Package ‘gasfluxes’

March 30, 2017

Type Package
Title Greenhouse Gas Flux Calculation from Chamber Measurements
Version 0.2-1
Date 2017-03-17
Maintainer Roland Fuss <roland.fuss@thuenen.de>

BugReports https://bitbucket.org/ecoRoland/gasfluxes/issues

Description Functions for greenhouse gas flux calculation from chamber measurements.

License GPL (>= 2)

Depends R (>= 3.1.2)
Imports sfsmisc (>= 1.0), data.table (>= 1.9.4), MASS (>= 7.3),
     AICcmodavg (>= 2.0), stats, graphics, grDevices

Suggests testthat

RoxygenNote 6.0.1

NeedsCompilation no

Author Roland Fuss [aut, cre],
Asger R. Pedersen [cph] (for code copied from the not-exported
     HMR:::.HMR.fit1 function (version 0.3.1))

Repository CRAN

Date/Publication 2017-03-30 15:42:11 UTC

R topics documented:

  gasfluxes-package ........................................ 2
  agg.fluxes ............................................. 2
  erfc ................................................... 3
  fluxMeas ............................................... 4
  gasfluxes ............................................... 4
  HMR.fit ................................................. 7
  HMR.orig .............................................. 9
Description

Gasfluxes provides functions for fitting non-linear concentration-time models as well as convenience functions for checking data and combining different calculation methods.

Details

The wrapper function for convenient flux calculation is `gasfluxes`. Several concentration-time models are implemented:

- `HMR.orig`: The original implementation of HMR.
- `HMR.fit`: A new implementation of HMR using partially linear least-squares. This is recommended over `HMR.orig`.
- `NDFE.fit`: An implementation of the NDFE model using partially linear least-squares.
- `lin.fit`: A simple linear model.
- `rlin.fit`: A simple linear model fit using robust regression.

Description

Aggregate a time series of fluxes to a cumulative flux value.

Usage

```r
agg.fluxes(fluxes, datetimes, timeunit = "hours")
```

Arguments

- `fluxes`: flux values
- `datetimes`: datetime values (POSIXct or POSIXlt)
- `timeunit`: the unit of time (denominator of the flux unit), supported are the explicit units supported by `difftime`
**Details**

The function uses linear interpolation. The unit of the cumulative flux is \([\text{fluxes}] \times \text{timeunit}\). NA values are removed and values sorted according to time order. If less than two non-NA value pairs are provided, NA is returned for the cumulative flux.

**Value**

A one-row data.frame with columns

<table>
<thead>
<tr>
<th>flux</th>
<th>the cumulative flux</th>
</tr>
</thead>
<tbody>
<tr>
<td>from</td>
<td>the start of the cumulation period</td>
</tr>
<tr>
<td>to</td>
<td>the end of the cumulation period</td>
</tr>
</tbody>
</table>

The return value being a data.frame is useful, when the function is used for "split-apply-combine" type operations to calculate groupwise cumulated values, e.g., using package data.table.

**Examples**

```r
# Some random example data
datetimes <- Sys.time() + (1:20)/24*3600
set.seed(42)
fluxes <- rlnorm(20, 5)
agg.fluxes(fluxes, datetimes)
```

---

**Description**

This is the complementary error function.

**Usage**

```r
erfc(x)
```

**Arguments**

- \(x\) a numeric vector

**Value**

A numeric vector, i.e., the erfc values.
**fluxeMeas**  
*Data from chamber N2O flux measurements.*

**Description**
A dataset containing data from 1329 chamber N2O flux measurements.

**Format**
A data.table with 5300 rows and 5 variables:
- serie: ID of flux measurement
- V: Volume (normalized by area, i.e., the height in m)
- A: Area (always 1)
- time: closing time in h
- C: N2O concentration in mg N / m^3

**Source**
own data (anonymized by not including site and treatment information)

---

**gasfluxes**  
*Flux calculation*

**Description**
A wrapper function for convenient flux calculation.

**Usage**
```r
.C = "C", methods = c("linear", "robust linear", "HMR", "NDFE"),  
select = NULL, k_HMR = log(1.5), k_NDFE = log(0.01), verbose = TRUE,  
plot = TRUE, ...)
```

**Arguments**
- **dat**  
a data.frame or data.table with data from flux measurements.
- **.id**  
character vector specifying the columns to be used as ID, multiple ID columns are possible.
- **.V**  
character specifying the column containing chamber volume values.
- **.A**  
character specifying the column containing chamber area values.
- **.times**  
character specifying the column containing chamber closing time values.
gasfluxes

.C character specifying the column containing concentration values.
docterms character: which methods to use for flux estimation. See details for available methods.
select character: specify a ruleset for selection of the final flux value, use NULL for no selection.

(k_hmr starting value for HMR.fit.

(k_NDFE starting value for NDFE.fit.

verbose logical: print progress messages?

plot create plots if TRUE (the default). The IDs are used as file names and should thus not include characters that are not allowed in file names, such as / or =. A directory "pics" is created in the working directory if it doesn’t exist. The plots are only intended to facilitate quick checking, not for publication quality graphs.

... further parameters

Details

Available methods are

"linear": lin.fit

"robust linear": rlin.fit

"HMR": HMR.fit

"original HMR": HMR.orig

"NDFE": NDFE.fit

Specifying other methods results in an error.
The default starting values for "HMR" and "NDFE", \( k = \log(\kappa) \) and \( k = \log(\tau) \), resp., assume that time is in hours. If you use a different time unit, you should adjust them accordingly. Note that nls is used internally by these functions and thus they should not be used with artificial "zero-residual" data.

Available selection algorithms currently are

"RF2011" The algorithm used, e.g., in Leiber-Sauheitl 2014 (doi:10.5194/bg-11-749-2014). This overwrites the methods parameter. The factor guarding against degenerate HMR fits can be set via the ellipsis as gfactor. Default is gfactor = 4.

"RF2011new" The same rules as "RF2011", but using the improved fitting function for HMR, which results in larger SE and p-values. Thus, it is less likely to select the HMR result. This selection algorithm might need further testing and validation.

Other selection algorithms could be implemented, but selection can always be done as a postprocessing step. E.g., if many data points are available for each flux measurement it is probably most sensible to use AICc.

The input data.frame or data.table should be in the following format:

<table>
<thead>
<tr>
<th>serie</th>
<th>V</th>
<th>A</th>
<th>time</th>
<th>C</th>
</tr>
</thead>
<tbody>
<tr>
<td>1:</td>
<td>ID1 0.522625 1 0.0000000 0.3317823</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
However, more than one ID column are possible. E.g., the first ID column could be the plot and a second ID column could be the date. Keep in mind that the combination of IDs must be a unique identifier for each flux measurement.

Units of the output depend on input units. It’s recommended to use \([V] = m^3\), \([A] = m^2\), \([\text{time}] = h\), \([C] = [\text{mass or mol}]/m^3\), which results in \([f_0] = [\text{mass or mol}]/m^2/h\). Since all algorithms use V/A, A can be input as 1 and V as the chamber height.

Value

A data.table with the results of the flux calculation. See the documentation of the fitting functions for details. If a selection algorithm has been specified, the last columns are the selected flux estimate, the corresponding standard error and p-value and the method with which the selected flux was estimated.

Examples

```r
# Not run:
# compare result of original HMR with plinear HMR
data(fluxMeas)
res <- gasfluxes(fluxMeas[1:400,],
  .times = "time", .C = "C",
  methods = c("HMR", "original HMR"), verbose = TRUE)

# number of successful fits
res[, sum(!is.na(HMR.kappa))]
res[, sum(!is.na(original.HMR.kappa))]

# Do the results differ?
plot(res[["HMR.f0"]], res[["original.HMR.f0"]])
abline(0, 1)

res <- gasfluxes(fluxMeas,
  .times = "time", .C = "C",
  methods = "HMR", verbose = TRUE)
# Error: time not sorted in flux ID ID556.
# Investigate the problem:
fluxMeas[serie %in% c("ID555", "ID556", "ID557")]
# serie     V     A       time      C
# 1: ID555 0.551625 1 0.0000000 0.3884388
# 2: ID555 0.551625 1 0.3333333 0.4125270
# 3: ID555 0.551625 1 0.6666667 0.3714207
# 4: ID555 0.551625 1 1.0000000 0.3735092
# 5: ID556 0.524250 1 0.0000000 0.3638239
```
Description

Fit the HMR model using the Golub-Pereyra algorithm for partially linear least-squares models.

Usage

\texttt{HMR.fit(t, C, A = 1, V, serie = "", k = log(1.5), verbose = TRUE, plot = FALSE, ...)}

Arguments

- \texttt{t} 
  time values (usually in hours)
- \texttt{C} 
  concentration values
- \texttt{A} 
  area covered by the chamber
- \texttt{V} 
  effective volume of the chamber
- \texttt{serie} 
  id of the flux measurement
- \texttt{k} 
  starting value for nls function
- \texttt{verbose} 
  logical, TRUE prints message after each flux calculation
- \texttt{plot} 
  logical, mainly intended for use in \texttt{gasfluxes}
- \texttt{...} 
  further parameters, currently none

Details

The HMR model (Pedersen et al., 2010) is \( C(t) = \phi + f_0 \frac{e^{-\kappa t}}{\kappa V/A} \). To ensure the lower bound \( \kappa > 0 \), the substitution \( \kappa = e^k \) is used. The resulting reparameterized model is then fit using \texttt{nls} with \texttt{algorithm = "plinear"}. This is computationally more efficient than the manual implementation in the HMR package and results in almost identical flux values. Flux standard errors and p-values
differ strongly from those reported by the HMR package <= version 0.3.1, but are equal to those reported by later versions.

The default starting value \( k = \log(\kappa) \) assumes that time is in hours. If you use a different time unit, you should adjust it accordingly.

There have been demands to return the initial concentration as predicted by the model as this is useful for checking plausibility. However, this can be easily calculated from the parameters and the equation of the model by setting \( t = 0 \), i.e., \( C_0 = \phi + f_0. \)

Note that nls is used internally and thus this function should not be used with artificial "zero-residual" data.

**Value**

A list of

- \( f0 \): flux estimate
- \( f0.se \): standard error of flux estimate
- \( f0.p \): p-value of flux estimate
- \( \kappa, \phi \): other parameters of the HMR model
- \( \text{AIC} \): Akaike information criterion
- \( \text{AICc} \): Akaike information criterion with small sample correction
- \( \text{RSE} \): residual standard error (sigma from summary.nls)
- \( \text{diagnostics} \): error or warning messages

**References**


**Examples**

```r
# a single fit
t <- c(0, 1/3, 2/3, 1)
C <- c(320, 341, 352, 359)
print(fit <- HMR.fit(t, C, 1, 0.3, "a"))
plot(C ~ t)
curve((fit$phi + fit$f0 * exp(-fit$kappa * x))/(-fit$kappa*0.3)),
     from = 0, to = 1, add = TRUE)
# compare with fitting function from HMR package 0.3.1
gasfluxes:::HMR.fit1(t, C, 1, 0.3, "a",
ngrid = 1000, LR.always = FALSE, FollowHMR = TRUE,
JGP = FALSE, PS = FALSE, PHMR = FALSE, npred = 500,
xtxt = "", ytxt = "", pctxt = "",
MSE.zero = 10 * max(.Machine$double.eps, .Machine$double.neg.eps),
bracketing.tol = 1e-07, bracketing.maxiter = 1000)[2:4]
```

## Not run:

#a dataset of 1329 chamber N2O flux measurements
data(fluxMeas)
fluxMeas[, n := length(time), by=serie]
print(fluxMeas)
fluxes <- fluxMeas[n > 3, HMR.fit(time, C, A, V, serie), by=serie]
print(fluxes)
plot(f0.se ~ f0, data = fluxes)
#one very large f0.se value (and several infinite ones not shown in the plot)
fluxes[is.finite(f0.se),][which.max(f0.se),]
plot(C~time, data=fluxMeas[serie==''ID940''],)
print(tmp <- fluxes[is.finite(f0.se),][which.max(f0.se),])
curve((tmp[, phi] + tmp[, f0] * exp(-tmp[, kappa] * x))/
 (-tmp[, kappa]*fluxMeas[serie==''ID940'', V[1]]/ 
 fluxMeas[serie==''ID940'',A[1]])},
 from = 0, to = 1, add = TRUE)
plot(f0.se ~ f0, data = fluxes[f0.se < 1e4,], pch = 16)
boxplot(fluxes[f0.se < 1e4, sqrt(f0.se)])

## End(Not run)

---

**HMR.orig**

**HMR fit using the Pedersen fitting algorithm**

**Description**

Fit the HMR model using the algorithm from the HMR package.

**Usage**

```r
HMR.orig(t, C, A = 1, V, serie = '', verbose = TRUE, ngrid = 1000,
         plot = FALSE, ...)
```

**Arguments**

- **t**: time values (usually in hours)
- **C**: concentration values
- **A**: area covered by the chamber
- **V**: effective volume of the chamber
- **serie**: id of the flux measurement
- **verbose**: logical, TRUE prints message after each flux calculation
- **ngrid**: see the HMR documentation
- **plot**: logical, mainly intended for use in `gasfluxes`
- **...**: further parameters, currently none
Details

The HMR model (Pedersen et al., 2010) is \( C(t) = \phi + f_0 e^{-\frac{t}{\kappa}} \). The algorithm from the HMR package version 0.3.1 is used for fitting. Note that this is very inefficient and standard errors and p-values are over-estimated. \texttt{HMR.fit} is recommended instead and this function is only provided to be able to reproduce results obtained with older versions of the HMR package.

Value

A list of

- \( f_0 \): flux estimate
- \( f_0.se \): standard error of flux estimate
- \( f_0.p \): p-value of flux estimate
- \( \kappa, \phi \): other parameters of the HMR model
- \( \text{AIC} \): Akaike information criterion
- \( \text{AICc} \): Akaike information criterion with small sample correction
- \( \text{diagnostics} \): error or warning messages

Author(s)

Asger R. Pedersen for code copied from the not-exported HMR::HMR.fit1 function, Roland Fuss

References


Examples

```r
# a single fit
fit <- HMR.orig(t, C, 1, 0.3, "a")
plot(C ~ t)
curve((fit$phi + fit$f0 * exp(-fit$kappa * x)) / (-fit$kappa * 0.3),
     from = 0, to = 1, add = TRUE)
```

lin.fit  

Linear concentration - time model

Description

Fit a linear model to concentration - time data.
**lin.fit**

**Usage**

```r
lin.fit(t, C, A = 1, V, serie = "", verbose = TRUE, plot = FALSE, ...)
```

**Arguments**

- `t`: time values (usually in hours)
- `C`: concentration values
- `A`: area covered by the chamber
- `V`: effective volume of the chamber
- `serie`: id of the flux measurement
- `verbose`: logical, TRUE prints message after each flux calculation
- `plot`: logical, mainly intended for use in `gasfluxes`
- `...`: further parameters, currently none

**Details**

This is basically a wrapper of R’s OLS fitting facilities. For now `lm` (and methods for objects of class "lm") is used, but this may change to more efficient alternatives in later versions.

**Value**

A list of

- `f0`: flux estimate
- `f0.se`: standard error of flux estimate
- `f0.p`: p-value of flux estimate
- `C0`: estimated concentration at t = 0 (intercept)
- `AIC`: Akaike information criterion
- `AICc`: Akaike information criterion with small sample correction
- `RSE`: residual standard error (sigma from summary.nls)
- `diagnostics`: error or warning messages

**Examples**

```r
# a single fit
t <- c(0, 1/3, 2/3, 1)
C <- c(320, 341, 352, 359)
print(fit <- lin.fit(t, C, 1, 0.3, "a"))
plot(C ~ t)
curve((fit$f0/0.3 * x + fit$c0), from = 0, to = 1, add = TRUE)
```
Description

Fit the non-steady-state diffusive flux estimator model using the Golub-Pereyra algorithm for partially linear least-squares models.

Usage

NDFE.fit(t, C, A = 1, V, serie = "", k = log(0.01), verbose = TRUE, plot = FALSE, ...)

Arguments

t time values (usually in hours)
C concentration values
A area covered by the chamber
V effective volume of the chamber
serie id of the flux measurement
k starting value for nls function
verbose logical, TRUE prints message after each flux calculation
plot logical, mainly intended for use in gasfluxes
...

Details

The NDFE model (Livingston et al., 2006) is

$$C(t) = C_0 + f_0 \tau^\frac{A}{V} \left[ \frac{2}{\sqrt{\pi \tau}} \sqrt{t/\tau} + e^{t/\tau} \text{erfc}(\sqrt{t/\tau}) - 1 \right].$$

To ensure the lower bound \( \tau > 0 \), the substitution \( \tau = e^k \) is used. The resulting reparameterized model is then fit using nls with algorithm = "plinear".

Note that according to the reference the model is not valid for negative fluxes. Warning: This function does not check if fluxes are positive. It’s left to the user to handle negative fluxes.

The default starting value \( k = \log(\tau) \) assumes that time is in hours. If you use a different time unit, you should adjust it accordingly.

Note that nls is used internally and thus this function should not be used with artificial "zero-residual" data.

Value

A list of

f0 flux estimate
f0.se standard error of flux estimate
rlin.fit  

f0.p  p-value of flux estimate
C0, tau other parameters of the NDFE model
AIC Akaike information criterion
AICc Akaike information criterion with small sample correction
RSE residual standard error (sigma from summary.nls)
diagnostics error or warning messages

References

Examples
# a single fit
t <- c(0, 1/3, 2/3, 1)
C <- c(320, 340, 355, 362)
print(fit <- NDFE.fit(t, C, 1, 0.3, "a"))
plot(C - t)
curve((fit$C0+fit$f0*fit$tau*x1/(2/sqrt(pi)*sqrt(x/fit$tau)) +
      exp(x/fit$tau)*erfc(sqrt(x/fit$tau)-1)),
      from = 0, to = 1, add = TRUE)
# note that the flux estimate is very uncertain because
# there are no data points in the region of high curvature

rlin.fit  \textit{Robust linear concentration - time model}

Description
Fit a linear model to concentration - time data using robust methods.

Usage
rlin.fit(t, C, A = 1, V, serie = "", verbose = TRUE, plot = FALSE, ...)

Arguments
\begin{itemize}
\item \texttt{t} time values (usually in hours)
\item \texttt{C} concentration values
\item \texttt{A} area covered by the chamber
\item \texttt{V} effective volume of the chamber
\item \texttt{serie} id of the flux measurement
\item \texttt{verbose} logical, TRUE prints message after each flux calculation
\item \texttt{plot} logical, mainly intended for use in \texttt{gasfluxes}
\item \ldots further parameters, currently none
\end{itemize}
Details
This is basically a wrapper of `rlm` using the Huber M estimator. This function never weights the first or last time point with zero with very few data points. However, there might exist "better" robust regression methods for flux estimation.

Value
A list of

- `f0`        flux estimate
- `f0.se`     standard error of flux estimate
- `f0.p`      p-value of flux estimate
- `C0`        estimated concentration at t = 0 (intercept)
- `weights`   robustness weights
- `diagnostics` error or warning messages

Examples

```r
# a single fit
t <- c(0, 1/3, 2/3, 1)
C <- c(320, 330, 315, 351)
print(fit <- rlin.fit(t, C, 1, 0.3, "a"))
plot(C ~ t)
curve((fit$f0/0.3 * x + fit$C0), from = 0, to = 1, add = TRUE)
```
Index

agg.fluxes, 2
erfc, 3
fluxMeas, 4
gasfluxes, 2, 4, 7, 9, 11–13
gasfluxes-package, 2
HMR.fit, 2, 5, 7, 10
HMR.orig, 2, 5, 9
lin.fit, 2, 5, 10
lm, 11
NDFE.fit, 2, 5, 12
nls, 7, 12
rlin.fit, 2, 5, 13
rlm, 14