respR — \\

```
devtools::install_github("januarharianto/respR")
```

An R package for the analysis of respirometry data

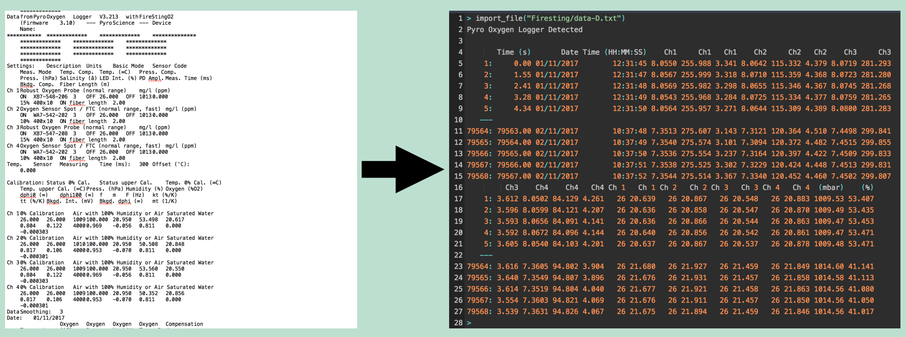
Januar Harianto[1], Nicholas **Carey**[2] & Maria **Byrne**[1] [1]The University of Sydney, NSW, Australia | [2]Scottish Association for Marine Science, Oban,

Summary. Respirometry is increasingly used to investigate physiological resilience in marine climate change studies, where ocean warming and acidification are expected to alter metabolism in ectotherms. Here, we present an R package, `respR`, designed to provide an *efficient* and *reproducible* workflow for the linear analysis of respirometry data.

Automatic file import.

The function **`import_file()`** uses string-matching algorithms to detect, format and tidy data outputs from popular machines such as **Firesting**, **PRESENS** and **Vernier**, so that data are imported *immediately*.

```
> import_file('Firesting/data-D.txt')
```

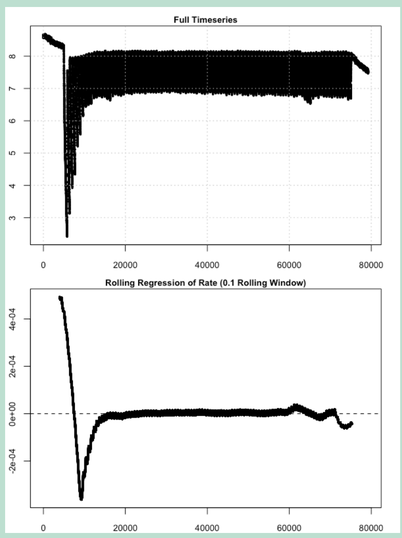


The function **`import_file()`** automatically converts a growing list of data file sources into data.frame objects that can be used for analyses in respR, or exported to open formats (e.g. csv) for long-term data storage.

Data & error visualisation.

Check for errors and visualise data with **`inspect()`**. Identify the source of common problems and fix issues rapidly.

```
> inspect(zeb_intermittent.rd)
```



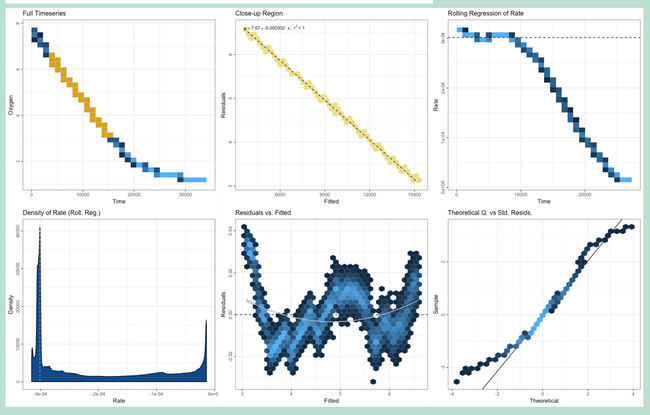

```
> inspect(zeb_intermittent.rd)
#> No issues detected while inspecting data frame.

#> # inspect # ------------------------------
#>               Time   O2
#> NA/NAN        pass pass
#> sequential    pass    -
#> duplicated    pass    -
#> evenly-spaced pass    -
>
```

Top: To minimise errors in data analysis, the function **`inspect()`** checks for common issues such as non-numeric data, non-sequential time, duplicates and uneven sampling intervals. **Left**: **`inspect()`** also plots the raw data and a rolling regression of the rate data to help identify stable regions.

Manual and automated analysis of rate metrics.

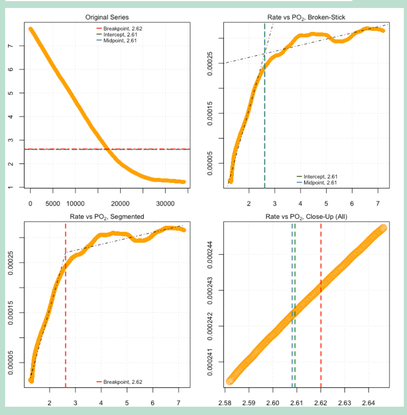
Automatically extract respirometry metrics such as maximum metabolic rate (MMR or $MO_{2,max}$), minimum rate (SMR, BMR or $MO_{2,min}$) and critical oxygen tension (P_{crit}) using **`auto_rate()`** or **`pcrit()`**. For more specific analyses, perform manual calculations with **`calc_rate()`**.

```
> auto_rate(squid.rd)
```



An example of the output plots generated automatically when **`auto_rate()`** is used. The plots show the subset of the data used to determine rate metrics, and present all the diagnostic plots required for the user to assess the accuracy of the methods used. In this instance, the function detected the "most linear" section of the data, highlighted in yellow.

```
> pcrit(squid.rd)
```



`respR` also supports automatic determination of critical oxygen tension (P_{crit}) using two well-known linear and non-linear methods. The function **`pcrit()`** will automatically generate visual plots of the break points detected. More methods are planned in future updates.

Strong support for unit conversions.

Use **`convert_DO()`** and **`convert_rate()`** to convert between weight- and volume-specific dissolved oxygen units and rates, with support for *temperature*, *salinity* and *pressure* inputs. We've implemented string matching algorithms such that your unit input strings are almost *always* recognised.

Oxygen units available in `respR`'s conversion functions. A total of 168 combinations of units are supported, with more planned in future versions of the package.

Parameter	Function	Units Supported
Oxygen (O_2)	**`convert_DO()`** **`convert_rate()`**	mg L^{-1}, µg L^{-1}, µmol L^{-1}, mL L^{-1}, µmol kg^{-1}, mmol kg^{-1}, mL kg-1, hPa, kPa, mmHg, inHg
Mass	**`convert_rate()`**	µg, mg, g, kg
Time	**`convert_rate()`**	second, minute, hour

Object-oriented, tidy programming.

Use *all*, or *any* function, with your current R workflow. `respR` works with `dplyr` pipes (`%>%`), so you can arrange, filter, summarise, select, subset, analyse and visualise data — all with one single chain of command.

```
urchins.rd %>%              # using the dataset,
  select(1, 15) %>%         # select cols. 1 and 15
  inspect()     %>%         # inspect the data, then
  auto_rate()   %>%         # detect most linear segment
  print()       %>%         # preview
  convert_rate("mg/l",
    "s", "mg/h/kg",
    0.6, 0.4)               # convert
```

This work is published in *Methods in Ecology and Evolution*. We acknowledge the financial support from the NSW Environmental Trust grant, the Endeavour Postdoctoral Fellowship and the Professor NGW & Mrs Ann Macintosh Memorial Scholarship.

@respr_pkg

Scan for more information:

Github Project Repository

Online User Documentation

Online Publication