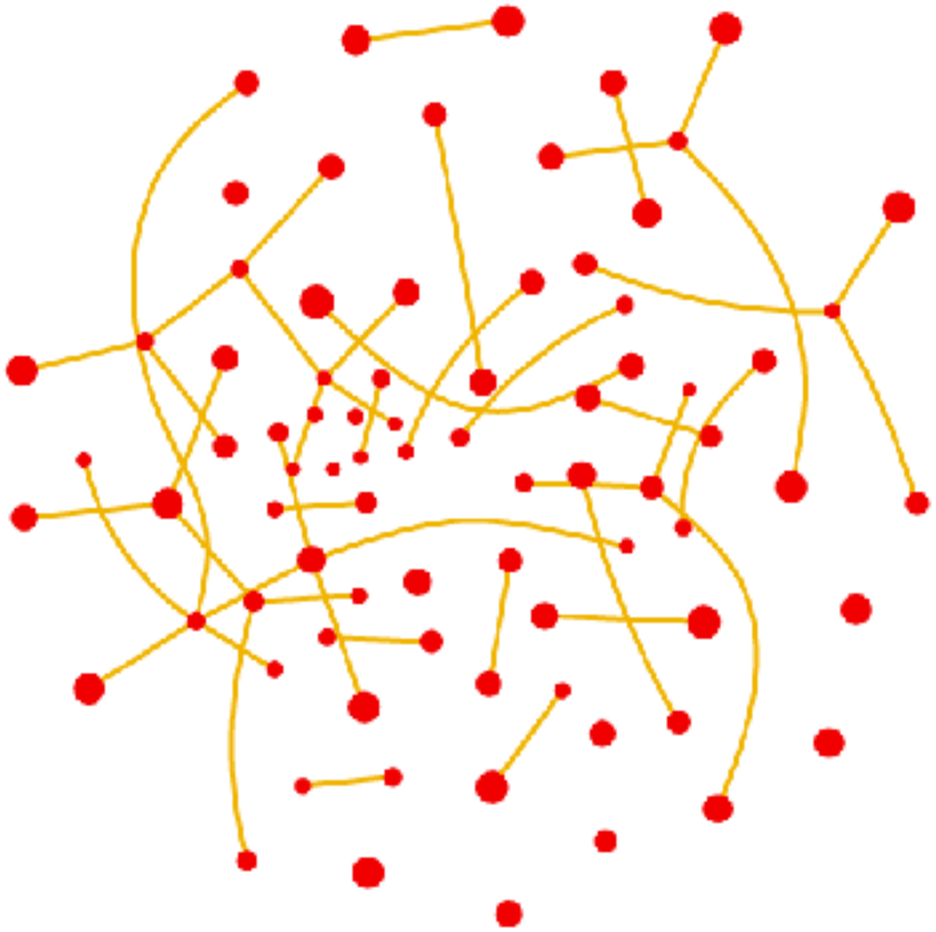


Working with Sensors

Working with sensor data

Garbage in

Garbage out

Issues with sensors

Variation across sensors / consistency

Noise/spurious readings

Sensitivity to change

Inconsistent ranges - not always 0 - 4095

+ Lots more.

Making it useful

Calibration is an important element of working with sensors.

It improves the:

- Consistency

- Reliability

- Precision

of readings from a sensor by making sure you have ‘checked’ it with a real world source.

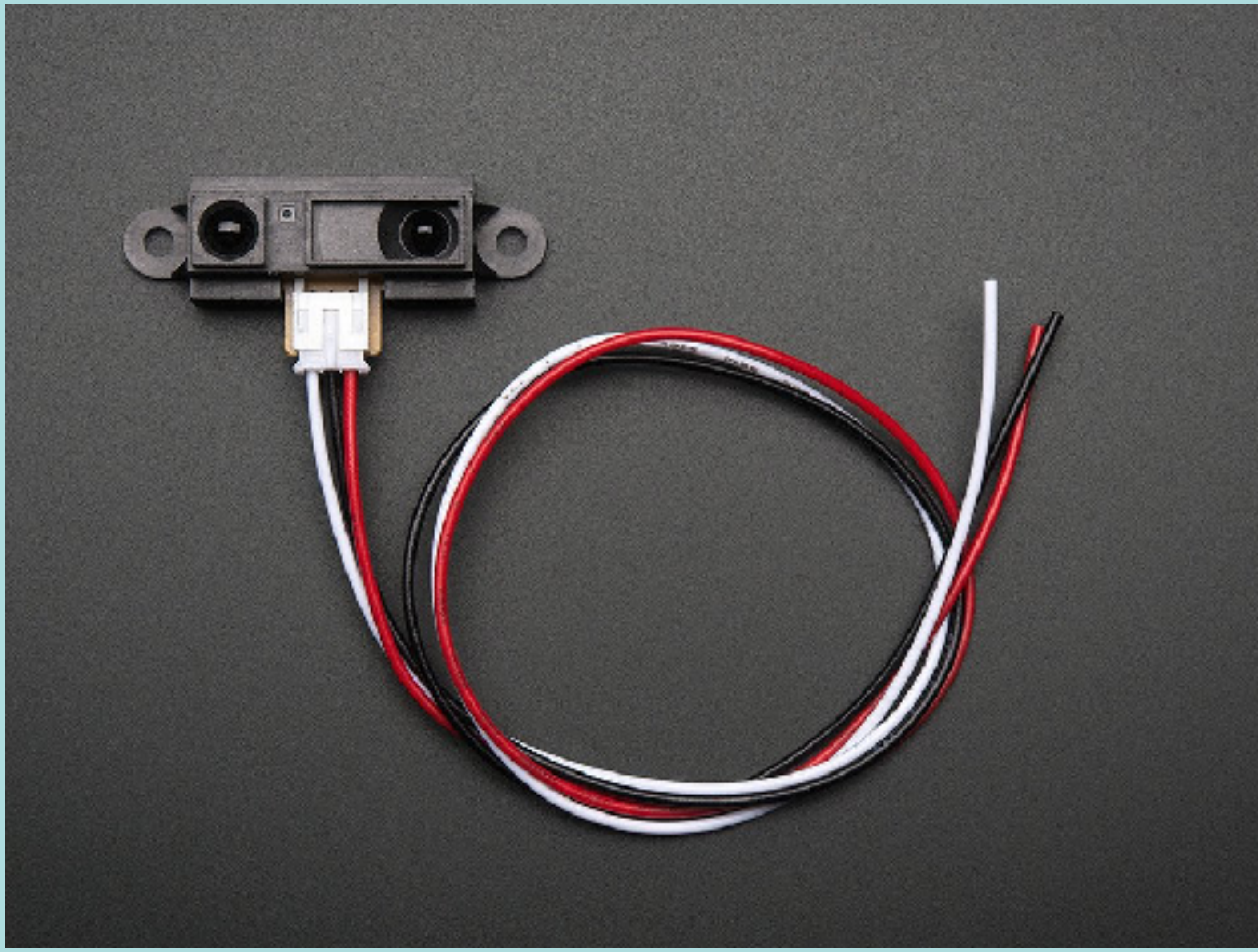
Making it useful

Two Examples

Example One: External Calibration

Maxbotix Ultrasonic Rangefinder - LV-EZ1 - LV-EZ1

To use, connect black wire to ground, red wire to 5V and white wire to analog input. The analog voltage out will range from 3V when an object is only 4" (10 cm) away and 0.4V when the object is 32" (80 cm) away



https://www.adafruit.com/products/164

Example One: External Calibration

Maxbotix Ultrasonic Rangefinder - LV-EZ1 - LV-EZ1

We know the scale (4" (10 cm) away and 0.4V when the object is 32" (80 cm) away)

It’s linear. Awesome.

But…

Does our sensor fully match?

https://www.adafruit.com/products/164

Example One: External Calibration

Maxbotix Ultrasonic Rangefinder - LV-EZ1 - LV-EZ1

Problem: Drift

Solution: One point calibration

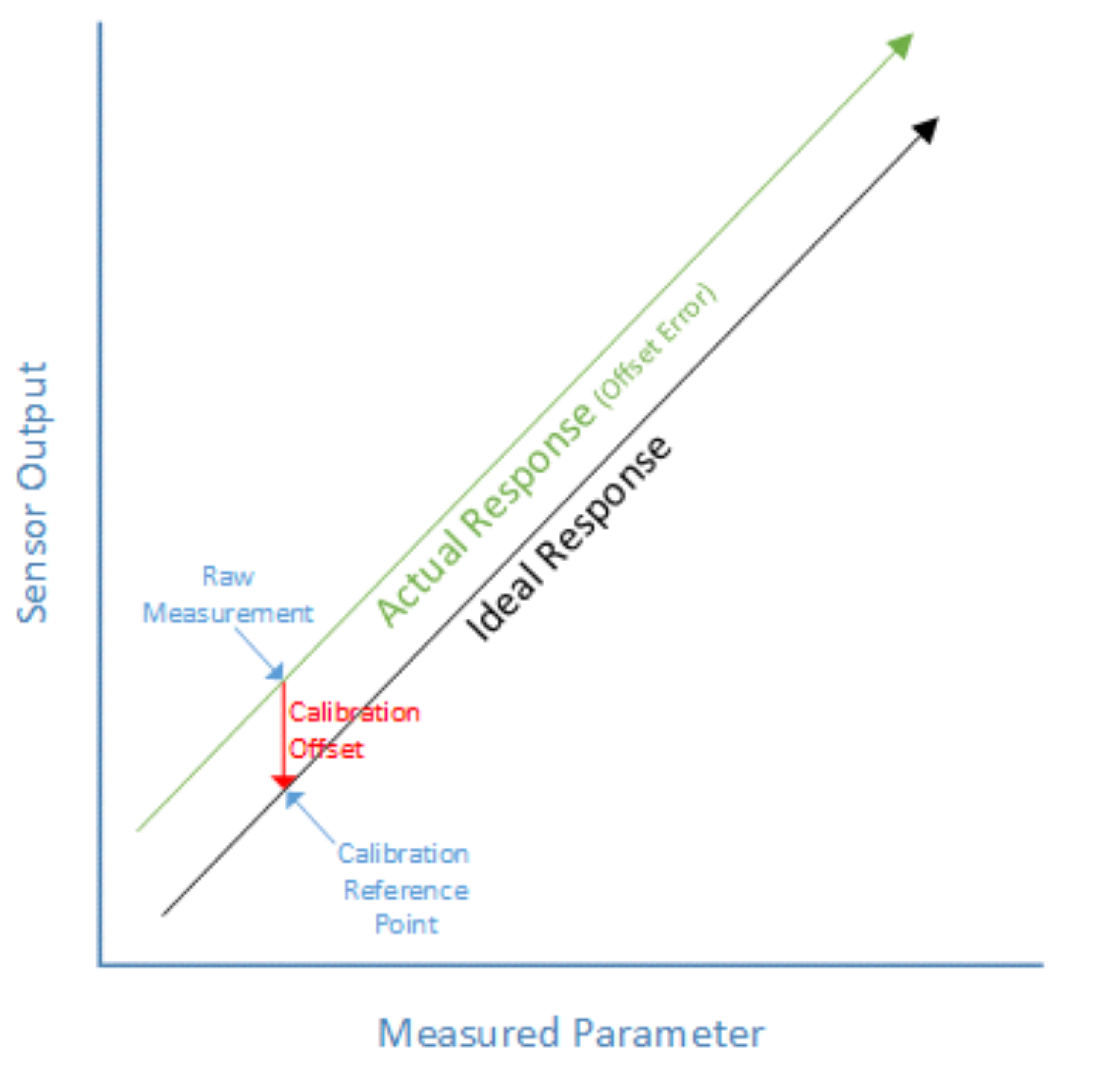
Solved by getting a reference reading that gives a real world validation of how much the sensor varies from the normal.

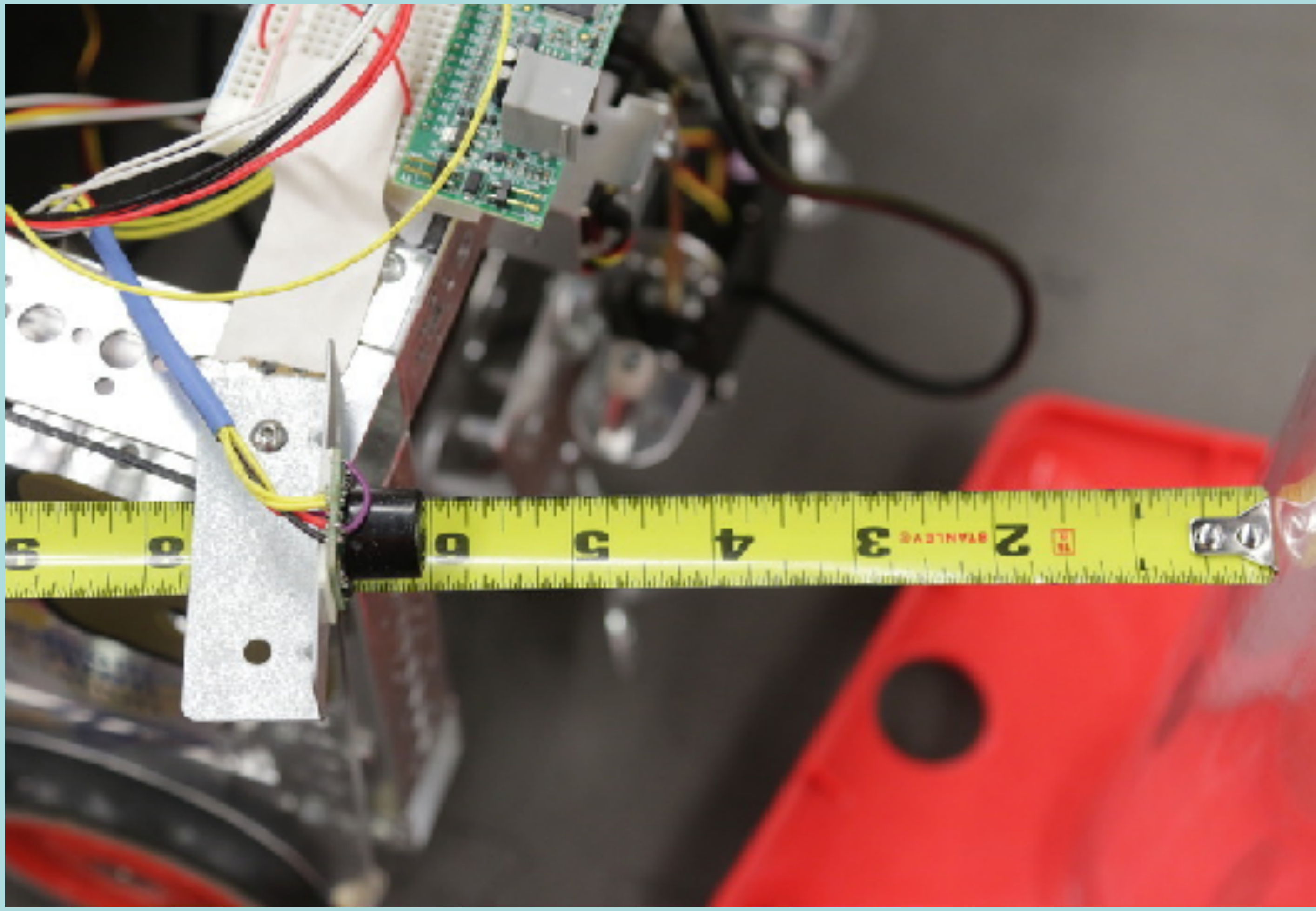
Then you can use this to offset values in your code.

https://www.adafruit.com/products/1343?gclid=CL_307WijtMCFVSHswod7lcHXg

Example One: External Calibration

Maxbotix Ultrasonic Rangefinder - LV-EZ1 - LV-EZ1





https://learn.adafruit.com/calibrating-sensors/single-point-calibration

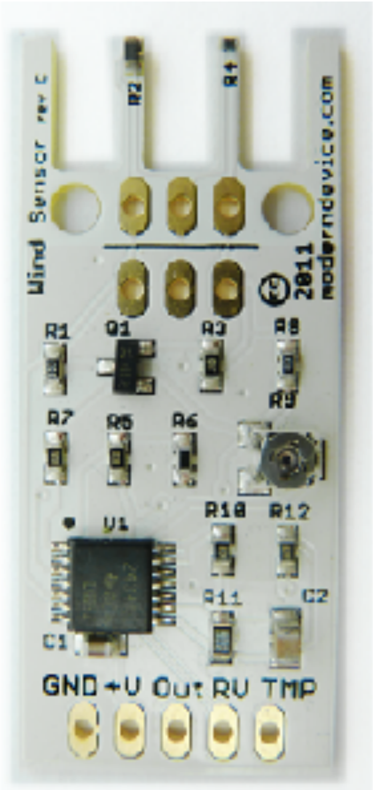
Example One: External Calibration

Lots of components need this kind of calibration

https://learn.adafruit.com/calibrating-sensors/single-point-calibration

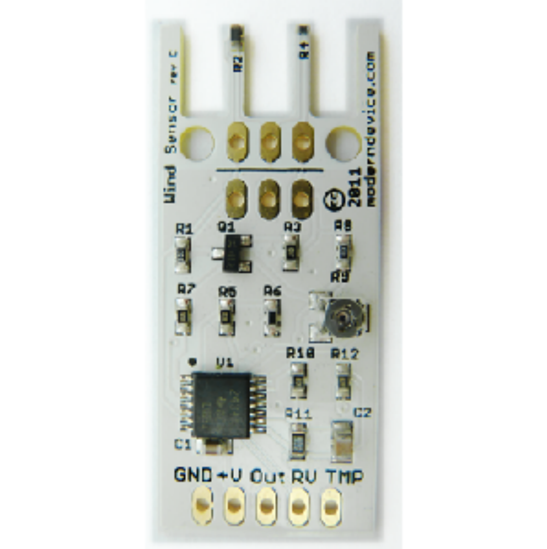
Example Two: Calibration to Current Environment

Some sensors live in environments with changing conditions and you need to get a 'baseline' reading to know what the ranges are.



https://www.adafruit.com/products/164

Example Two: Calibration to Current Environment

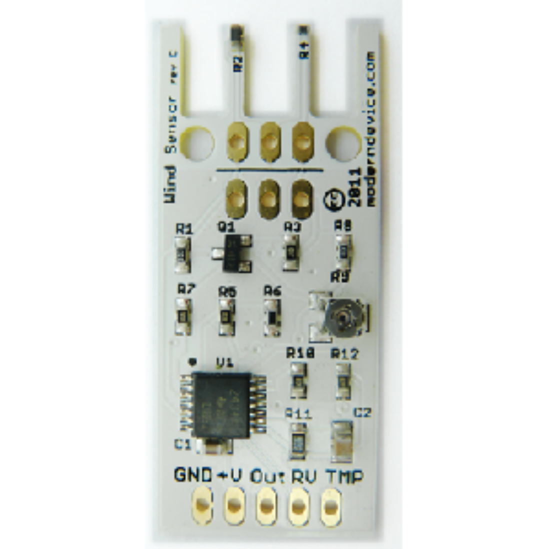


Analog: Isn’t always 0-4095

What is our ***baseline***?

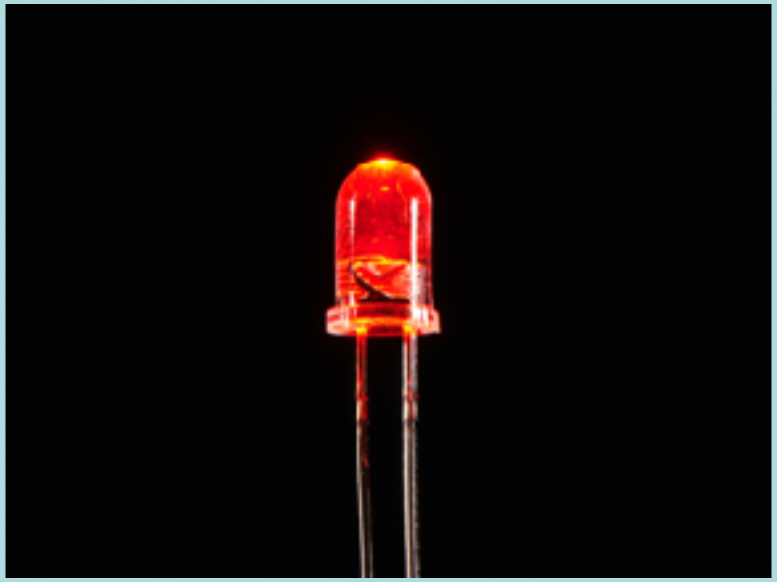
https://www.adafruit.com/products/164

Example Two: Calibration to Current Environment



Analog: Isn't always 0-4095

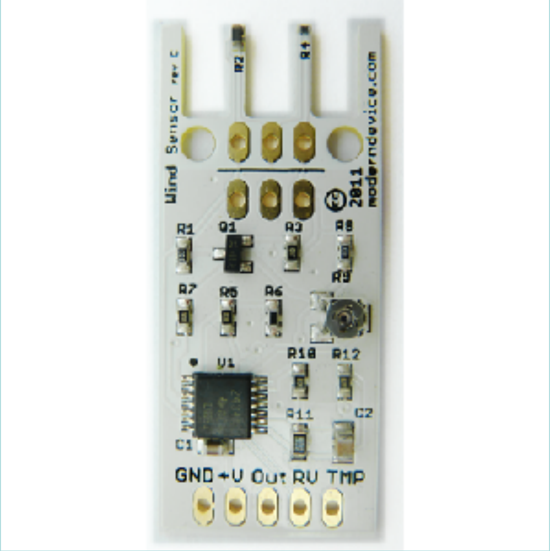
What is our *baseline*?



How do we map this into other information spaces reliably?

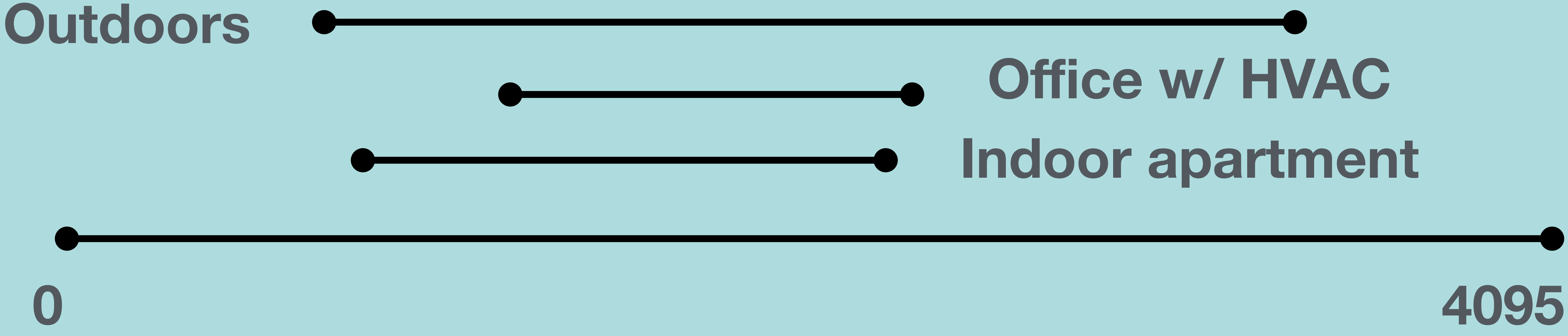
https://www.adafruit.com/products/164

Example Two: Calibration to Current Environment



Analog: Isn’t always 0-4095

What is our ***baseline***?


Outdoors

Office w/ HVAC

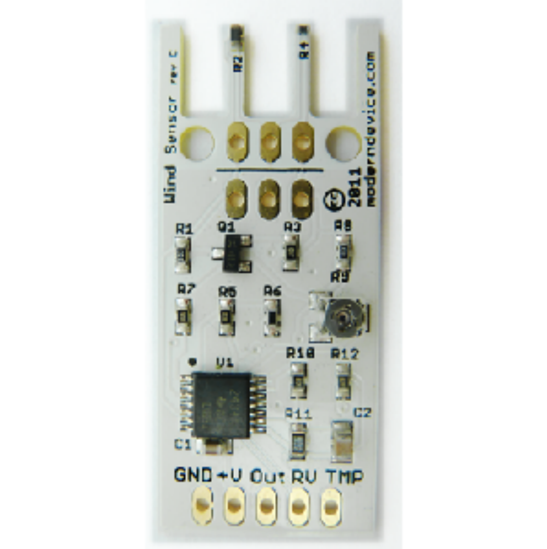
Indoor apartment

0

4095

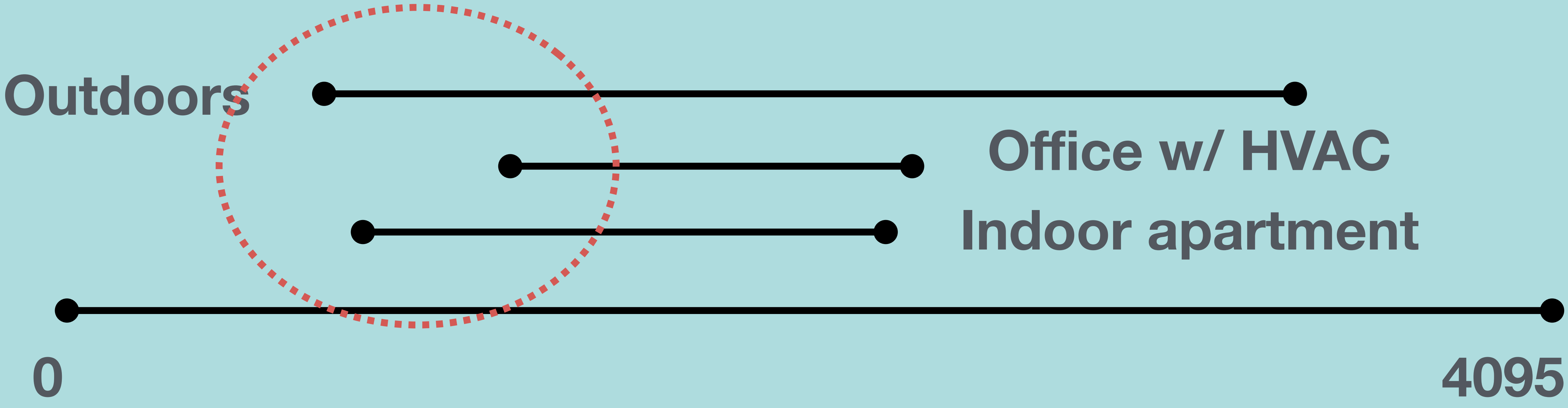
https://www.adafruit.com/products/164

Example Two: Calibration to Current Environment



Analog: Isn’t always 0-4095

What is our ***baseline***?



Outdoors

Office w/ HVAC

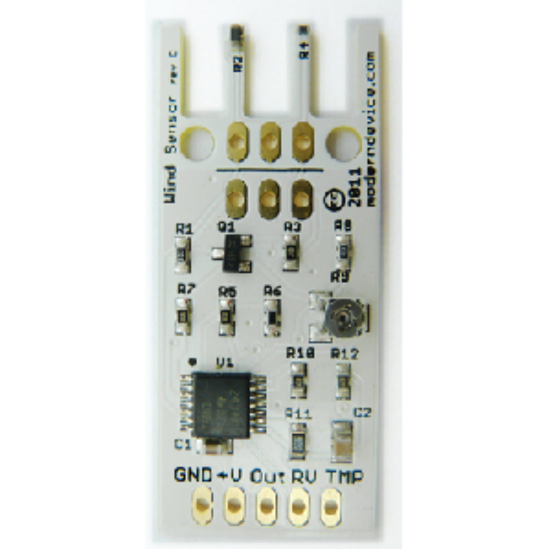
Indoor apartment

0

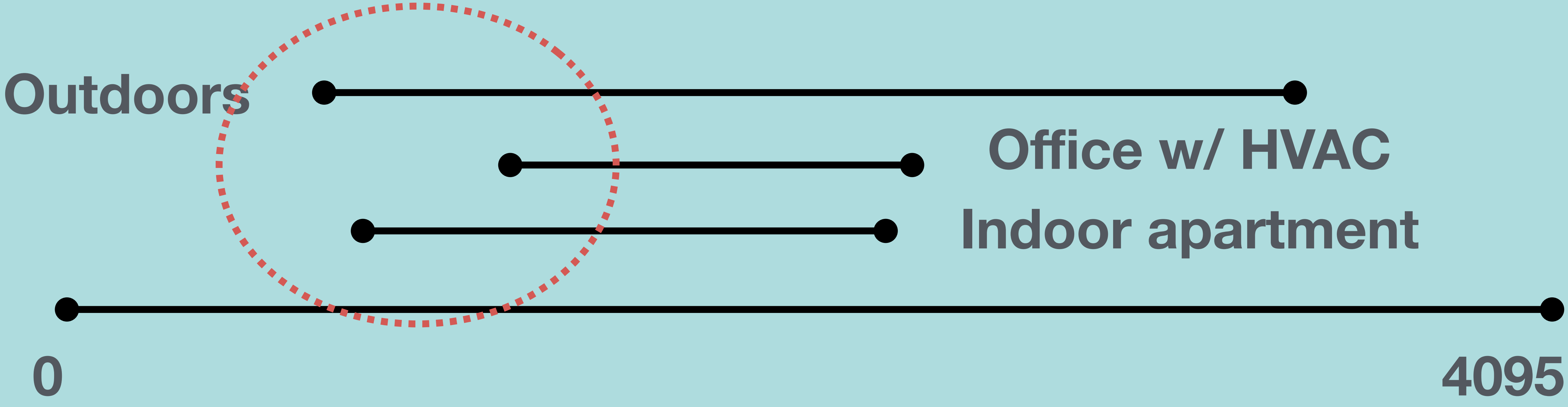
4095

https://www.adafruit.com/products/164

Example Two: Calibration to Current Environment



When we startup:
Let’s collect a bunch of samples.
See what our readings are
Use this to set our baseline



Outdoors

Office w/ HVAC

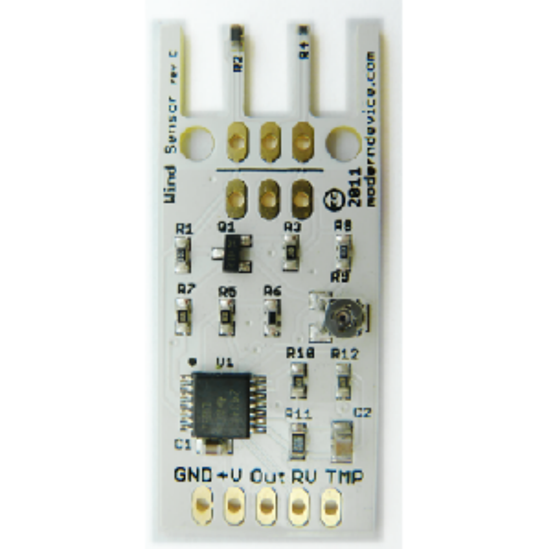
Indoor apartment

0

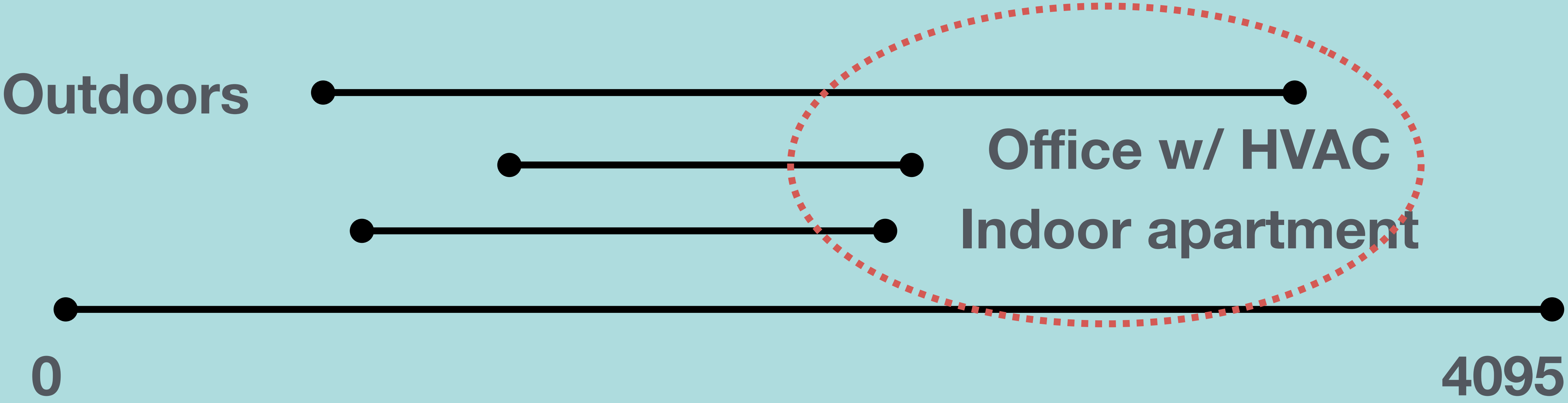
4095

https://www.adafruit.com/products/164

Example Two: Calibration to Current Environment



We can also look for the upper bound of our readings too



https://www.adafruit.com/products/164

Why Calibrate?

To account for user variation (FSR sensitivity)

To account for context variation (background noise, etc.)

To account for sensor drift / variation in technical components

+ lots more

When Calibrating

Know your sensor, it's limits and it's purpose

Ask is it actually needed

How much precision do you need?
i.e. how much work should you put into calibration?

Who does the work to calibrate?
You as system developer (Q&A and testing)
The user in configuring for them (during setup?)
The system (ongoing, automatic)

Know your Data

If your application is data driven:

Get to know that data now

Know your Data

If your application is data driven:

Get to know that data now

To do this

Collect some data from real world instances

Collecting Samples

1. The Easy way

Create a basic circuit with your sensor

Write out the sensor readings to the Serial Monitor

Connect via USB, pop open terminal and pipe out the output

```
OSX

screen /dev/tty.usbmodem* 9600 >> someFile.txt

(or whatever matches the string you see in the
Particle Dev’s serial monitor
```

Collecting Samples

1. The Easy way

Create a basic circuit with your sensor
Write out the sensor readings to the Serial Monitor
Connect via USB, pop open terminal and pipe out the output

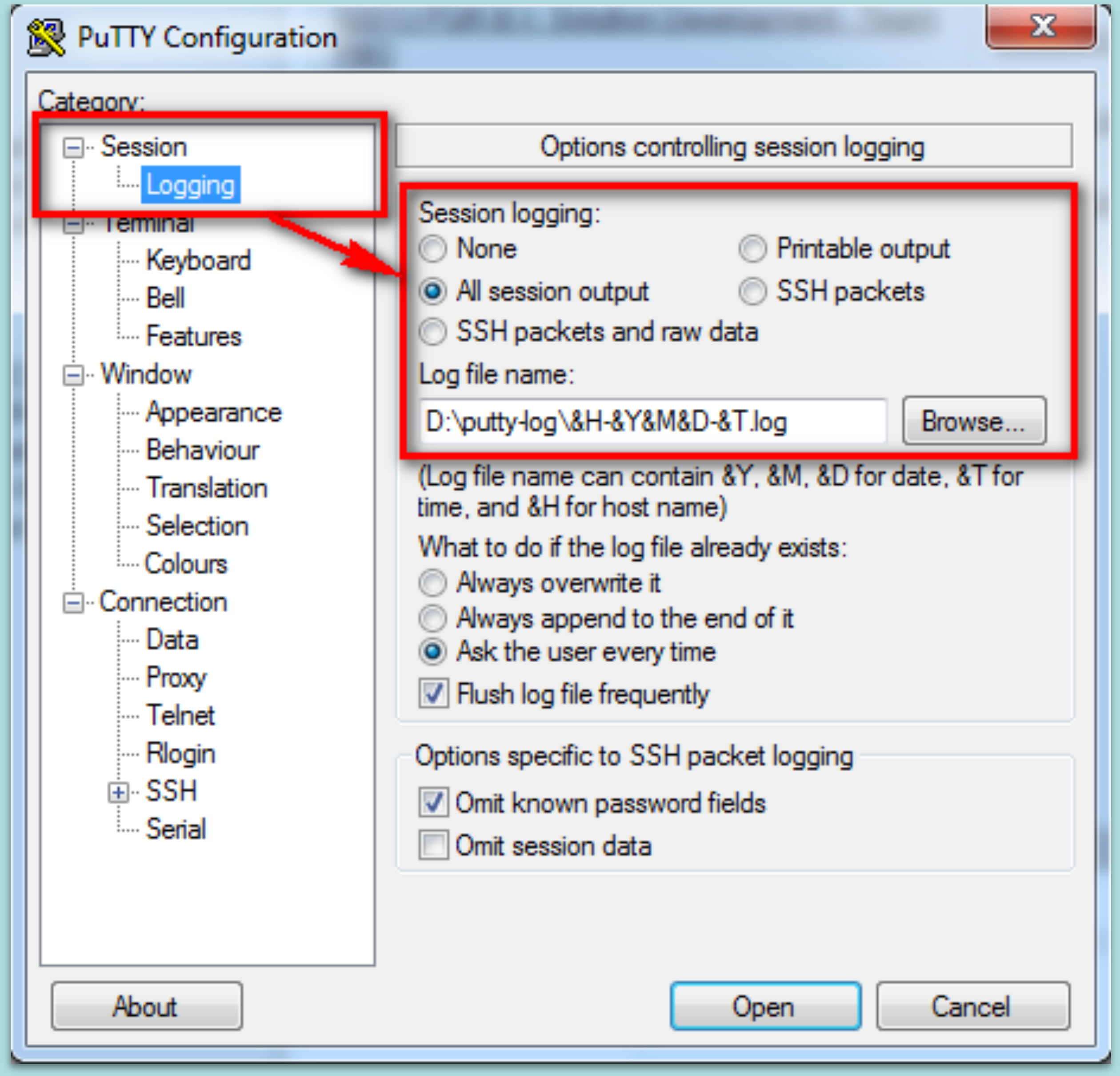
```
CLI

particle serial monitor >> filename.txt

(or whatever matches the string you see in the
Particle Dev’s serial monitor
```

Collecting Samples

Windows:



Options controlling session logging

Session logging:

○ None ○ Printable output

● All session output ○ SSH packets

○ SSH packets and raw data

Log file name:

D:\putty-log\&H-&Y&M&D-&T.log Browse...

(Log file name can contain &Y, &M, &D for date, &T for time, and &H for host name)

What to do if the log file already exists:

○ Always overwrite it

○ Always append to the end of it

● Ask the user every time

☑ Flush log file frequently

Options specific to SSH packet logging

☑ Omit known password fields

□ Omit session data

Collecting Samples

Format your Serial output as CSV
—-> Comma Separated Values
Nothing to serial <u>BUT</u> the data you want

```
void dumpInfo(){
  // Start by outputting the time
  // Thur Apr 6 07:08:47 2016
  Serial.print( Time.timeStr() );
  // each value must be followed by a comma
  Serial.print( “,”);
  // write out each sensor value you
  // want to keep a log of
  Serial.print( sensorReading );
  Serial.print( “,” )
  Serial.print( anotherSensor );
  // the last one doesn’t need a comma after it
  // but you do need to start a new line
  Serial.println( “” );
}
```

Collecting Samples

2. The Less Easy Way

Use Particle.publish and listen for events using cURL in the command line

OSX

```
curl ‘https://api.particle.io/v1/events/event-name?
access_token=[access_token]'
```

Collecting Samples

2. The Less Easy Way

Use Particle.publish and listen for events using cURL in the command line

You’ll have to clean it up once you’ve got it all

OSX

```
event: my-event-name
data:
{"data":"Info","ttl":"60","published_at":"2017-04-06
T16:43:31.787Z","coreid":"31002d001447343338333633"}
```

Collecting Samples

3. The Involved but Useful way

Log it to Google Cloud or a similar platform for data capture

```
Visit:


https://github.com/rickkas7/google_cloud_tutorial
```

Collecting Samples

3. The Involved but Useful way

Log it to Google Cloud or a similar platform for data capture

It’s work. A lot of work.

Visit:

https://github.com/rickkas7/google_cloud_tutorial