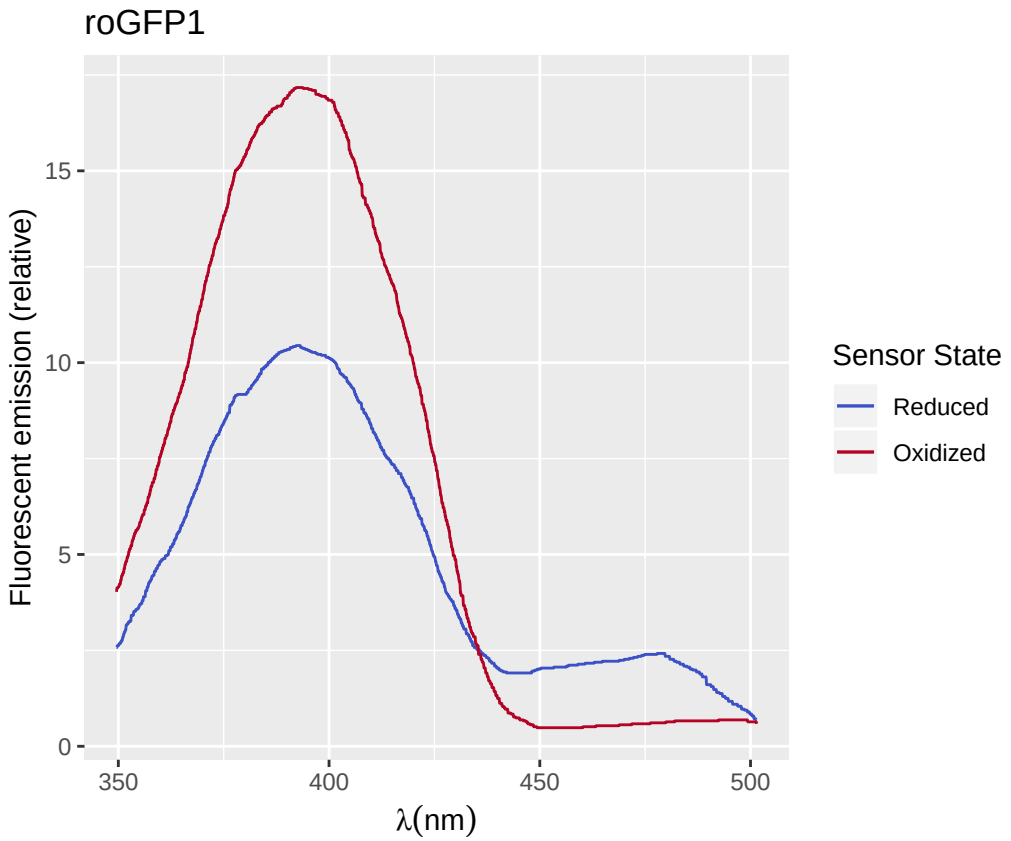
# Spectral Correction

Consider some spectra. Let's use roGFP1:

```
roGFP1_data <- read.csv("../Raw_Spectra/rogfp1.csv", header = TRUE)
roGFP1_spectra <- spectraMatrixFromValues(
    lambdas_minimum = roGFP1_data$Lambda_Reduced,
    values_minimum = roGFP1_data$Values_Reduced,
    lambdas_maximum = roGFP1_data$Lambda_Oxidized,
    values_maximum = roGFP1_data$Values_Oxidized)
gfp1_spectraPlot <- plotSpectra(roGFP1_spectra, "Reduced", "Oxidized") + ggtitle("roGFP1")
gfp1_spectraPlot +
  theme(aspect.ratio=1)
```




Let's say when that sensor is oxidized, it is actually 95% oxidized. Similarly, when the sensor is reduced, it's actually only 90% reduced. This is an extreme case for the sake of having nice(r) numbers

We can describe the *95% oxidized* curve's intensity as a function of the **limiting** oxidized curve's intensity and and the **limiting** reduced curve's intensity.

First, let's look at JUST the intensity at $400nm$. We'll then expand to any wavelength.

$$I_{400} = I_X * P_X + I_R * P_R$$

Where $I_{400}$ is the intensity at $400nm$, $I_X$ and $I_R$ are the intensity of the limiting oxidized and reduced curves, respectively, and $P_X$ and $P_R$ are the preportions of the oxidized and reduced curves that $I_{400}$ comes from.

We have two unknowns in that equation: $I_X$ and $I_R$. To solve for two unknowns, we'll need two equations. That's why we'll need to use intensity values from two non-limiting curves–e.g. the 95% oxidized and 90% reduced curves in this example.

Let's fill in some blanks. What's the intensity at $400nm$ of our two curves?

```
# Oxidized curve intensity at 400
roGFP1_spectra@values_maximum[abs(roGFP1_spectra@lambdas - 400) < 0.05]
```

```
## [1] 16.84076
```

```
# Reduced curve intensity at 400
roGFP1_spectra@values_minimum[abs(roGFP1_spectra@lambdas - 400) < 0.05]
```

```
## [1] 10.11465
```

So our intensities are 16.84 for our 95% oxidized curve and 10.11 for our 90% reduced curve. We can now make a system of equations:

$$16.84 = I_X * 0.95 + I_R * 0.05$$

and

$$10.11 = I_X * 0.10 + I_R * 0.90$$

We can describe this as an augmented matrix:

$$\left[\begin{array}{cc|c} 0.95 & 0.05 & 16.84 \\ 0.10 & 0.90 & 10.11 \end{array}\right]$$

And R is nice enough to be able to solve matricies for us

```
solve(matrix(c(0.95, 0.10, 0.05, 0.90), nrow = 2), c(16.84, 10.11))
```

```
## [1] 17.235882  9.318235
```

So our values were 16.84 and 10.11 but they should have been 17.24 and 9.32.

Cool, so we can apply this general principle for all of the intensity values in our spectra.

First, we get the intensity values across all lambdas (I'll just show a few)

```
# Oxidized curve intensity
head(roGFP1_spectra@values_maximum)
```

```
## [1] 4.025478 4.127389 4.127389 4.127389 4.152866 4.152866
```

```
# Reduced curve intensity 0
head(roGFP1_spectra@values_minimum)
```

```
## [1] 2.573248 2.573248 2.624204 2.624204 2.624204 2.624204
```

We can generally say:

$$I_1 = I_X * 0.95 + I_R * 0.05$$

and

$$I_2 = I_X * 0.10 + I_R * 0.90$$

For all intensity values $I$ from the 1st (95% oxidized) and 2nd (90% reduced) curves.

That gives us a matrix for every intensity value, and we can solve all of those:

```
new_maximum <- c()
new_minimum <- c()

interleaved_intensities <- c(rbind(roGFP1_spectra@values_maximum, roGFP1_spectra@values_minimum))

for ( index in seq(1,length(interleaved_intensities),2) ) {
    b <- interleaved_intensities[index:(index+1)]
    a <- matrix(c(0.95, 0.10, 0.05, 0.90), nrow = 2)
    answer <- solve(a, b)

    new_maximum <- c(new_maximum, answer[1])
    new_minimum <- c(new_minimum, answer[2])

}

head(new_maximum)
```

```
## [1] 4.110903 4.218809 4.215811 4.215811 4.242788 4.242788
```

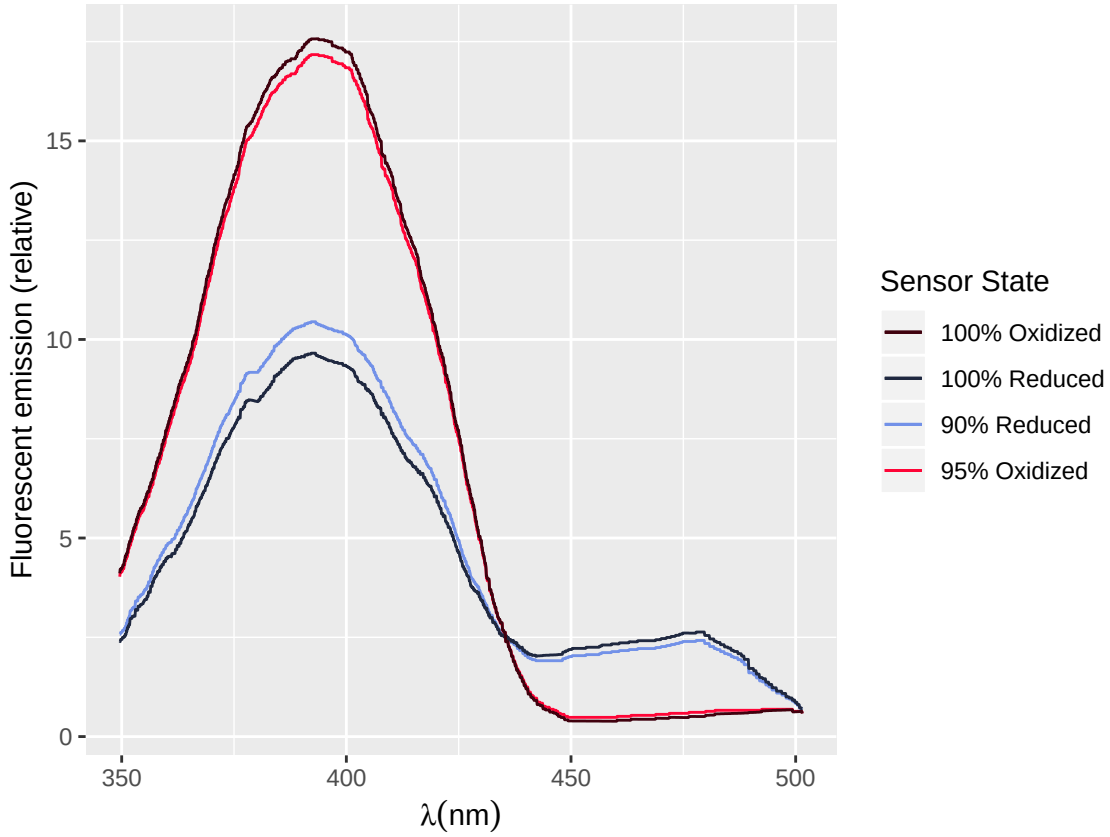
```
head(new_minimum)
```

```
## [1] 2.402398 2.390408 2.447359 2.447359 2.444361 2.444361
```

Plot of the adjusted spectra:

```
roGFP1_adjSpectra <- new("sensorSpectra", lambdas = roGFP1_spectra@lambdas,
                         values_maximum = new_maximum,
                         values_minimum = new_minimum)

ggplot() +
    geom_line(aes(x = roGFP1_spectra@lambdas,
                  y = roGFP1_spectra@values_minimum, color = "90% Reduced")) +
    geom_line(aes(x = roGFP1_spectra@lambdas,
                  y = roGFP1_spectra@values_maximum, color = "95% Oxidized")) +
    geom_line(aes(x = roGFP1_adjSpectra@lambdas,
                  y = roGFP1_adjSpectra@values_minimum, color = "100% Reduced")) +
    geom_line(aes(x = roGFP1_adjSpectra@lambdas,
                  y = roGFP1_adjSpectra@values_maximum, color = "100% Oxidized")) +
    scale_color_manual(name = "Sensor State",
                       values = c("90% Reduced" ="#7291E8",
                                  "100% Reduced" ="#1F2840",
                                  "95% Oxidized" ="#FF0537",
                                  "100% Oxidized" ="#40010E")) +
    xlab(expression(lambda (nm)))  +
    ylab("Fluorescent emission (relative)") +
    theme(aspect.ratio=1)
```




And I can make a function that generalizes this method:

```
fix_spectra <- function(curve_1, pmax_1, curve_2, pmax_2) {
    new_maximum <- c()
    new_minimum <- c()

interleaved_intensities <- c(rbind(curve1, curve2))

for ( index in seq(1,length(interleaved_intensities),2) ) {
    b <- interleaved_intensities[index:(index+1)]
    a <- matrix(c(pmax_1, 1-pmax_1, pmax_2, 1-pmax_2), nrow = 2)
    answer <- solve(a, b)

    new_maximum <- c(new_maximum, answer[1])
    new_minimum <- c(new_minimum, answer[2])

}

return(data.frame(new_maximum = new_maximum, new_minimum = new_minimum))

}
```