Package 'FluxSeparator'

May 7, 2026

Title Separation of Diffusive and Ebullitive Fluxes

Version 1.0.1

Description Separates diffusive and ebullitive (bubble) fluxes from continuous concentration measurements using a running variance approach. Ebullitive events are identified when the running variance exceeds a user-set threshold. Diffusive fluxes are calculated via linear regression on the non-ebullitive portion of the data. See Sø et al. (2024) <doi:10.1029/2024JG008035> for details.

License MIT + file LICENSE

URL https://github.com/JonasStage/FluxSeparator, https://doi.org/10.1029/2024JG008035

BugReports https://github.com/JonasStage/FluxSeparator/issues

Encoding UTF-8

RoxygenNote 7.3.3

Depends R (>= 3.5)

Imports broom, dplyr, ggplot2, ggpubr, graphics, HMR, lubridate, magrittr, purrr, readr, rlang, stats, stringr, tidyr, TTR

Suggests knitr, rmarkdown, testthat (>= 3.0.0)

VignetteBuilder knitr

Config/testthat/edition 3

LazyData true

NeedsCompilation no

Author Jonas Stage Sø [aut, cre] (ORCID: <https://orcid.org/0000-0002-3222-3706>), Kaj Sand-Jensen [ctb] (ORCID: <https://orcid.org/0000-0003-2534-4638>), Theis Kragh [ctb] (ORCID: <https://orcid.org/0000-0002-9760-2571>)

Maintainer Jonas Stage Sø <Jonassoe@biology.sdu.dk>

Repository CRAN

Date/Publication 2026-03-29 16:50:03 UTC

Contents

FluxSeparator-package 2
diffusive_flux 2
DIY_sensor_data 5
ebullitive_flux 6
ppm_to_umol 9
read_CH4_files 10

Index **12**

`FluxSeparator-package` *FluxSeparator: Separation of Diffusive and Ebullitive Fluxes*

Description

Separates diffusive and ebullitive (bubble) fluxes from continuous concentration measurements using a running variance approach.

Author(s)

Jonas Stage Sø <Jonassoe@biology.sdu.dk>

`diffusive_flux` *Diffusive flux*

Description

Separates the diffusive and ebullitive fluxes, to calculate the diffusive flux, as a linear function of concentrations over time. This is done by firstly finding all events that are considered ebullitive (for more info see `ebullitive_flux`).

Several factors can be set to determine what is considered ebullitive events, remove observations before doing diffusive flux, number of observations used in the diffusive flux calculation, cutoffs if concentrations start too high and number of observations needed in the linear model.

Output data is converted to concentration change per hour.

Usage

```
diffusive_flux(
  data,
  concentration_values = "pred_CH4",
  runvar_cutoff = 0.5,
  remove_observations_prior = 200,
  number_of_observations_used = 400,
  show_plots = TRUE,
  IndexSpan = 30,
```

```
  cutoff_start_value = Inf,
  number_of_observations_required = 50,
  number_of_pumpcycles_in_plot = 50,
  smooth_data = FALSE,
  look_for_bubbles = TRUE,
  Hutchinson_Mosier_correction = FALSE,
  volume,
  area,
  min_obs_per_cycle = 100,
  time_gap_seconds = 30,
  runvar_window = 5,
  smooth_window = 10
)
```

Arguments

`data`	Your data frame. Must contain columns `datetime`, `PumpCycle`, `station`, and `tempC`.
`concentration_values`	Name of your variable representing the concentration.
`runvar_cutoff`	Cutoff of the running variance, which is used to determine if an increase in concentration is an ebullitive event. Lower values increases number of ebullitive events registered.
`remove_observations_prior`	Remove n number of observations before calculating the diffusive flux by a linear slope.
`number_of_observations_used`	Number of observations used to calculate the diffusive flux by a linear slope.
`show_plots`	Show plots which can assist in the determination of good fits for the model. A boolean variable which should be TRUE or FALSE.
`IndexSpan`	Number of observations which are included before and after an ebullitive event, to ensure the entire event is determined.
`cutoff_start_value`	Variable indicating what the maximum starting concentration can be. Defaults to `Inf` (no filtering).
`number_of_observations_required`	Number of observations required in each cycle for the function to compute a linear model on the data.
`number_of_pumpcycles_in_plot`	Number of cycles which are plotted. Used only if `show_plots = TRUE`.
`smooth_data`	Computes a running mean on the concentration data five times, to smoothen data if data is low bit resolution. See Sø et al. (2023) for more information.
`look_for_bubbles`	Can be used for the function to not consider ebullitive events. Can be useful when calculating diffusive CO2 flux.

`Hutchinson_Mosier_correction`
: Can be used to correct flux measurements based on the Hutchinson-Mosier correction (1981). However, fluxes are only calculated of three points. A boolean variable which should be TRUE or FALSE.

`volume`
: Volume of the chamber used for calculating fluxes (L). This is only needed if `Hutchinson_Mosier_correction = TRUE`.

`area`
: Surface area of the chamber used for calculating fluxes (m^2). This is only needed if `Hutchinson_Mosier_correction = TRUE`.

`min_obs_per_cycle`
: Minimum number of observations per pump cycle to include in analysis (default 100).

`time_gap_seconds`
: Time gap in seconds used to separate event groups when detecting bubbles (default 30).

`runvar_window`
: Window size for the running variance calculation (default 5).

`smooth_window`
: Window size for the running mean smoothing passes (default 10). Only used when `smooth_data = TRUE`.

Value

A data frame containing the following:

- station - The station from the input data
- PumpCycle - Cycle number
- datetime_start - Start time of the cycle
- datetime_end - End time of the cycle
- slope_concentration_hr - The diffusive flux per hour ($ppm\,h^{-1}$)
- slope_standard_error - The standard error of the flux
- n_obs_included_in_lm - Number of observations used to calculate the diffusive flux
- r2 - Variance explained by the linear model
- temp - Average temperature within the chamber (C)
- hmr_slope - Flux calculated using the HMR package ($ppm\,h^{-1}$). This is only returned if the method selected by the HMR package is either *Hutchinson-Mosier correction* or *No flux*, in case of *LR* used the function will return NA. Only given if the Hutchinson-Mosier correction is calculated.
- hmr_se - Standard error of the *hmr_slope*. This is only returned if the *hmr_slope* is calculated.
- hmr_pvalue - The p-value for the null hypothesis of zero flux. This is only returned if the *hmr_slope* is calculated.
- hmr_lower95 - The lower end-point of the 95\ for the flux. This is only returned if the *hmr_slope* is calculated.
- hmr_upper95 - The upper end-point of the 95\ for the flux. This is only returned if the *hmr_slope* is calculated.
- method - The method used by the HMR package to calculate flux. This is only returned if the *hmr_slope* is calculated.

When `show_plots = TRUE`, the returned data frame carries an attribute `"plots"` containing a list of **ggplot2** objects.

Author(s)

Jonas Stage Sø <Jonassoe@biology.sdu.dk>

References

Sø et al. (2024). Self-Made Equipment for Automatic Methane Diffusion and Ebullition Measurements From Aquatic Environments. doi:10.1029/2024JG008035.

Sø et al. (2023). Methane and carbon dioxide fluxes at high spatiotemporal resolution from a small temperate lake. doi:10.1016/j.scitotenv.2023.162895.

Hutchinson, G.L. and Mosier, A.R. (1981). Improved soil cover method for field measurement of nitrous oxide fluxes. Soil Science Society of America Journal, 45, pp. 311-316.

Pullens, J.W.M., Abalos, D., Petersen, S.O. and Pedersen, A.R. (2023). Identifying criteria for greenhouse gas flux estimation with automatic and manual chambers: A case study for N2O. European Journal of Soil Science, 74, e13340. doi:10.1111/ejss.13340.

See Also

ebullitive_flux, ppm_to_umol

Examples

```
library(FluxSeparator)

data(DIY_sensor_data)

DIY_sensor_data %>%
  diffusive_flux(cutoff_start_value = 450, show_plots = TRUE)
  # 450 would be good for CO2, while 5 could be good for CH4
```

DIY_sensor_data *Data from Lake Lyng*

Description

A subset of data from the paper Methane and carbon dioxide fluxes at high spatiotemporal resolution from a small temperate lake. One measurement containing only diffusive flux and one containing ebullitive events.

Usage

```
DIY_sensor_data
```

Format

A data frame with 2,201 rows and 12 variables:

datetime Date and time of measurement

RH Relative humidity (percent)

tempC Temperature in degrees Celsius

CH4smV Methane sensor voltage

K33_RH Relative humidity measured by the CO2 sensor

K33_Temp Temperature in degrees Celsius measured by the CO2 sensor

K33_CO2 CO2 concentration in ppm

SampleNumber Sample number in this pump cycle

PumpCycle Pump cycle which counts upwards after the chamber has been flushed

pred_CH4 Predicted methane concentration, calculated using the read_CH4_files function and calibration values

station Station name, needed for diffusive and ebullitive flux calculations

sensor Sensor identification number

Source

Sø et al. (2023). Methane and carbon dioxide fluxes at high spatiotemporal resolution from a small temperate lake. doi:10.1016/j.scitotenv.2023.162895

Examples

```
data(DIY_sensor_data)
head(DIY_sensor_data)
```

---

ebullitive_flux *Ebullitive flux*

Description

Separates the diffusive and ebullitive fluxes, to calculate the ebullitive flux, as the change in concentration from ebullitive events. This is done by computing a running variance; if the running variance exceeds a customizable cutoff value it is considered an ebullitive event.

Additional factors can be set to determine what is considered ebullitive events. Output data is converted to concentration change per hour.

Usage

```
ebullitive_flux(
  data,
  concentration_values = "pred_CH4",
  top_selection = "last",
  runvar_cutoff = 0.5,
  show_plots = TRUE,
  IndexSpan = 30,
  concentration_diffusion_cutoff = 1,
  number_of_pumpcycles_in_plot = 24,
  smooth_data = FALSE,
  min_obs_per_cycle = 100,
  time_gap_seconds = 30,
  runvar_window = 5,
  smooth_window = 10
)
```

Arguments

`data`	Your data frame. Must contain columns `datetime`, `PumpCycle`, `station`, and `tempC`.
`concentration_values`	Name of your variable representing the concentration.
`top_selection`	Can be set to `"last"` or `"max"` to either use the last or maximum concentration value in each ebullitive event.
`runvar_cutoff`	Cutoff of the running variance, which is used to determine if an increase in concentration is an ebullitive event. Lower values increase the number of ebullitive events registered.
`show_plots`	Show diagnostic plots. A logical; defaults to `TRUE`.
`IndexSpan`	Number of observations which are included before and after an ebullitive event, to ensure the entire event is captured.
`concentration_diffusion_cutoff`	Minimum concentration change that is considered an ebullitive event.
`number_of_pumpcycles_in_plot`	Number of cycles which are plotted. Used only if `show_plots = TRUE`.
`smooth_data`	Computes a running mean on the concentration data five times, to smooth data if data is low bit resolution. See Sø et al. (2023) for more information.
`min_obs_per_cycle`	Minimum number of observations per pump cycle required for processing (default 100).
`time_gap_seconds`	Time gap in seconds used to separate bubble event groups (default 30).
`runvar_window`	Window size for the running variance computation (default 5).
`smooth_window`	Window size for each running-mean smoothing pass (default 10). Only used when `smooth_data = TRUE`.

Value

A data frame containing the following:

- station
  The station column from the input data
- PumpCycle
  Cycle number
- datetime_start
  Start time of the cycle
- datetime_end
  End time of the cycle
- sum_bubbles_concentration
  The sum of the differences in concentration caused by bubbles
- n_bubbles
  Number of bubbles detected. Bear in mind that this function has difficulties detecting the number of bubbles if they are close to each other
- pumpcycle_duration_hr
  Length of the cycle duration in hours
- temp
  Average temperature within the chamber
- bubbles_per_time
  Amount of bubbles divided by the duration of the cycle in hours
- concentration_per_time
  Ebullitive flux, as the sum of concentration change divided by the duration in hours

When `show_plots = TRUE`, a `"plots"` attribute is attached to the result containing a list of **ggplot2** objects.

Author(s)

Jonas Stage Sø <Jonassoe@biology.sdu.dk>

References

Sø et al. (2024). Self-Made Equipment for Automatic Methane Diffusion and Ebullition Measurements From Aquatic Environments. doi:10.1029/2024JG008035.

Sø et al. (2023). Methane and carbon dioxide fluxes at high spatiotemporal resolution from a small temperate lake. doi:10.1016/j.scitotenv.2023.162895.

See Also

`diffusive_flux`, `ppm_to_umol`

Examples

```
library(FluxSeparator)

data(DIY_sensor_data)

DIY_sensor_data %>%
  ebullitive_flux()
```

---

ppm_to_umol *ppm_to_µmol*

Description

Conversion of $ppmV$ to $\mu mol\ m^{-2}\ h^{-1}$ using the ideal gas law.

Usage

```
ppm_to_umol(pressure, concentration, volume, temperature_C, area)
```

Arguments

pressure	Air pressure during measurement (Pa).
concentration	Concentration of the gas in ppm (µmol/mol).
volume	Volume of the chamber used for measuring in m^3.
temperature_C	Temperature in degrees Celsius in the chamber.
area	Surface area of the chamber used in m^2.

Value

A numeric vector of flux values in $\mu mol\ m^{-2}\ h^{-1}$.

Author(s)

Jonas Stage Sø <Jonassoe@biology.sdu.dk>

Examples

```
# Convert a single value
ppm_to_umol(pressure = 101325, concentration = 10,
            volume = 0.01, temperature_C = 20, area = 0.05)
```

read_CH4_files *read_CH4_files*

Description

A function to ease the import of data from DIY sensors, which reads a CSV file, calculates the absolute humidity, V0, RsR0, and the methane concentration.

Usage

```
read_CH4_files(data, files, join_model_coef = TRUE, model_coef_data = NULL)
```

Arguments

data	A data frame containing the path, sensor identification, and model coefficients for this specific sensor. Model coefficients can also be read in as a separate data frame and defined in the model_coef_data variable.
files	A vector supplying the path to the file being read.
join_model_coef	Boolean variable. Join data with dataframe model_coef_data to convert sensor voltage signal to methane concentration.
model_coef_data	Data frame consisting of the calibration values used to convert sensor voltage signal to methane concentration. Required when join_model_coef = TRUE.

Value

A data frame output including all the original values, with the exception of the model coefficients.

pred_CH4 Computed from the calibration model. The CH4 concentration calculated from the sensor resistance, expressed in ppm.

Author(s)

Jonas Stage Sø <Jonassoe@biology.sdu.dk>

References

Sø et al. (2023). Methane and carbon dioxide fluxes at high spatiotemporal resolution from a small temperate lake. doi:10.1016/j.scitotenv.2023.162895

Sø et al. (2024). Self-Made Equipment for Automatic Methane Diffusion and Ebullition Measurements From Aquatic Environments. doi:10.1029/2024JG008035

See Also

ebullitive_flux, diffusive_flux, ppm_to_umol

Examples

```
## Not run:
library(FluxSeparator)

# read in model coef
model_coef <- read_csv("model_coef.csv")

# path to DIY sensors files
path_to_files <- list.files(pattern = ".csv")

# create data frame for path, sensor and station.
data_path <- tibble(path = path_to_files,
                    sensor = c(1, 2, 3, 4),
                    station = c(1, 2, 4, 3))


# join with model_coef and calculate CH4 in ppm.
read_CH4_files(data_path, path,
               model_coef_data = model_coef)

#### Example using join_model_coef = FALSE ####

# join with model_coef.
joined_data_path <- left_join(data_path, model_coef, by = join_by(sensor))

# calculate CH4 in ppm.
read_CH4_files(joined_data_path,
               path,
               join_model_coef = FALSE)

## End(Not run)
```

Index

∗ **datasets**
DIY_sensor_data, 5

diffusive_flux, 2, *8*, *10*
DIY_sensor_data, 5

ebullitive_flux, *2*, *5*, 6, *10*

FluxSeparator *(*FluxSeparator-package*)*, 2
FluxSeparator-package, 2

ppm_to_umol, *5*, *8*, 9, *10*

read_CH4_files, 10