respR – R Package Documentation

Januar Harianto *and* ***Nicholas Carey***

Contents

1 Introduction **5**

2 Why use `respR`? **7**
2.1 Other R respirometry packages 7
2.2 Linear detection 8
2.3 Reproducibility 8
2.4 Using other similar R packages 8
2.5 Using `respR` 9

USAGE 13

3 Installation **13**

4 Quick start **15**
4.1 Example data 15
4.2 Demonstration 15

METHODS 19

5 Aquatic respirometry **19**
5.1 Closed-chamber 19
5.2 Intermittent 19
5.3 Flow-through 20
5.4 Open-tank 20

EXAMPLES 23

6 Intermittent respirometry **23**
6.1 Example one 23
6.2 Example two 23

ARTICLES **27**

7 Articles **27**

Chapter 1

Introduction

The R package `respR` provides a structural, reproducible workflow for the processing and analysis of respirometry data. Although the focus of the package is on aquatic respirometry, `respR` is largely unitless, and so can process linear relationships in any time-series data. All analytical methods used are peer-reviewed and rigorously tested.

Use `respR` to:

- automatically import raw data from various oxygen sensing equipment;
- rapidly test data for issues before analysis;
- explore and visualise timeseries data;
- calculate linear segments of data manually or automatically;
- convert units of oxygen consumption; and
- export results quickly for reporting.

The goal of this guide is to walk you through the `respR` package to import, analyse and convert your data. Before we begin, you will need to install the package.

Chapter 2

Why use `respR`?

We have designed `respR` to be able to explore, process and analyse *any and all* aquatic respirometry data, independent of the system used to collect it or type of experiment. Because of the unitless nature of the majority of functions in the package, other respirometry data (e.g. aerobic), or other time series data examining other variables can be explored and analysed in `respR`.

The use of simple data structures (numeric vector and data frames) means there should be a low barrier to entry for anyone not completely new to R. The unitless nature of the data (requiring only paired values of numeric time-elapsed, and an oxygen amount in any unit) greatly reduces the inputs each function requires, and simplifies any analysis using `respR` in comparison to other packages (see A comparison of respR with other R packages). Once the user has imported and prepared their data to this form (see Importing your data), data analysis using respR is simple and intuitive.

2.1 Other R respirometry packages

Both rMR and FishResp centre around processing intermittent-flow, swim tunnel respirometry data with multiple, regularly-spaced replicates, particularly from Loligo Systems equipment. If this does not describe your experiment there is little point in considering these packages, as analysis of other respirometry experiments in them is challenging, if not impossible.

`respR` is more than capable of processing these intermittent flow experiments with a little forethought and data organisation (see here), and support for this will get better in the coming months. However, we would encourage users to explore these other packages and how they work; they are useful options and may contain functionality that suits your particular analyses or workflows better than `respR`.

2.2 Linear detection

Importantly, the packages mentioned above only allow manual selection of data regions over which to determine rates, such as over a specified time period. This is perfectly acceptable for many analyses, and `respR` has this functionality (although in a much more flexible implementation), but `respR` also has the `auto_rate()` function which identifies linear regions of respirometry data.

This powerful function allows identification of *most linear, minimum and maximum* rates in an **independent, objective, and statistically robust manner**. We would encourage the respirometry community to explore this objective method rather than rely on manual selection, which can leave investigators open to accusations of cherry picking and bias when reporting metabolic rates.

Another R package, `LoLinR`, can identify linear sections in time series data, athough using a fundamentally different method than `auto_rate()`. However, `LoLinR` is extremely computationally inefficient, taking literally hours to days to process typical respirometry datasets. It does appear to perform well, even after subsampling longer data to the shorter lengths the function can handle, but `auto_rate`appears to perform equally well and can process these data in seconds without modification (see here).

2.3 Reproducibility

Our other main objective with `respR` was to provide a solution for reporting analyses of respirometry data in an easily reproducible form. See Open science and reproducibility using respR for an example, but in summary, an entire `respR` analysis can be reported in only a few lines of code. Our careful selection of descriptive function names and input operators allow - we hope - this code to be readable and convey easy understanding about what is being done in the analysis, even without additional comments. Inclusion of a raw data file with this code would allow anyone to reproduce it easily, and scrutinise each stage. We hope this makes the job of investigators, reviewers and editors easier.

2.4 Using other similar R packages

We see only a few scenarios in which other R packages might be considered for analysing respirometry data:

- `LoLinR` - If your data is shorter than around 500 datapoints, and you would like to use a different linear detection method other than `auto_rate` in `respR`, `LoLinR` can achieve this. Note however, this package is not respirometry focussed. You would still need to utilise `respR` or other solutions to format and import respirometry data, apply background corrections, and convert the resulting slopes to particular units. If your data are longer than around 400-500 in length, `LoLinR` is not a practical option because of the time needed to process the data.

- `rMR`/`FishResp` - If you are doing intermittent-flow, swim tunnel experiments with regularly spaced replicates (particularly using Loligo Systems equipment) these are good options for analysising your data. These data are also able to be analysed in `respR` however: see link.
- Gas/Air respirometry - We don't know of any options in R to analyse air respirometry experiments, and at present `respR` is chiefly designed to analyse aquatic respirometry data. However, the great majority of the exploratory and analytical functions are unit agnostic. The only step requiring units is conversion of slopes to rates in the function `convert_rate()`. While we have not done this, we imagine air respirometry data could easily be explored and rates determined, and most experienced investigators in air respirometry would be able to convert these to units themselves. We plan to add this support, but as aquatic biologists we have no experience with these data. Please get in touch if you can help with this.

2.5 Using `respR`

In our opinion, you should use `respR` for most of your aquatic respirometry data.

We have built `respR` to accept any and all types of experiment or data source easily. We have not processed every data format or variation of experiment, and there may be circumstances we have not anticipated. So, if for whatever reason some part of `respR` does not work for you please do let us know, and we will work to accommodate it.

USAGE

Chapter 3

Installation

Use the devtools package to install a stable version of respR:

```r
# install devtools
install.packages("devtools")
# use devtools to install respR from GitHub
devtools::install_github("januarharianto/respR")
```

The developmental version is mostly stable, but contains bleeding-edge improvements. Do not use it unless you need new functionality that is still being tested in this branch. You can install the package using the code below:

```r
# install dev version
devtools::install_github("januarharianto/respR", ref = "develop")
```

Next, load the respR package into your workspace:

```r
library(respR)
```

Check out our Quick start guide if you are using respR for the first time.

Chapter 4

Quick start

`respR` has been designed to be simple to use, even for novice R users. Here we show you some examples of how data is analysed using various functions of the package.

4.1 Example data

Example data are available when `respR` is loaded: `urchins.rd()`, `intermittent.rd()`, `zeb_intermittent.rd()`, `sardine.rd()`, `squid.rd()`, `flowthrough.rd()`. Because lazy loading is implemented, the data can be called immediately. To view the list of available datasets and their descriptions, run the code:

```
data(package = "respR")
```

4.2 Demonstration

Let's analyse a relatively complex dataset quickly using `respR`.

METHODS

Chapter 5

Aquatic respirometry

There are four broad methodological approaches in aquatic respirometry: *closed-chamber*, *intermittent-flow*, *flow-through* and *open-tank*.

5.1 Closed-chamber

- Use `calc_rate()` and `auto_rate()` to measure closed-chamber respirometry.

In **closed-chamber** respirometry, O_2 decrease is measured within a hermetically sealed chamber of known volume, sometimes set within a closed loop to allow mixing of the environment within the chamber. Oxygen recordings may be continuous through use of an oxygen probe, periodic through withdrawing water or gas samples at set intervals, or a two-point measurement consisting of the initial and final concentrations. Metabolic rates are estimated from the O_2 timeseries by assuming a linear relationship between variables, and estimates of metabolic rate are straightforward in constant volume respirometry using the equation:

$$VO_2 = \dot{O}_2 V$$

where $\dot{O}_2$ is the slope of the regression that describes the rate of change in O_2 concentration over time, or in the case of a two-point measurement, the difference in O_2 concentration divided by time elapsed, and V is the volume of fluid in the container (Lighton 2008).

5.2 Intermittent

- Use `calc_rate()` to manually measure intermittent respirometry.

- Use `auto_rate()` if segments are manually identified prior to running the analysis.

In **intermittent-flow** respirometry, O_2 concentration is measured as described above, but periodically the chamber is flushed with new water or air, returning it to initial conditions, resealed, and the experiment repeated (Svendsen et al. 2016). This technique is essentially the same as closed respirometry, but with the ability to conduct replicates easily. Depending on the metabolic rate metric being investigated, final respiration rate can be calculated as the mean of the measures (e.g. Carey et al. 2016), or the lowest or highest rates recorded in any trial (e.g. Stoffels 2015).

5.3 Flow-through

- Use `calc_rate.ft()` to measure flow-through respirometry.

Flow-through respirometry involves a closed chamber, but with a regulated flow of air or water through it at a precisely determined rate. After equilibrium has been achieved, the oxygen concentration differential between the incurrent and excurrent channels, along with the flow rate, allows calculation of the oxygen extracted from the flow volume per unit time:

$$\dot{V}O_2 = (C_iO_2 - C_eO_2)FR$$

where $\dot{V}O_2$ is the rate of O_2 consumption over time, C_iO_2 and C_eO_2 are the incurrent and excurrent O_2 concentrations, and FR is the flow rate through the system (Lighton 2008).

5.4 Open-tank

- Use `calc.rate()` and `auto_rate()` to measure open-tank respirometry, and then use `adjust_rate()` to include oxygen flux adjustments.

A final method is **open-tank** respirometry, in which a tank or semi-enclosed area open to the atmosphere is used, but the input or mixing rate of oxygen from the surroundings has been quantified or found to be negligible relative to oxygen consumption of the specimens (Leclercq et al. 1999). It is seldom used, but for some applications it is a sufficient and practical methodology (Gamble et al. 2014). The common equation used for open respirometry is:

$$\dot{V}O_2 = \dot{O}_2V + \phi_d$$

where $\dot{O}_2V$ is the slope of the regression that relates O_2 concentration to time, V is the volume of the arena and ϕ_d is the oxygen flux as determined by Fick's Law (Leclercq et al. 1999).

EXAMPLES

Chapter 6

Intermittent respirometry

Some *significant* applications are demonstrated in this chapter.

6.1 Example one

6.2 Example two

ARTICLES

Chapter 7

Articles