Introduction to respiratoR

J. Ryan Shipley

25 November, 2019

The respiratoR package was created assist with the processing and analysis of respirometry data in a reproductible format, R. It provides a series of functions that assist the user with drift correction from baseline data, regardless of gas type or whether single channel or multiplexed setup, and the calculation of metabolic rates for a variety of flow configurations.The package is not to be used as a black box, but rather as a tool during the researcher supervised workflow when processing raw gas concentration values into meaningful data.

Installation of package and dependencies

The current package is (and will possibly forever) only be available via Github. It also loads *ggplot2* and *cowplot* for creating attractive output plots, these will be later included as internal dependencies. Unfortunately, you will have to load them manually for now.

```
library(usethis)
  use_git_config(user.name = "Trollock", user.email = "jrs566@cornell.edu")
library(respiratoR)
library(ggplot2)
library(cowplot)
#these will need to be loaded now, later they will be included as dependecies
library(dplyr)
library(tidyr)
library(zoo)
library(data.table)
```

Default dataset

Load the default dataset into R.

```
data(metabolic_data)
```

Filling empty multiplexed data channels

```
head(metabolic_data)
```

```
##     Oxygen        CO2      kPa     Flow      WVP  Deg_C Marker
## 1 20.96194 0.07086053 98.33588 285.6372 1.574868 33.225     66
## 2 20.95897 0.06601670 98.20837 284.9496 1.576118 33.230     -1
## 3 20.96093 0.07054803 98.32838 285.3716 1.575180 33.235     -1
## 4 20.96007 0.06523544 98.13713 284.9809 1.576430 33.233     -1
## 5 20.95851 0.06148539 98.22713 284.6841 1.577680 33.223     -1
## 6 20.95897 0.06257915 98.20837 285.0903 1.577680 33.222     -1
```

The dataset has the expected columns of **Oxygen**, **CO2**, **kPa**, **Flow**, **WVP**, **Deg_C**, and **Marker**. All of these are typical outputs from commercial respirometry data collection equipment (hint… Expedata), with the exception of the last column, **Marker**. This is a column that designates whenever the Oxygen/Carbon Dioxide analyzer is in a multiplexed configuration and is switching channels to sample another chamber. It only sends an output when the channel changes, and it we want to fill these values for the *entire* time a channel is being sampled.

The function `fill_multiplex` requires the name of the dataframe, the channel designating samples (in this case "Marker"), the value or values of non informative data (in this case -1), and finally the value of baseline channels (66 and 53, respectively). Designating the baseline channels now will make it easier to drift correct the data in the next step.

```
metabolic_data <- fill_multiplex(metabolic_data, "Marker", -1, c(66, 53))
```

We have done two things with this function, 1) we created an index in the new first column of the dataframe that is the smaple number and 2) we have filled the **Marker** column with values for animal chambers and *Baseline* indicating those samples that are for baseline drift correction.

```
head(metabolic_data)
```

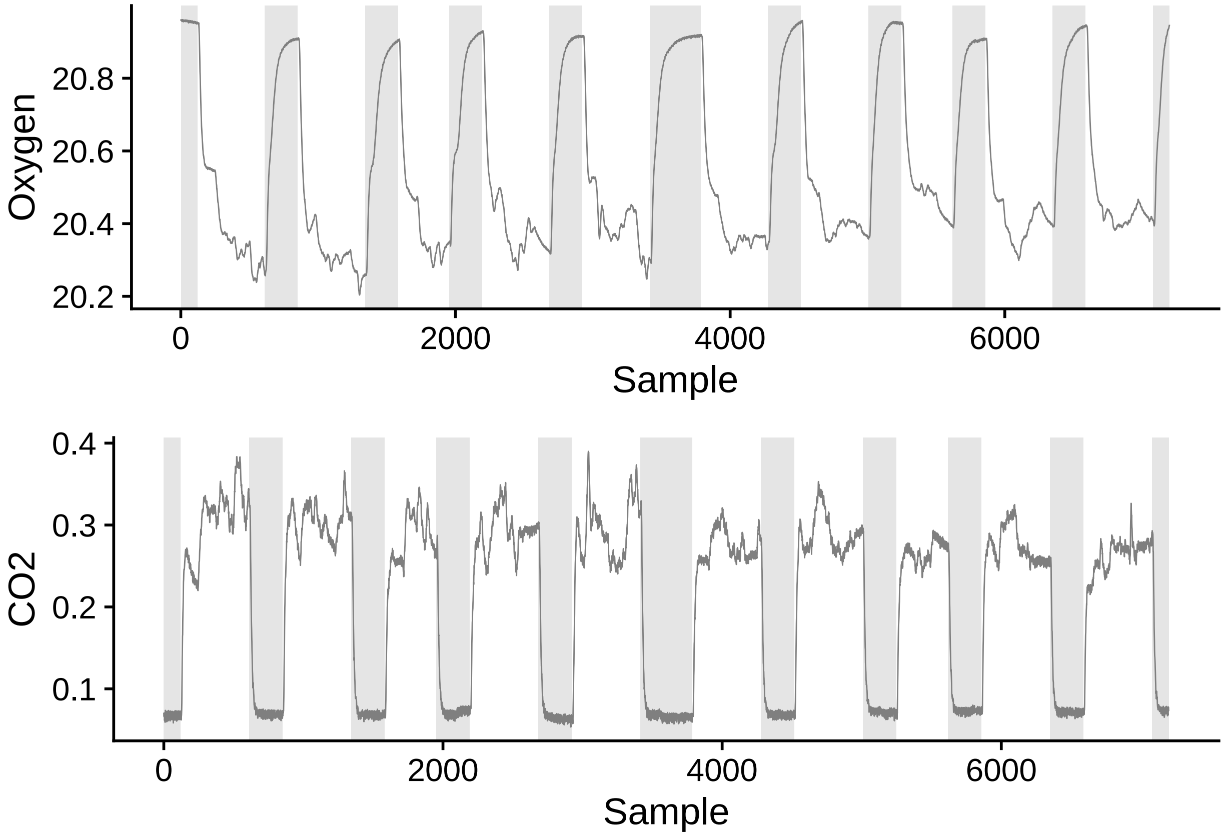
```
##   Sample   Oxygen        CO2      kPa     Flow      WVP  Deg_C   Marker
## 1      1 20.96194 0.07086053 98.33588 285.6372 1.574868 33.225 Baseline
## 2      2 20.95897 0.06601670 98.20837 284.9496 1.576118 33.230 Baseline
## 3      3 20.96093 0.07054803 98.32838 285.3716 1.575180 33.235 Baseline
## 4      4 20.96007 0.06523544 98.13713 284.9809 1.576430 33.233 Baseline
## 5      5 20.95851 0.06148539 98.22713 284.6841 1.577680 33.223 Baseline
## 6      6 20.95897 0.06257915 98.20837 285.0903 1.577680 33.222 Baseline
```

Visual inspection of baseline data

We want to visually inspect the metabolic data and look at the quality of the baseline data. We need to create a dataframe with the beginning and end times for each baseline period. We can use the function `base_bg_rect` to do this, and then plot the results in ggplot. All this function requires in the name of the respirometry data dataframe and the name of the column with the multiplexer markers.

```
base_rect <- base_bg_rect(metabolic_data, "Marker")
```

Here we can see the oxygen and carbon dioxide values as they equilibrate with local atmospheric values. There is however, some drift evident in the data and we want to correct for this.



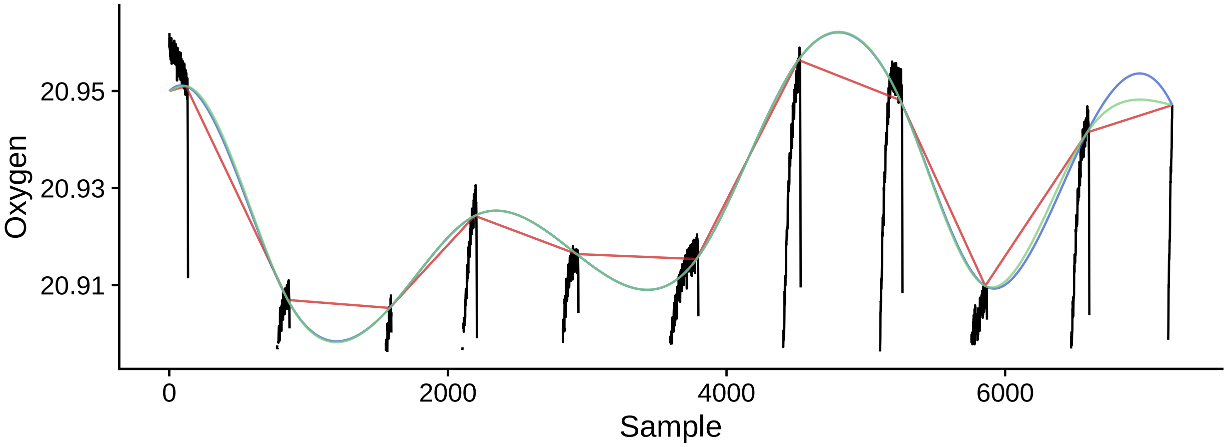
Drift correction of oxygen data

First, we will correct the drift in the previous plot in the oxygen data. We can use the function `baseline_corr` to do this. It requires a little more information that the previous functions, wanting the name of the dataframe, the column name with the channel markers, the column name to be drift corrected, the column name with the sample number, and the expected baseline value to correct the data to. Oxygen is typically correct to the value 20.95% as this is close to atmospheric concentrations.

```r
o2_corr <- baseline_corr(metabolic_data, "Marker", "Oxygen", "Sample", 20.950)
```

The function will attempt to correct the data using three different methods, 1) a simple linear interpolation, 2) a ffm (Forsythe, Malcolm, and Moller) cubic spline, and 3) a natural spline. The splines typically only vary at the beginning and the ends of the datasets, where there are few point to control the fit. Here we can see a zoomed in version (notice the y-axis values) of the original oxygen data with the 3 different fits. For this we can use the function `drift_plot`, which wants only 3 inputs, the corrected dataframe and the name of the start stop times contained in the baseline dataframe, “base_rect”, and the name of the sample, e.g. “Oxygen”.

```r
drift_plot(o2_corr, base_rect, "Oxygen")
```

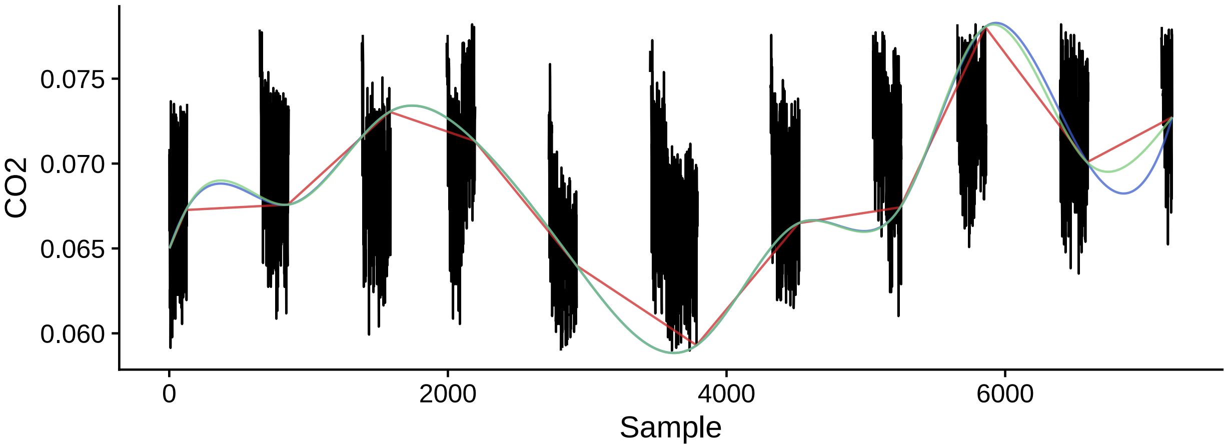


When inspecting the data, it appears the green line has the best fit so we will proceed with this data. In the future, I will implement a method of estimating variance based on the different fits so we can quantify the uncertainty in the different spline fits we correct the data to. We will repeat this with the carbon dioxide data.

```
co2_corr <- baseline_corr(metabolic_data, "Marker", "CO2", "Sample", 0.065)
```

Once again we can visualize the spline fits of the data, this time the carbon dioxide.

```
drift_plot(co2_corr, base_rect, "CO2")
```



Here we simply select the corrected columns from the two dirft corrected dataframes, and bind them to the end of the original metabolic data dataframe.

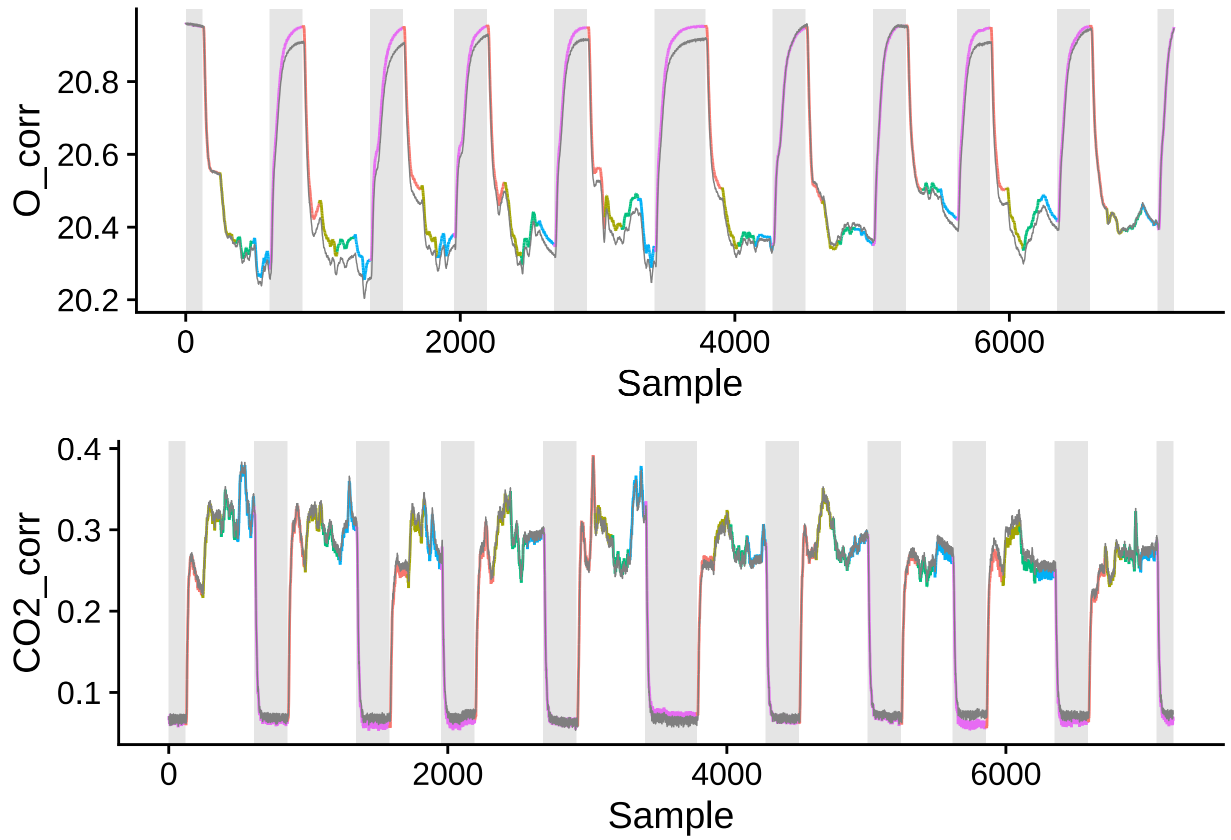
```
metabolic_data <- cbind(metabolic_data, O_corr = o2_corr$corr_fmm, CO2_corr = co2_corr$corr_fmm)
```

Now we can finally visualize our drift corrected data in comparison with the original data, and see how well it fits the data in between baseline measurements.

```
o2raw <- ggplot()+
            geom_rect(data=base_rect, aes(xmin = min, xmax = max, ymin = -Inf, ymax = Inf), fill = "grey
            geom_line(data=metabolic_data, aes(Sample, O_corr, group=1, color=as.factor(Marker)))+
            geom_line(data=metabolic_data, aes(Sample, Oxygen), color="grey50", size=0.25)+
            theme_cowplot()+
            theme(legend.position="none")

co2raw <- ggplot()+
            geom_rect(data=base_rect, aes(xmin = min, xmax = max, ymin = -Inf, ymax = Inf), fill = "grey
            geom_line(data=metabolic_data, aes(Sample, CO2_corr, group=1, color=as.factor(Marker)))+
            geom_line(data=metabolic_data, aes(Sample, CO2), color="grey50", size=0.25)+
            theme_cowplot()+
            theme(legend.position="none")

plot_grid(o2raw, co2raw, ncol=1)
```



Calculation of metabolic rate

```
metabolic_data <- equation_9_7(metabolic_data, 0.00065, 0.0162)
```