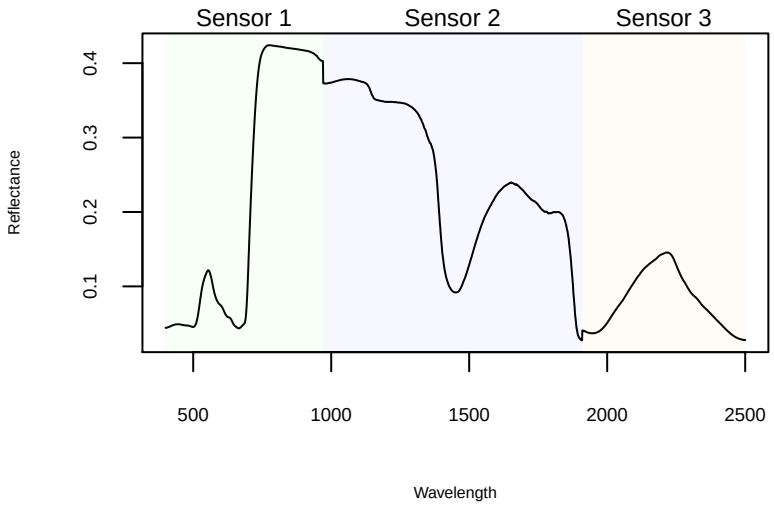
How to match sensors in raw spectra

Jose Eduardo Meireles and Anna Schweiger

Raw spectra may have abrupt reflectance 'jumps' in the interface between sensors, and those need to be matched (spliced). In a raw spectrum collected with a 3-sensor instrument (say SVC), these regions may look like this:



In this case, sensor 1 goes from 400 to 970nm, sensor 2 from 971 to 1909 and sensor 3 from 1910 to 2500. These **boundaries will vary** depending on the individual instrument, so you'll have to plot and zoom your spectra to figure out where the sensor limits are.

Aside from a spectra object, the `match_sensors` function takes a few other important arguments:

1. `splice_at`: A numeric vector with the start wavelengths of the 2nd and 3rd sensors. As said before, you'll need to plot your spectra to find out what these wavelengths are. This argument must have length 1 (defines 2 sensors) or length 2 (defines 3 sensors).
2. `fixed_sensor`: Which sensor should be used as a reference? If `splice_at` is length 2 (3 sensor case), `fixed_sensor` must be the 2nd one. Otherwise, you can choose to fix either the 1st or 2nd sensor.
3. `interpolate_wvl`: Extention around `splice_at` wavelengths over which the matching algorithm computes the conversion factors.

Here are a few combinations of different matching parameters to give you an idea of how the function is called and what the outputs look like:

```
# 3-sensor case, fixing sensor 2
a = match_sensors(spec_with_jump, splice_at = c(971, 1910), fixed_sensor = 2,
                  interpolate_wvl = c(2, 2))

# 3-sensor case, tried to fix sensor 1 but sensor 2 gets fixed anyways!
b = match_sensors(spec_with_jump, splice_at = c(971, 1910), fixed_sensor = 1,
                  interpolate_wvl = c(2, 2))
```

```r
# Define 2 sensors divided at 971nm. Fixing the 2nd sensor
c = match_sensors(spec_with_jump, splice_at = c(971),       fixed_sensor = 2,
                  interpolate_wvl = c(2, 2))

# Define 2 sensors divided at 971nm. Fixing the 1st sensor
d = match_sensors(spec_with_jump, splice_at = c(971),       fixed_sensor = 1,
                  interpolate_wvl = c(2, 2))

# Define 2 sensors divided at 1910nm. Fixing the 2nd sensor
e = match_sensors(spec_with_jump, splice_at = c(1910),      fixed_sensor = 2,
                  interpolate_wvl = c(2, 2))

# Define 2 sensors divided at 1910nm. Fixing the 1st sensor
f = match_sensors(spec_with_jump, splice_at = c(1910),      fixed_sensor = 1,
                  interpolate_wvl = c(2, 2))
```