug/m3
Pm1_0_cf_1_b: Channel B CF=1 PM1.0 particulate mass in ug/m3
Pm2_5_cf_1_b: Channel B CF=1 PM2.5 particulate mass in ug/m3
Pm10_0_cf_1_b: Channel B CF=1 PM10.0 particulate mass in ug/m3
P_0_3_um_b: Channel B 0.3 micrometer particle counts per deciliter of air
P_0_5_um_b: Channel B 0.5 micrometer particle counts per deciliter of air

P_1_0_um_b: Channel B 1.0 micrometer particle counts per deciliter of air
P_2_5_um_b: Channel B 2.5 micrometer particle counts per deciliter of air
P_5_0_um_b: Channel B 5.0 micrometer particle counts per deciliter of air
P_10_0_um_b: Channel B 10.0 micrometer particle counts per deciliter of air
Gas: FIRMWARE 4.10 and up: Bosch BSEC IAQ when BME680 gas sensor is present

PA-II NOTES:
Each sensor contains two identical laser counters, hence channel A and B. If these two channels do not agree to some extent then there is something wrong with one or both channels.

Plantower PMS sensor notes:
ATM is "**atmospheric**", meant to be used for outdoor applications
CF=1 is meant to be used for indoor or controlled environment applications
Original (incorrect) assumption: PurpleAir uses CF=1 values on the map. This value is lower than the ATM value in higher measured concentrations.
New (correct) statement: Due to the firmware swapping the labels, PurpleAir has inadvertently always used CF=ATM on the map. It is ATM values that are lower than CF=1 at high concentrations. This mislabeling has been present since day one and was the result of a lack of information on the sensor specs at the time we first started using the Plantower particle counters. The PurpleAir download tool had the labels switched on the 20 October 2019.

Using JavaScript to calculate the AQI for PM2.5

The AQI is calculated using the EPA's breakpoints as described in this link:
https://aqs.epa.gov/aqsweb/documents/codetables/aqi_breakpoints.html

In JavaScript, this line is how you convert PM2.5ug/m3 to AQI using the functions below:

var AQI = aqiFromPM(pm25value);
var AQIDescription = getAQIDescription(AQI); //A short description of the provided AQI
var AQIMessage = getAQIMessage(AQI); // What the provided AQI means (a longer description)

And here are the functions:

```
function aqiFromPM(pm) {

    if (isNaN(pm)) return "-";
    if (pm == undefined) return "-";
    if (pm < 0) return pm;
    if (pm > 1000) return "-";
      /*
          Good                          0 - 50         0.0 - 15.0         0.0 – 12.0
      Moderate                      51 - 100          >15.0 - 40         12.1 – 35.4
      Unhealthy for Sensitive Groups   101 – 150     >40 – 65          35.5 – 55.4
      Unhealthy                        151 – 200         > 65 – 150       55.5 – 150.4
```

```
  Very Unhealthy                  201 – 300 > 150 – 250     150.5 – 250.4
  Hazardous                           301 – 400         > 250 – 350     250.5 – 350.4
  Hazardous                           401 – 500         > 350 – 500     350.5 – 500
  */
  if (pm > 350.5) {
    return calcAQI(pm, 500, 401, 500, 350.5);
  } else if (pm > 250.5) {
    return calcAQI(pm, 400, 301, 350.4, 250.5);
  } else if (pm > 150.5) {
    return calcAQI(pm, 300, 201, 250.4, 150.5);
  } else if (pm > 55.5) {
    return calcAQI(pm, 200, 151, 150.4, 55.5);
  } else if (pm > 35.5) {
    return calcAQI(pm, 150, 101, 55.4, 35.5);
  } else if (pm > 12.1) {
    return calcAQI(pm, 100, 51, 35.4, 12.1);
  } else if (pm >= 0) {
    return calcAQI(pm, 50, 0, 12, 0);
  } else {
    return undefined;
  }

}
function bplFromPM(pm) {
if (isNaN(pm)) return 0;
if (pm == undefined) return 0;
if (pm < 0) return 0;
  /*
       Good                           0 - 50         0.0 - 15.0         0.0 – 12.0
  Moderate                      51 - 100          >15.0 - 40        12.1 – 35.4
  Unhealthy for Sensitive Groups   101 – 150     >40 – 65          35.5 – 55.4
  Unhealthy                           151 – 200         > 65 – 150       55.5 – 150.4
  Very Unhealthy                  201 – 300 > 150 – 250     150.5 – 250.4
  Hazardous                           301 – 400         > 250 – 350     250.5 – 350.4
  Hazardous                           401 – 500         > 350 – 500     350.5 – 500
  */
  if (pm > 350.5) {
  return 401;
  } else if (pm > 250.5) {
  return 301;
  } else if (pm > 150.5) {
  return 201;
  } else if (pm > 55.5) {
  return 151;
  } else if (pm > 35.5) {
  return 101;
  } else if (pm > 12.1) {
  return 51;
  } else if (pm >= 0) {
  return 0;
  } else {
  return 0;
  }

}
function bphFromPM(pm) {
//return 0;
```

```
if (isNaN(pm)) return 0;
if (pm == undefined) return 0;
if (pm < 0) return 0;
  /*
       Good                              0 - 50         0.0 - 15.0         0.0 – 12.0
  Moderate                          51 - 100           >15.0 - 40        12.1 – 35.4
  Unhealthy for Sensitive Groups   101 – 150     >40 – 65          35.5 – 55.4
  Unhealthy                              151 – 200         > 65 – 150       55.5 – 150.4
  Very Unhealthy                    201 – 300 > 150 – 250     150.5 – 250.4
  Hazardous                              301 – 400         > 250 – 350     250.5 – 350.4
  Hazardous                              401 – 500         > 350 – 500     350.5 – 500
  */
  if (pm > 350.5) {
  return 500;
  } else if (pm > 250.5) {
  return 500;
  } else if (pm > 150.5) {
  return 300;
  } else if (pm > 55.5) {
  return 200;
  } else if (pm > 35.5) {
  return 150;
  } else if (pm > 12.1) {
  return 100;
  } else if (pm >= 0) {
  return 50;
  } else {
  return 0;
  }

}

function calcAQI(Cp, Ih, Il, BPh, BPl) {

  var a = (Ih - Il);
  var b = (BPh - BPl);
  var c = (Cp - BPl);
  return Math.round((a/b) * c + Il);

}


function getAQIDescription(aqi) {
  if (aqi >= 401) {
    return 'Hazardous';
  } else if (aqi >= 301) {
    return 'Hazardous';
  } else if (aqi >= 201) {
    return 'Very Unhealthy';
  } else if (aqi >= 151) {
    return 'Unhealthy';
  } else if (aqi >= 101) {
    return 'Unhealthy for Sensitive Groups';
  } else if (aqi >= 51) {
    return 'Moderate';
  } else if (aqi >= 0) {
    return 'Good';
```

```
    } else {
      return undefined;
    }
  }

  function getAQIMessage(aqi) {
    if (aqi >= 401) {
      return '>401: Health alert: everyone may experience more serious health effects';
    } else if (aqi >= 301) {
      return '301-400: Health alert: everyone may experience more serious health effects';
    } else if (aqi >= 201) {
      return '201-300: Health warnings of emergency conditions. The entire population is more likely to be affected. ';
    } else if (aqi >= 151) {
      return '151-200: Everyone may begin to experience health effects; members of sensitive groups may experience more serious health effects.';
    } else if (aqi >= 101) {
      return '101-150: Members of sensitive groups may experience health effects. The general public is not likely to be affected.';
    } else if (aqi >= 51) {
      return '51-100: Air quality is acceptable; however, for some pollutants there may be a moderate health concern for a very small number of people who are unusually sensitive to air pollution.';
    } else if (aqi >= 0) {
      return '0-50: Air quality is considered satisfactory, and air pollution poses little or no risk';
    } else {
      return undefined;
    }
  }
```