Package ‘gauseR’

November 28, 2021

Title Lotka-Volterra Models for Gause’s ‘Struggle for Existence’

Version 1.1

Description A collection of tools and data for analyzing the Gause microcosm experiments, and for fitting Lotka-Volterra models to time series data. Includes methods for fitting single-species logistic growth, and multi-species interaction models, e.g. of competition, predator/prey relationships, or mutualism. See documentation for individual functions for examples. In general, see the lv_optim() function for examples of how to fit parameter values in multi-species systems. Note that the general methods applied here, as well as the form of the differential equations that we use, are described in detail in the Quantitative Ecology textbook by Lehman et al., available at <http://hdl.handle.net/11299/204551>, and in Lina K. Mühlbauer, Maximilienne Schulze, W. Stanley Harpole, and Adam T. Clark. ‘gauseR’: Simple methods for fitting Lotka-Volterra models describing Gause’s ‘Struggle for Existence’ in the journal Ecology and Evolution.

Imports deSolve, stats, graphics

Suggests knitr, rmarkdown

VignetteBuilder knitr

License GPL-3

Encoding UTF-8

LazyData true

RoxygenNote 7.1.1

NeedsCompilation no

Author Adam Clark [aut, cre] (<https://orcid.org/0000-0002-8843-3278>), Lina Mühlbauer [aut], Maximilienne Schulze [aut]

Maintainer Adam Clark <adam.tclark@gmail.com>

Repository CRAN

Date/Publication 2021-11-28 12:20:02 UTC

R topics documented:

gauseR ................................................................. 3

1
gauseR

gauseR: Simple methods for fitting Lotka-Volterra models describing Gause’s "Struggle for Existence"

Description

A collection of tools and data for analyzing the Gause microcosm experiments, and for fitting Lotka-Volterra models to time series data. Includes methods for fitting single-species logistic growth, and multi-species interaction models, e.g., of competition, predator/prey relationships, or mutualism. See documentation for individual functions for examples. In general, see the `lv_optim()` function for examples of how to fit parameter values in multi-species systems.

Authors

Adam Clark, Lina Muehlbauer, and Maximilienne Schulze.

Applications

Note that the general methods applied here, as well as the form of the differential equations that we use, are described in detail in the Quantitative Ecology textbook by Lehman et al., cited below. Using the default functions, species dynamics therefore follow the form:

\[
dni/dt = ni \cdot (ri + aii \cdot ni + \text{sum}_j(aij \cdot nj))
\]

Source


Examples

# primary wrapper function
?gause_wrapper  # automatically runs functions to get starting values and fit parameter values using simulated ODE dynamics.

# individual functions
?get_lag  # generate time-lagged variables for estimating per-capita growth
?percap_growth  # generate estimates of per-capita growth rates for species
?get_logistic  # logistic growth function
?lv_interaction  # function for simulating Lotka-Voterra ODE models
gause_1931_AmN_f01

?lv_interaction_log # a version of the lv_interaction computed in log abundance space, 
# which typically works better for optimization
?lv_optim # methods for fitting complex n-species models
?test_goodness_of_fit # tests goodness of fit of model results, with an R2-like statistic
?ode_prediction # generic function for simulating time series, 
# to be used with other optimizers

**gause_1931_AmN_f01**  
*Growth of population of the flour beetle Tribolium confusum in 16 and 64 grams of flour*

**Description**

A dataset containing the abundance of Tribolium confusum, grown in monoculture with different quantities of food (flour). Gause’s goal was to determine the influence of ecological factors on population growth.

**Usage**

`gause_1931_AmN_f01`

**Format**

A data frame with 18 rows and 6 variables:

- **Paper**  Paper from which data are drawn
- **Figure**  Figure number in paper
- **Species**  Name of Species: Tribolium confusum
- **Time**  Day of experiment
- **Individuals**  Number of Individuals
- **Treatment**  Treatments: 16 and 64g flour per starting beetle pair

**Source**

gause_1931_AmN_f02

The influence of quantity of food on the asymptotic population of Tribolium confusum.

Description
A dataset containing the population size of Tribolium confusum at the upper asymptote, as a characteristic for the population being in equilibrium, under different quantities of food (flour). Gause’s goal was to determine the influence of ecological factors on population growth.

Usage

gause_1931_AmN_f02

Format
A data frame with 4 rows and 5 variables:

- **Paper** Paper from which data are drawn
- **Figure** Figure number in paper
- **Species** Name of Species: Tribolium confusum
- **Individuals** Number of Individuals at population equilibrium
- **Treatment** Treatments: 16, 32, 64 and 128g flour per starting beetle pair

Source

gause_1931_AmN_f03

The influence of temperature on the asymptotic population of Moina macrocopa.

Description
A dataset containing the population size of Moina macrocopa at the upper asymptote, as a characteristic for the population being in equilibrium, under different temperatures. Gause’s goal was to determine the influence of ecological factors on population growth.

Usage

gause_1931_AmN_f03
Format

A data frame with 3 rows and 5 variables:

- **Paper**: Paper from which data are drawn
- **Figure**: Figure number in paper
- **Species**: Name of Species: Tribolium confusum
- **Individuals**: Number of Individuals at population equilibrium
- **Treatment**: Treatments: 20, 25 and 35 degrees Celsius

Source


Description

A dataset containing the abundance of Saccharomyces cerevisiae, grown in different temperatures (5.7 to 41 degrees C). The Volume is measured in "Amount of yeast", which refers to a standardized index, based on the alcohol production per Unit of Yeast.

Usage

gause_1932_QR_t05

Format

A data frame with 102 rows and 7 variables:

- **Paper**: Paper from which data are drawn
- **Table**: Table number in paper
- **Experiment**: Number of series: 1, 2 or 3
- **Species**: Name of Species: Saccharomyces cerevisiae
- **Time**: Hour of experiment
- **Volume**: "Amount of Yeast" of Species
- **Treatment**: Treatments: Temperatures (5.7 to 41 degrees C)

Source

Raw data on the abundances and volumes of Saccharomyces cerevisiae and Schizosaccharomyces kephir

Description

A dataset containing the abundances (number of cells) and volumes of Saccharomyces cerevisiae and Schizosaccharomyces kephir grown in monoculture and in mixture. The data for two experiments with different time periods is reported.

Usage

gause_1934_book_app_t01

Format

A data frame with 60 rows and 10 variables:

- **Paper**  Paper from which data are drawn
- **Table**  Table number in paper
- **Experiment**  Number of experiment: 1 or 2
- **Species**  Name of Species: Saccharomyces cerevisiae and Schizosaccharomyces kephir
- **Time**  Day of experiment
- **Volume_Species**  Volume of yeast species in the mixture, estimated from counted cells
- **Squares**  Number of Squares counted
- **Individuals_Square**  Average Number of cells per Square
- **Volume_Total**  Total volume of yeast
- **Treatment**  Treatments: Monoculture and Mixture

Source

Alcohol production of Saccharomyces cerevisiae and Schizosaccharomyces kephir

Description
A dataset containing the alcohol production of Saccharomyces cerevisiae and Schizosaccharomyces kephir cultivated under anaerobic and aerobic conditions, in percent and per unit of yeast volume.

Usage
gause_1934_book_app_t02

Format
A data frame with 28 rows and 7 variables:

Paper  Paper from which data are drawn
Table  Table number in paper
Species   Name of Species: Saccharomyces cerevisiae, Schizosaccharomyces kephir
Time    Hour of experiment
Alcohol  Alcohol production in percent
Alcohol.Yeast.Volume  Alcohol production per unit of yeast volume
Treatment  Treatments: Anaerobic and Aerobic

Source

Raw data of Paramecium caudatum and Paramecium aurelia grown in Monoculture and Mixture

Description
A dataset containing the mean abundances of Paramecium caudatum and Paramecium aurelia, grown in mixture and monoculture. This dataset contains the mean abundances of three experiments. Note, that for day 20, single values are missing and only the mean is reported.

Usage
gause_1934_book_app_t03
Format

A data frame with 104 rows and 6 variables:

- **Paper**: Paper from which data are drawn
- **Table**: Table number in paper
- **Species**: Name of Species: *Paramecium caudatum* and *Paramecium aurelia*
- **Time**: Day of experiment
- **Individuals**: Number of Individuals
- **Treatment**: Treatments: Monoculture and Mixture

Source


---

Description

A dataset containing the abundance of *Paramecium caudatum* and *Paramecium aurelia*, grown in monoculture and in mixture on buffered medium with high wild bacteria concentration ("one loop" medium) and low wild bacteria concentration ("half loop" medium). The number of individuals is reported for monoculture and the number of individuals of the species in the mixture.

Usage

gause_1934_book_app_t04

Format

A data frame with 68 rows and 7 variables:

- **Paper**: Paper from which data are drawn
- **Table**: Table number in paper
- **Species**: Name of Species: *Paramecium caudatum* and *Paramecium aurelia*
- **Time**: Day of experiment
- **Individuals**: Number of Individuals
- **Individuals_Mixture**: Number of Individuals per Species in Mixture
- **Treatment**: Treatments: One Loop and Half Loop

Source

Raw data of Stylonychia pustulata in monoculture and mixture with and Paramecium aurelia and P. caudatum

Description

A dataset containing the abundances of Stylonychia pustulata grown in monoculture and mixture with P. aurelia and P. caudatum and the abundances of P. aurelia and P. caudatum in this mixtures on the medium of Osterhout. This dataset contains the raw data of three different experiments: Stylonychia pustulata grown in monoculture, in mixture with P.aurelia and in mixture wit P.caudatum. The abundances of three to five cultures per experiment and the calculated mean abundances of the experiments are reported. Note, that for day 20 - 25, single values are missing and only the mean is reported.

Usage

gause_1934_book_app_t05

Format

A data frame with 575 rows and 8 variables:

Paper  Paper from which data are drawn
Table   Table number in paper
Time    Day of experiment
Experiment Number of experiment: 1,2 or 3
Culture  Number of culture: 1,2,3,4,5 or Mean
Species  Name of Species: Stylonychia pustulata, Paramecium caudatum and Paramecium aurelia
Individuals Number of Individuals
Treatment  Treatments: Monoculture and Mixture P.aurelia and Mixture P.caudatum

Source

Growth of Paramecium caudatum

Description
A dataset containing the growth in abundance of Paramecium caudatum over six days.

Usage

Format
A data frame with 8 rows and 5 variables:

- **Paper**: Paper from which data are drawn
- **Figure**: Figure number in paper
- **Species**: Name of Species: Paramecium caudatum
- **Time**: Day of experiment
- **Individuals**: Number of Individuals

Source

Growth of Saccharomyces cerevisiae

Description
A dataset containing the growth in volume of Saccharomyces cerevisiae under anaerobic conditions. The Volume is measured in "Amount of yeast", which refers to a standardized index, based on the alcohol production per Unit of Yeast.

Usage

Format
A data frame with 9 rows and 5 variables:

- **Paper**: Paper from which data are drawn
- **Figure**: Figure number in paper
- **Species**: Name of Species: Saccharomyces cerevisiae
- **Time**: Hour of experiment
- **Volume**: "Amount of Yeast" of Species
**Source**


**Description**

A dataset containing the abundance of Saccharomyces cerevisiae cultivated in monoculture with a medium change in different time periods under anaerobic conditions. The Number of Individuals is measured in Number of cells per 1/250 mm3.

**Usage**

**Format**

A data frame with 29 rows and 6 variables:

- **Paper**  Paper from which data are drawn
- **Figure** Figure number in paper
- **Species** Name of Species: Saccharomyces cerevisiae
- **Time** Hour of experiment
- **Individuals** Number of cells per 1/250 mm3
- **Treatment** Treatments: medium change every 3, 12, 24 hours and control

**Source**


**Description**

A dataset containing yeast volume and alcohol concentration for two replicates of S. cerevisiae grown in monoculture.

**Usage**

gause_1934_book_f11
Format

A data frame with 11 rows and 7 variables:

- **Paper**  Paper from which data are drawn
- **Figure**  Figure number in paper
- **Species**  Species name
- **Time**  Experiment time in hours
- **Alcohol_Percent**  Alcohol concentration, in percent
- **Yeast_Volume**  Yeast volume, listed in papers as ‘amount’
- **Treatment**  Experiment replicate, under two ‘somewhat different’ growth medium concentrations

Source


---

*A dataset containing the growth in volume of* Saccharomyces cerevisiae *cultivated under anaerobic conditions with added alcohol. Gause measured the effect of alcohol on reaching a saturated population. The saturation is measured in percent of the saturated population grown without additional alcohol.*

Usage

gause_1934_book_f12

Format

A data frame with 6 rows and 5 variables:

- **Paper**  Paper from which data are drawn
- **Figure**  Figure number in paper
- **Species**  Name of Species: Saccharomyces cerevisiae
- **Alcohol**  Additional alcohol in percent
- **Population**  Percentage of saturated population reached

Source

Description

A dataset containing the growth in volume of Saccharomyces cerevisiae and Schizosaccaromyces kephir cultivated in the mixed population (two series of experiments) under anaerobic conditions. Gause also measured the volume of the mixed population. The Volume is measured in "Amount of yeast", which refers to a standardized index, based on the alcohol production per Unit of Yeast.

Usage

gause_1934_book_f13

Format

A data frame with 47 rows and 5 variables:

- **Paper**  Paper from which data are drawn
- **Figure**  Figure number in paper
- **Species**  Name of Species: Saccharomyces cerevisiae, Schizosaccaromyces kephir and Mixed Population
- **Time**  Hour of experiment
- **Volume**  "Amount of Yeast" of Species

Source


Description

A dataset containing the growth in volume of Saccharomyces cerevisiae cultivated separately and in the mixed population (two series of experiments) with Schizosaccaromyces kephir under anaerobic conditions. The Volume is measured in "Amount of yeast", which refers to a standardized index, based on the alcohol production per Unit of Yeast.

Usage

gause_1934_book_f14
Description

A dataset containing the growth in volume of Schizosaccaromyces kephir cultivated separately and in the mixed population (two series of experiments) with Saccharomyces cerevisiae under anaerobic conditions. The Volume is measured in "Amount of yeast", which refers to a standardized index, based on the alcohol production per Unit of Yeast.

Usage

gause_1934_book_f15

Format

A data frame with 24 rows and 6 variables:

Paper Paper from which data are drawn
Figure Figure number in paper
Species Name of Species: Schizosaccaromyces kephir
Time Hour of experiment
Volume "Amount of Yeast" of Species
Treatment Treatments: Monoculture and Mixture

Source

Description

A dataset containing the growth in volume of *Schizosaccaromyces kephir* cultivated separately and in the mixed population with *Saccharomyces cerevisiae* under aerobic conditions. The Volume is measured in "Amount of yeast", which refers to a standardized index, based on the alcohol production per Unit of Yeast.

Usage

gause_1934_book_f16

Format

A data frame with 27 rows and 6 variables:

- **Paper**  Paper from which data are drawn
- **Figure**  Figure number in paper
- **Species**  Name of Species: *Schizosaccaromyces kephir*, *Saccharomyces cerevisiae* and Total_Yeast
- **Time**  Hour of experiment
- **Volume**  "Amount of Yeast" of Species
- **Treatment**  Treatments: Monoculture and Mixture

Source


Description

A dataset containing the abundance of *Paramecium caudatum* and *Stylonychia mytilis*, grown in monoculture and in mixture on buffered medium without wild bacteria.

Usage

gause_1934_book_f18
Format

A data frame with 28 rows and 6 variables:

- **Paper**  Paper from which data are drawn
- **Figure**  Figure number in paper
- **Species**  Name of Species: Paramecium caudatum and Stylonychia mytilis
- **Time**  Day of experiment
- **Individuals**  Number of Individuals
- **Treatment**  Treatments: Monoculture and Mixture

Source


---

Description

A dataset containing the abundance of Paramecium caudatum and Stylonychia mytilis, grown in monoculture and in mixture on buffered medium containing wild bacteria.

Usage

gause_1934_book_f19

Format

A data frame with 17 rows and 6 variables:

- **Paper**  Paper from which data are drawn
- **Figure**  Figure number in paper
- **Species**  Name of Species: Paramecium caudatum and Stylonychia mytilis
- **Time**  Day of experiment
- **Individuals**  Number of Individuals
- **Treatment**  Treatments: Monoculture and Mixture

Source

growth of Paramecium caudatum and Paramecium aurelia in monoculture

Description

A dataset containing the abundances and the volume of Paramecium caudatum and Paramecium aurelia, to determine the differences in reaching the saturating population regarding Volume and Number of individuals. "Volume" refers to a standardized index, meant to make the abundances of species comparable based on their relative sizes.

Usage

gause_1934_book_f21

Format

A data frame with 87 rows and 6 variables:

- **Paper**: Paper from which data are drawn
- **Figure**: Figure number in paper
- **Species**: Name of Species: Paramecium caudatum and Paramecium aurelia
- **Time**: Day of experiment
- **Volume**: "Volume" of Species
- **Individuals**: Number of Individuals

Source


Paramecium competition experiment

Description

A dataset containing the abundances of two Paramecium species grown in monoculture and mixture. Note, is for the same experiment as gause_1934_science_f02_03, except that data were digitized separately, and therefore have small variations. These might be useful for estimating observation error in the data digitization process. "Volume" refers to a standardized index, meant to make the abundances of species comparable based on their relative sizes.

Usage

gause_1934_book_f22
**Format**

A data frame with 72 rows and 4 variables:

- **Paper** Paper from which data are drawn
- **Figure** Figure number in paper
- **Day** Day of experiment
- **Species1** Name of Species 1
- **Volume_Species1** Volume of Paramecium caudatum
- **Species2** Name of Species 2
- **Volume_Species2** Volume of Paramecium aurelia
- **Treatment** Treatments: Pa and Pc monocultures, or mixture

**Source**


---

**Description**

A dataset containing the volume of Paramecium caudatum and Paramecium aurelia, grown in monoculture on buffered medium with two different wild bacteria concentrations ("one" and "half loop"). "Volume" refers to a standardized index, meant to make the abundances of species comparable based on their relative sizes.

**Usage**

gause_1934_book_f23

**Format**

A data frame with 61 rows and 6 variables:

- **Paper** Paper from which data are drawn
- **Figure** Figure number in paper
- **Species** Name of Species: Paramecium caudatum and Paramecium aurelia
- **Time** Day of experiment
- **Volume** "Volume" of Species
- **Treatment** Treatments: One Loop and Half Loop

**Source**

Description

A dataset containing the volume of Paramecium caudatum and Paramecium aurelia, grown in monoculture and in mixture on buffered medium with low wild bacteria concentration ("half loop" medium). "Volume" refers to a standardized index, meant to make the abundances of species comparable based on their relative sizes.

Usage

gause_1934_book_f24

gause_1934_book_f25

Description

A dataset containing the volume of Paramecium caudatum and Paramecium aurelia, grown in monoculture and in mixture on buffered medium with high wild bacteria concentration ("one loop" medium). "Volume" refers to a standardized index, meant to make the abundances of species comparable based on their relative sizes.

Usage

gause_1934_book_f25
Format

A data frame with 57 rows and 6 variables:

- **Paper** Paper from which data are drawn
- **Figure** Figure number in paper
- **Species** Name of Species: Paramecium caudatum and Paramecium aurelia
- **Time** Day of experiment
- **Volume** "Volume" of Species
- **Treatment** Treatments: Monoculture and Mixture

Source


---

**Growth of Stylorzychia pustulata in Monoculture and in Mixture**

Description

A dataset containing the abundances of Stylorzychia pustulata cultivated separately, and in the mixed populations with Paramecium caudatum and Paramecium aurelia and the abundances of P. caudatum and P. aurelia grown in mixture with S. pustulata.

Usage

- `gause_1934_book_f26`

Format

A data frame with 104 rows and 6 variables:

- **Paper** Paper from which data are drawn
- **Figure** Figure number in paper
- **Species** Name of Species: Paramecium caudatum, Paramecium aurelia and Stylorzychia pustulata
- **Time** Day of experiment
- **Individuals** Number of Individuals
- **Treatment** Treatments: Monoculture and Mixture(with Species)

Source

Elementary interaction between Didinium nasutum and Paramecium caudatum

Description

A dataset containing the abundances of Paramecium caudatum and Didinium nasutum grown in mixture. Didinium was introduced at day two. "Individuals” refers to the number of individuals per 0.5 c.c..

Usage

gause_1934_book_f28

Format

A data frame with 12 rows and 6 variables:

- **Paper**  Paper from which data are drawn
- **Figure**  Figure number in paper
- **Species**  Name of Species: Paramecium caudatum and Didinium nasutum
- **Time**  Day of experiment
- **Individuals**  Number of Individuals
- **Treatment**  Treatments: Monoculture and Mixture

Source


Paramecium/Didinium predator-prey experiment

Description

A dataset containing the abundances of Paramecium caudatum and Didinium nasutum grown in mixture. Didinium was introduced at different days. "Volume" refers to a standardized index, meant to make the abundances of species comparable based on their relative sizes.

Usage

gause_1934_book_f29
Format
A data frame with 62 rows and 7 variables:

- **Paper**  Paper from which data are drawn
- **Figure**  Figure number in paper
- **Species**  Name of Species: Didinium nasutum and Paramecium caudatum
- **Time**  Day of experiment
- **Volume**  Volume of Didinium
- **Individuals**  Number of Individuals
- **Treatment**  Treatments: D. nasutum introduced after 0, 24, 36 and 48 hrs

Source

---

Description
A dataset containing the abundances of Didinium nasutum and Paramecium caudatum grown in mixture on the medium of Osterhout.

Usage
gause_1934_book_f30

Format
A data frame with 16 rows and 6 variables:

- **Paper**  Paper from which data are drawn
- **Figure**  Figure number in paper
- **Time**  Day of experiment
- **Species**  Name of Species: Didinium nasutum and Paramecium caudatum
- **Individuals**  Number of Individuals
- **Treatment**  Treatment: Osterhout medium

Source
**gause_1934_book_f31**  The interaction between Didinium nasutum and Paramecium caudatum on oat medium

**Description**

A dataset containing the abundances of Didinium nasutum and Paramecium caudatum grown in mixture on oat medium with sediment.

**Usage**

gause_1934_book_f31

**Format**

A data frame with 12 rows and 6 variables:

- **Paper**  Paper from which data are drawn
- **Figure**  Figure number in paper
- **Time**  Day of experiment
- **Species**  Name of Species: Didinium nasutum and Paramecium caudatum
- **Individuals**  Number of Individuals
- **Treatment**  Treatment: Oat medium

**Source**


---

**gause_1934_book_f32**  Didinium/Paramecium predator/prey experiment

**Description**

A dataset containing the abundances of Paramecium caudatum and Didinium nasutum grown in mixture. Note, is for the same experiment as gause_1934_science_f01, except that data were digitized separately, and therefore have small variations. These might be useful for estimating observation error in the data digitization process.

**Usage**

gause_1934_book_f32
Format

A data frame with 17 rows and 8 variables:

Paper Paper from which data are drawn
Figure Figure number in paper
Day Day of experiment
Prey Name of Prey Species
Individuals_Prey Number of Prey Individuals
Predator Name of Predator Species
Individuals_Predator Number of Predator Individuals
Immigration Is immigration occurring in this time-step? (yes or no)

Source


---

gause_1934_book_f39.1 The interaction between Paramecium bursaria and Schizosaccharomyces pombe

Description

A dataset containing the abundances of Paramecium bursaria and Schizosaccharomyces pombe grown in mixture.

Usage

gause_1934_book_f39.1

Format

A data frame with 36 rows and 5 variables:

Paper Paper from which data are drawn
Figure Figure number in paper
Time Day of experiment
Species Name of Species: Paramecium bursaria and Schizosaccharomyces pombe
Individuals Number of Individuals

Source

Didinium/Paramecium predator/prey experiment

Description
A dataset containing the abundances of Paramecium caudatum and Didinium nasutum grown in mixture. Note, is for the same experiment as gause_1934_book_f32, except that data were digitized separately, and therefore have small variations. These might be useful for estimating observation error in the data digitization process.

Usage

gause_1934_science_f01

Format
A data frame with 17 rows and 3 variables:

- Paper  Paper from which data are drawn
- Figure  Figure number in paper
- Day     Day of experiment
- Prey    Name of Prey Species
- Individuals_Prey  Number of Prey Individuals
- Predator Name of Predator Species
- Individuals_Predator  Number of Predator Individuals
- Immigration  Is immigration occurring in this time-step? (yes or no)

Source

Paramecium competition experiment

Description
A dataset containing the abundances of two Paramecium species grown in monoculture and mixture. Note, is for the same experiment as gause_book_1934_f22, except that data were digitized separately, and therefore have small variations. These might be useful for estimating observation error in the data digitization process. "Volume" refers to a standardized index, meant to make the abundances of species comparable based on their relative sizes.
Usage
gause_1934_science_f02_03

Format
A data frame with 63 rows and 4 variables:

- **Paper**  Paper from which data are drawn
- **Figure**  Figure number in paper
- **Day**  Day of experiment
- **Species1**  Name of Species 1
- **Volume_Species1**  Volume of Paramecium caudatum
- **Species2**  Name of Species 2
- **Volume_Species2**  Volume of Paramecium aurelia
- **Treatment**  Treatments: Pa and Pc monocultures, or mixture

Source

Description
A dataset containing the abundance of Cheyletus eruditus and Aleuiroglyphus agilis, as a predator-prey system under different food conditions for the prey (wheat and millet). Gause’s goal was to determine the influence of ecological factors on predator-prey dynamics. The number of individuals is reported as Individuals per 0.2 g prey food.

Usage
gause_1936_AnE_f01

Format
A data frame with 34 rows and 6 variables:

- **Paper**  Paper from which data are drawn
- **Figure**  Figure number in paper
- **Time**  Day of experiment
- **Species**  Name of Species: Cheyletus eruditus and Aleuiroglyphus agilis
- **Individuals**  Number of Individuals per 0.2 g
- **Treatment**  Treatments: Wheat, Millet, Wheat+Millet
Source


Interaction between predators (Cheyletus eruditus) and prey (Aleuroglyphus agilis) with occasional immigration

Description

A dataset containing the abundance of Cheyletus eruditus and Aleuroglyphus agilis, as a predator-prey system with an occasional immigration of the prey on the 63rd day. The number of individuals is reported as Individuals per 0.2 g prey food. Wheat flour was used as food.

Usage

gause_1936_AnE_f03.1

Format

A data frame with 22 rows and 6 variables:

- **Paper**: Paper from which data are drawn
- **Figure**: Figure number in paper
- **Time**: Day of experiment
- **Species**: Name of Species: Cheyletus eruditus and Aleuroglyphus agilis
- **Individuals**: Number of Individuals per 0.2 g
- **Treatment**: Treatments: Immigration or NA

Source

Description

A dataset containing the abundance of Cheyletus eruditus and Aleuroglyphus agilis, as a predator-prey system with an artificial everyday immigration of predator and prey. The number of individuals is reported as Individuals per 0.2 g prey food. Wheat flour was used as food.

Usage

gause_1936_AnE_f03.3a

gause_1936_AnE_f03.3b

Format

A data frame with 24 rows and 6 variables:

- **Paper**: Paper from which data are drawn
- **Figure**: Figure number in paper
- **Time**: Day of experiment
- **Species**: Name of Species: Cheyletus eruditus and Aleuroglyphus agilis
- **Individuals**: Number of Individuals per 0.2 g
- **Treatment**: Treatment: Immigration

Source

Format

A data frame with 26 rows and 6 variables:

Paper  Paper from which data are drawn
Figure  Figure number in paper
Time  Day of experiment
Species  Name of Species: Cheyletus eruditus and Aleuiroglyphus agilis
Individuals  Number of Individuals per 0.2 g
Treatment  Treatment: Immigration

Source


Description

A dataset containing the abundance of Cheyletus eruditus and Aleuiroglyphus agilis, as a predator-prey system on semoletta and wheat flour. A fraction at the beginning of each experiment shows the initial relation between predators and prey; e.g. 10/5 means 10 prey and 5 predators (reported in variable "Treatment"). Dataset includes age structured population abundances. The number of individuals is reported as Individuals per 0.2 g prey food. Wheat flour was used as food.

Usage

gause_1936_AnE_t02

Format

A data frame with 191 rows and 12 variables:

Paper  Paper from which data are drawn
Table  Table number in paper
Time  Day of experiment
Species  Name of Species: Cheyletus eruditus and Aleuiroglyphus agilis
Total_Individuals  Number of all Individuals of Species per 0.2 g
Female  Number of female individuals per 0.2 g
Male  Number of male individuals per 0.2 g
Imago  Number of adult Individuals per 0.2 g
Hexapod  Number of hexapod larvae stage Individuals per 0.2 g
Octapod  Number of octapod larvae stage Individuals per 0.2 g
Initial_Fraction  Initial Number of prey/predator
Treatment  Treatments: Wheat and Semoletta
Source


Description

A dataset containing the abundance of Paramecium bursaria and Saccharomyces exiguus, as a predator-prey system. The primary difference among experimental replicates is the initial abundance of the two species. The number of individuals is reported as Individuals per 0.5 cm³ for Paramecium, and as number of individuals per 0.1 cm³ for Saccharomyces.

Usage

Usage

Format

A data frame with 266 rows and 6 variables:

- **Paper**  Paper from which data are drawn
- **Table**  Table number in paper
- **Experiment**  Experimental replicate number
- **Time**  Day of experiment
- **Species**  Name of Species: Cheyletus eruditus and Aleuiroglyphus agilis
- **Individuals**  Number of all Individuals of Species per 0.2 g

Source

gause_wrapper

Automated wrapper for Gause fitting functions

Description

Automatically runs routine for finding starting values and optimal parameter values for a Lotka-Volterra interaction system. Using the default functions, species dynamics follow the form \( \frac{dn_i}{dt} = n_i \cdot (r_i + a_{ii} \cdot n_i + \text{sum}_j(a_{ij} \cdot n_j)) \) where \( r_i \) are the elements of vector \( r \), and \( a_{ij} \) are the elements of matrix \( A \).

Usage

```r
gause_wrapper(
  time, species, 
  N_starting = NULL, 
  r_starting = NULL, 
  A_starting = NULL, 
  doplot = TRUE, 
  keeptimes = FALSE, 
  parm_signs = NULL, 
  doopt = TRUE, 
  ...
)
```

Arguments

time Vector of time steps corresponding to observations in species data.frame.

species A data.frame with one column per species to be fitted. Note - column names cannot include white spaces or non-standard special characters.

N_starting Optional starting values for initial abundances.

r_starting Optional starting values for species growth rates. If a value is set to zero, it forces that parameter to zero in the fitting. Values of NA are ignored. Defaults to NULL (no starting values).

A_starting Optional starting values for species interaction coefficients. If a value is set to zero, it forces that parameter to zero in the fitting. Values of NA are ignored. Defaults to NULL (no starting values).

doplot Logical. Should the resulting model be plotted? Defaults to TRUE.

keeptimes Should predictions be given for the points in the "time" vector, or for a list of 100 evenly spaced time points? Defaults to FALSE.

parm_signs Optional variable specifying signs for parameters. Defaults to NULL (automatically selected).

doopt Should optimizer be used (if TRUE), or should the initial linearized estimates be applied (if FALSE)? Defaults to TRUE.

... Optional additional arguments to be passed to ode and optim functions.
get_lag

Value

A list with simulated time series (out), parameter estimates (parameter_intervals), optimizer output (optout), and raw data used for fitting (rawdata).

Examples

```r
# load competition data
data("gause_1934_science_f02_03")

# subset out data from species grown in mixture
mixturedat<-gause_1934_science_f02_03[gause_1934_science_f02_03$Treatment=="Mixture",]

# extract time and species data
time<-mixturedat$Day
species<-data.frame(mixturedat$Volume_Species1, mixturedat$Volume_Species2)
colnames(species)<-c("P_caudatum", "P_aurelia")

# run wrapper
gause_out<-gause_wrapper(time=time, species=species)
```

get_lag

Description

Calculates time-lagged observations for variable x, separated by treatment.

Usage

```r
get_lag(x, time, tau = 1, treatment = NULL, mindt = 0, maxdt = Inf)
```

Arguments

- **x**: The time series from which time lagged observations are desired (e.g. population sizes)
- **time**: The time steps corresponding to each observation
- **tau**: Number of time steps to use between lagged components - defaults to 1
- **treatment**: An optional vector of treatment conditions - time lags will only be computed separately within treatments - defaults to NULL (i.e. no treatments)
- **mindt**: Minimum dt allowed between observations - defaults to 0
- **maxdt**: Maximum dt allowed between observations - defaults to Inf
get_logistic

Value

Returns a data.frame with 7 columns: x (unlagged time series data); laggedx (lagged time series data); xmid (average of time series and lagged time series values); dt (time lag between x and laggedx); time (time for observation x); laggedtime (time for observation laggedx); treatment (treatment for observation)

Examples

data(gause_1934_science_f02_03)
lagged_data <- get_lag(x=gause_1934_science_f02_03$Volume_Species1,
  time = gause_1934_science_f02_03$Day,
  treatment = gause_1934_science_f02_03$Treatment)

get_logistic

Logistic Growth

Description

Calculates logistic growth for population based on formula \( N_t = \frac{K \times (N_0 \times \exp(r \times t))}{K + N_0 \times (\exp(r \times t) - 1)} \)

Usage

get_logistic(time, N0, r, K)

Arguments

time The time steps corresponding to each observation
N0 Initial Population Size
r Growth rate
K Carrying Capacity

Value

population size \( N \) for each time steps as a vector

Examples

# load Gause competition data
data(gause_1934_science_f02_03)
# extract monoculture data for P.c.
Pcmono<-gause_1934_science_f02_03[gause_1934_science_f02_03$Treatment=="Pc",]

# calculate lag and per-capita growth
lagged_data_Pc <- get_lag(x=Pcmono$Volume_Species1,
  time = Pcmono$Day)
Pcmono$dNNdt_Pc <- percap_growth(x=lagged_data_Pc$x, laggedx=lagged_data_Pc$laggedx,
dt=lagged_data_Pc$dt)

# fit linear model to get dN/dt = r + s*N
mod_Pc<-lm(dNNdt_Pc~Volume_Species1, Pcmono)
rsn_pars<-coef(mod_Pc)

# transform into logistic growth parameters
logistic_pars<-c(r=unname(rsn_pars["(Intercept)"]),
                 K=unname(-rsn_pars["(Intercept)"]/rsn_pars["Volume_Species1"]))

# fit with nls, using linear model estimates as starting values for parameters
nls_mod<-nls(Volume_Species1~get_logistic(time = Day, N0, r, K),
              data=Pcmono,
              start=c(N0=unname(Pcmono$Volume_Species1[which.min(Pcmono$Day)]),
                     r=unname(logistic_pars["r"]), K=unname(logistic_pars["K"])))
summary(nls_mod)

# plot results
plot(Volume_Species1~Day, Pcmono, type="b", ylab="P. caudatum Volume")
timesq<-seq(0, 30, length=100)
Ntest<-get_logistic(time = timesq, N0=coef(nls_mod)["N0"], r=coef(nls_mod)["r"],
                     K=coef(nls_mod)["K"])
lines(timesq, Ntest, col="red")

---

**huffaker_1963**  
**Huffaker Mite Data**

---

**Description**

A dataset containing the abundances mite species from some of the Huffaker experiments.

**Usage**

huffaker_1963

**Format**

A data frame with 168 rows and 6 variables:

- **Paper**  Paper from which data are drawn
- **Figure**  Figure number in paper
- **Species**  Species name
- **Weeks**  Experiment week
- **Individuals**  Number of individuals
- **Treatment**  60 vs. 24-week experiments
lv_interaction

Lotka-Volterra Interactions

Description
Calculates $dn/dt$ for $n$ species in a Lotka-Volterra system, following the form: $dni/dt = ni * (ri + aii * ni + \sum_j (aij * nj))$. Note that $aii$ coefficients can be positive or negative, although positive coefficients risk having the system run to infinite population sizes, which will crash the function.

Usage

```
lv_interaction(time, n, parms)
```

Arguments

- `time`: The time steps corresponding to each observation - exists to interface with `ode` function, but should be left blank.
- `n`: A vector of species abundances
- `parms`: A vector of parameters - the first $n$ elements should be the growth rates $r1$, $r2$, ... $rn$ for all $n$ species. The remaining terms should be the elements of the interaction matrix $A$, listed in the order $a11$, $a12$, ... $a1n$, $a21$, $a22$, ... $a2n$, ... $an1$, $an2$, ... $ann$.

Value

vector of growth rates for each species

Examples

```
# load data from competition experiment
data(gause.1934_science_f02.03)

# subset data to include just mixtures
mixturedata<-gause.1934_science_f02.03[gause.1934_science_f02.03$Treatment=="Mixture",]

# get time-lagged observations for each species
Pc_lagged<-get_lag(x = mixturedata$Volume_Species1, time = mixturedata$Day)  
Pa_lagged<-get_lag(x = mixturedata$Volume_Species2, time = mixturedata$Day)

# calculate per-capita growth rates
Pc_dNNdt<-percap_growth(x = Pc_lagged$x, laggedx = Pc_lagged$laggedx, dt = Pc_lagged$dt)  
Pa_dNNdt<-percap_growth(x = Pa_lagged$x, laggedx = Pa_lagged$laggedx, dt = Pa_lagged$dt)

# fit linear models to dNNdt, based on average
```
# abundances between current and lagged time steps
Pc_mod_dat<-data.frame(Pc_dNNdt=Pc_dNNdt, Pc=Pc_lagged$laggedx, Pa=Pa_lagged$laggedx)
mod_comp_Pc<-lm(Pc_dNNdt~Pc+Pa, data=Pc_mod_dat)

Pa_mod_dat<-data.frame(Pa_dNNdt=Pa_dNNdt, Pa=Pa_lagged$laggedx, Pc=Pc_lagged$laggedx)
mod_comp_Pa<-lm(Pa_dNNdt~Pa+Pc, data=Pa_mod_dat)

# model summaries
summary(mod_comp_Pc)
summary(mod_comp_Pa)

# extract parameters
# note - linear regressions give us dynamics in the form:
# dni/nidt = (Intercept) + (n1_slope) * n1 + (n2_slope) n2
# and thus:
# dni/dt = n1*((Intercept) + (n1_slope) * n1 + (n2_slope) n2)

# growth rates
r1 <- unname(coef(mod_comp_Pc)["(Intercept)"])
r2 <- unname(coef(mod_comp_Pa)["(Intercept)"])

# self-limitation
a11 <- unname(coef(mod_comp_Pc)["Pc"])
a22 <- unname(coef(mod_comp_Pa)["Pa"])

# effect of Pa on Pc
a12 <- unname(coef(mod_comp_Pc)["Pa"])

# effect of Pc on Pa
a21 <- unname(coef(mod_comp_Pa)["Pc"])

# run ODE:
# make parameter vector:
parms <- c(r1, r2, a11, a12, a21, a22)
initialN <- c(1, 1)
out <- deSolve::ode(y=initialN, times=1:25, func=lv_interaction, parms=parms)
matplot(out[,1], out[,-1], type="l",
	xlab="time", ylab="N", col=c("black","red"), lty=c(1,3), lwd=2, ylim=c(0, 150))
legend("topleft", c("Pc", "Pa"), col=c(1,2), lwd=2, lty=c(1,3))

# now, plot in points from data
points(mixturedata$Day, mixturedata$Volume_Species1, col=1)
points(mixturedata$Day, mixturedata$Volume_Species2, col=2)

---

**lv_interaction_log**

**Lotka-Volterra Interactions in Log Space**

**Description**

Calculates dn/dt for n species in a Lokta-Volterra system, in log space, following the form: dlog(ni)/dt = (ri + aii * ni + sum_j(aij * nj)) This form can be helpful for optimization routines where species abundances are close to zero.
Usage

```r
lv_interaction_log(time, n_log, parms)
```

Arguments

- **time**: The time steps corresponding to each observation - exists to interface with ode function, but should be left blank.
- **n_log**: A vector of species abundances, in log space
- **parms**: A vector of parameters - the first n elements should be the growth rates r1, r2, ... rn for all n species. The remaining terms should be the elements of the interaction matrix A, listed in the order a11, a12, ... an1, a21, a22, ... a2n, ... ann.

Value

- vector of growth rates for each species in log space

---

**lv_optim**

*Optimizer for Lotka-Volterra Interactions*

Description

Identifies optimal parameter values for a Lotka-Volterra interaction system.

Usage

```r
lv_optim(
  pars,
  opt_data,
  parm_signs,
  standardize = TRUE,
  odefun = lv_interaction_log
)
```

Arguments

- **pars**: A vector of parameter values in log space to be optimized. Must include a logged starting abundance for each species, followed by the logged absolute values of the growth rates, followed by the logged absolute value of the elements of the interaction matrix.
- **opt_data**: Abundance data for optimization. Must include one column labeled 'time' with time steps, and a column for each species abundance.
- **parm_signs**: A vector that provides the desired sign of each parameter (i.e. -1 or 1). If value is zero, then the term is held at zero (but should be left out of the pars vector).
- **standardize**: A logical, defaulting to TRUE - should error be calculated based on standardized values of outputs? Allows for more equal weighting of observed variables.
- **odefun**: The function to use to simulate the ODE - defaults to `lv_interaction_log`
Value

squared error between model fits for given parameter values, and observations

Examples

```r
# load data from competition experiment
data(gause_1934_book_f32)

# keep all data - no separate treatments exist for this experiment
predatorpreydata<-gause_1934_book_f32

# get time-lagged observations for each species
prey_lagged<-get_lag(x = predatorpreydata$Individuals_Prey, time = predatorpreydata$Day)
predator_lagged<-get_lag(x = predatorpreydata$Individuals_Predator, time = predatorpreydata$Day)

# calculate per-capita growth rates
prey_dNNdt<-percap_growth(x = prey_lagged$x, laggedx = prey_lagged$laggedx, dt = prey_lagged$dt)
predator_dNNdt<-percap_growth(x = predator_lagged$x,
  laggedx = predator_lagged$laggedx, dt = predator_lagged$dt)

# fit linear models to dNNdt, based on average
# abundances between current and lagged time steps
prey_mod_dat<-data.frame(prey_dNNdt=prey_dNNdt, prey=prey_lagged$laggedx,
predator=predator_lagged$laggedx)
mod_prey<-lm(prey_dNNdt~prey+predator, data=prey_mod_dat)
predator_mod_dat<-data.frame(predator_dNNdt=predator_dNNdt,
predator=predator_lagged$laggedx, prey=prey_lagged$laggedx)
mod_predator<-lm(predator_dNNdt~predator+prey, data=predator_mod_dat)

# model summaries
summary(mod_prey)
summary(mod_predator)

# extract parameters
# growth rates
r1 <- unname(coef(mod_prey)["(Intercept)"])
r2 <- unname(coef(mod_predator)["(Intercept)"])

# self-limitation
a11 <- unname(coef(mod_prey)["prey"])
a22 <- unname(coef(mod_predator)["predator"])

# effect of Pa on Pc
a12 <- unname(coef(mod_prey)["predator"])
# effect of Pc on Pa
a21 <- unname(coef(mod_predator)["prey"])

# run ODE:
# make parameter vector:
parms <- c(r1, r2, a11, a12, a21, a22)
```
lv_optim

initialN <- c(4, 0.1)
out <- deSolve::ode(y=initialN, times=seq(1, 17, length=100), func=lv_interaction, parms=parms)
matplot(out[,1], out[,-1], type="l",
  xlab="time", ylab="N", col=c("black","red"), lty=c(1,3), lwd=2, ylim=c(0,60))
legend("topright", c("Pc", "Dn"), col=c(1,2), lwd=2, lty=c(1,3))

# now, plot in points from data
points(predatorpreydata$Day, predatorpreydata$Individuals_Predator, col=2)
points(predatorpreydata$Day, predatorpreydata$Individuals_Prey, col=1)

# uh-oh - This is a bad fit. This suggests that our linear model
# approximation isn’t very good. Instead, we should try optimizing
# directly using the ode solver

# Re-run using an optimizer
# Data for the optimizer:
# Must have a column with time steps labeled 'time', and
# columns for each species in the community.
opt_data<-data.frame(time=predatorpreydata$Day, Prey=predatorpreydata$Individuals_Prey,
                     Predator=predatorpreydata$Individuals_Predator)

# Save the signs of the parameters -
# optimizer works in log space, so these
# must be specified separately
parm_signs<-sign(parms)

# parameter vector for optimizer -
# must be a vector with, first, the
# starting abundances in log space,
# and second, the parameter values,
# again in log space
pars<-c(log(initialN), log(abs(parms)))

# run optimizer
optout<-optim(par = pars, fn = lv_optim, hessian = TRUE,
               opt_data=opt_data, parm_signs=parm_signs)

# extract parameter vector:
parms <- exp(optout$par[-c(1:2)])*parm_signs
initialN <- exp(optout$par[1:2])

out <- deSolve::ode(y=initialN, times=seq(1, 17, length=100), func=lv_interaction, parms=parms)
matplot(out[,1], out[,-1], type="l",
  xlab="time", ylab="N", col=c("black","red"), lty=c(1,3), lwd=2, ylim=c(0,60))
legend("topright", c("Pc", "Dn"), col=c(1,2), lwd=2, lty=c(1,3))

# now, plot in points from data
points(predatorpreydata$Day, predatorpreydata$Individuals_Predator, col=2)
points(predatorpreydata$Day, predatorpreydata$Individuals_Prey, col=1)

# get rough estimate of confidence intervals
fisher_info<-solve(-optout$hessian)
optout$par_sd<-sqrt(abs(diag(fisher_info)))
parm_signs_sp<-c(rep(1, ncol(opt_data)-1), parm_signs)
parameter_intervals<-data.frame(lower_sd=exp(optout$par-optout$par_sd)*parm_signs_sp,
                                  mu=exp(optout$par)*parm_signs_sp,
                                  upper_sd=exp(optout$par+optout$par_sd)*parm_signs_sp)
rownames(parameter_intervals)<-c("prey", "predator", "r1", "r2", "a11", "a12", "a21", "a22")
parameter_intervals

mclaren_1994_f03  Wolf, Moose, and Fir dynamics from Isle Royale

Description
A dataset containing the abundances of wolves, moose, and fir trees from the Isle Royale study of McLaren et al.

Usage
mclaren_1994_f03

Format
A data frame with 140 rows and 7 variables:

Paper  Paper from which data are drawn
Figure  Figure number in paper
year    Year of measurements
Species Species name
width   Width of tree rings
individuals Number of wolf or moose individuals
AET.mm. AET water availability index

Source
ode_prediction

Optimizer extension

Description

Takes in parameter values in the form returned by the gause_wrapper function, and calculates expected abundances for all n species, returned as a single vector. This function is potentially useful in combination with other optimizer software, e.g. as might be used for hypothesis testing.

Usage

ode_prediction(pars_full, time, N)

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pars_full</td>
<td>Initial Population Size</td>
</tr>
<tr>
<td>time</td>
<td>A vector of times. Must be repeated once per species.</td>
</tr>
<tr>
<td>N</td>
<td>Number of species. Can be either a number, or a vector the same length as time.</td>
</tr>
</tbody>
</table>

Value

a stacked vector with predicted abundances for all species

Examples

```r
# load competition data
data("gause_1934_science_f02_03")

# subset out data from species grown in mixture
mixturedat<-gause_1934_science_f02_03[gause_1934_science_f02_03$Treatment=="Mixture",]

# extract time and species data
time<-mixturedat$Day
species<-data.frame(mixturedat$Volume_Species1, mixturedat$Volume_Species2)
colnames(species)<-c("P_caudatum", "P_aurelia")

# run wrapper
gause_out<-gause_wrapper(time=time, species=species)

# number of species
N<-ncol(gause_out$rawdata)-1

# parameters
pars_full<-c(gause_out$parameter_intervals$mu)

# data.frame for optimization
fittigdata<-data.frame(y=unlist(gause_out$rawdata[,1]),
                       time=gause_out$rawdata$time,
                       N=N)

yest<-ode_prediction(pars_full, time=fittigdata$time, N=fittigdata$N)
```
plot(fittigdata$y, yest, xlab="observation", ylab="prediction")
abline(a=0, b=1, lty=2)

#example of how to apply function, using nls()
mod<-nls(y~ode_prediction(pars_full, time, N),
        start = list(pars_full=pars_full),
        data=fittigdata)
summary(mod)

---

`percap_growth`  

**Per-capita growth rate**

**Description**

Calculates per-capita growth rate, using log ratios following the formula \( \frac{dN}{Ndt} = \frac{\log(N(t)/N_0)}{dt} \).

**Usage**

`percap_growth(x, laggedx, dt)`

**Arguments**

- `x` : Abundance  
- `laggedx` : Lagged abundance  
- `dt` : Time lag between observations

**Value**

Per-capita growth rate

**Examples**

data(gause_1934_science_f02_03)
lagged_data <- get_lag(x=gause_1934_science_f02_03$Volume_Species1,
        time = gause_1934_science_f02_03$Day,
        treatment = gause_1934_science_f02_03$Treatment)
dNNdt <- percap_growth(x=lagged_data$x, laggedx=lagged_data$laggedx,
        dt=lagged_data$dt)
Description

Tests goodness of fit for predictions vs. observations. This statistic can be thought of in the same way as a classic "R2", except that it measures scatter around the 1-1 line, rather than around a fitted regression line of observed vs. predicted values. Value close to 1 indicate that predictions match observations closely. Values at or below zero indicate that predictions do not match observations any better than the grand mean taken across all observations.

Usage

test_goodness_of_fit(observed, predicted, bycolumn = FALSE, droptimecol = TRUE)

Arguments

- **observed**: A vector or matrix of observed values.
- **predicted**: A vector or matrix of predicted values.
- **bycolumn**: If TRUE, then separate values are calculated for each column in observed and predicted.
- **droptimecol**: If TRUE, will automatically remove the column labeled "time" in the predicted variable. This is useful for dealing with the default output of the gause_wrapper function. Defaults to FALSE.

Examples

```r
#load competition data
data("gause_1934_science_f02_03")

#subset out data from species grown in mixture
mixturedat<-gause_1934_science_f02_03[gause_1934_science_f02_03$Treatment=="Mixture",]

#extract time and species data
time<-mixturedat$Day
species<-data.frame(mixturedat$Volume_Species1, mixturedat$Volume_Species2)
colnames(species)<-c("P_caudatum", "P_aurelia")

#run wrapper
#note - keeptimes=TRUE is needed, so that predicted time steps match observed time steps
gause_out<-gause_wrapper(time=time, species=species, keeptimes = TRUE)

#calculate goodness of fit
test_goodness_of_fit(observed=species, predicted=gause_out)

# > 0.9 for both time series - these are good fits!
```
Index

* Gause
  gause_wrapper, 32
  get_lag, 33
  get_logistic, 34
  lv_interaction, 36
  lv_interaction_log, 37
  lv_optim, 38
  ode_prediction, 42
  percap_growth, 43
* Lokta-Volterra
  gause_wrapper, 32
  lv_interaction, 36
  lv_interaction_log, 37
  lv_optim, 38
* competition
  get_lag, 33
  get_logistic, 34
  ode_prediction, 42
  percap_growth, 43
* datasets
  gause_1931_AmN_f01, 4
  gause_1931_AmN_f02, 5
  gause_1931_AmN_f03, 5
  gause_1932_RR_t05, 6
  gause_1934_book_app_t01, 7
  gause_1934_book_app_t02, 8
  gause_1934_book_app_t03, 8
  gause_1934_book_app_t04, 9
  gause_1934_book_app_t05, 10
  gause_1934_book_f04, 11
  gause_1934_book_f09, 11
  gause_1934_book_f10, 12
  gause_1934_book_f11, 12
  gause_1934_book_f12, 13
  gause_1934_book_f13, 14
  gause_1934_book_f14, 14
  gause_1934_book_f15, 15
  gause_1934_book_f16, 16
  gause_1934_book_f18, 16
  gause_1934_book_f19, 17
  gause_1934_book_f21, 18
  gause_1934_book_f22, 18
  gause_1934_book_f23, 19
  gause_1934_book_f24, 20
  gause_1934_book_f25, 20
  gause_1934_book_f26, 21
  gause_1934_book_f28, 22
  gause_1934_book_f29, 22
  gause_1934_book_f30, 23
  gause_1934_book_f31, 24
  gause_1934_book_f32, 24
  gause_1934_book_f39, 25
  gause_1934_science_f01, 26
  gause_1934_science_f02-03, 26
  gause_1936_AnE_f01, 27
  gause_1936_AnE_f03.1, 28
  gause_1936_AnE_f03.3a, 29
  gause_1936_AnE_f03.3b, 29
  gause_1936_AnE_t02, 30
  gause_1936_AnE_t03, 31
  huffaker_1963, 35
  mclaren_1994_f03, 41
* growth
  percap_growth, 43
* interaction
  gause_wrapper, 32
  lv_interaction, 36
  lv_interaction_log, 37
  lv_optim, 38
* logistic growth
  get_logistic, 34
* optimization
  gause_wrapper, 32
  lv_optim, 38
  ode_prediction, 42
* time lag
  get_lag, 33

45
INDEX

gause_1931_AmN_f02, 5
gause_1931_AmN_f03, 5
gause_1932_QR_t05, 6
gause_1934_book_app_t01, 7
gause_1934_book_app_t02, 8
gause_1934_book_app_t03, 8
gause_1934_book_app_t04, 9
gause_1934_book_app_t05, 10
gause_1934_book_f04, 11
gause_1934_book_f09, 11
gause_1934_book_f10, 12
gause_1934_book_f11, 12
gause_1934_book_f12, 13
gause_1934_book_f13, 14
gause_1934_book_f14, 14
gause_1934_book_f15, 15
gause_1934_book_f16, 16
gause_1934_book_f18, 16
gause_1934_book_f19, 17
gause_1934_book_f21, 18
gause_1934_book_f22, 18
gause_1934_book_f23, 19
gause_1934_book_f24, 20
gause_1934_book_f25, 20
gause_1934_book_f26, 21
gause_1934_book_f28, 22
gause_1934_book_f29, 22
gause_1934_book_f30, 23
gause_1934_book_f31, 24
gause_1934_book_f32, 24
gause_1934_book_f39.1, 25
gause_1934_science_f01, 26
gause_1934_science_f02, 26
gause_1936_AnE_f01, 27
gause_1936_AnE_f03.1, 28
gause_1936_AnE_f03.3a, 29
gause_1936_AnE_f03.3b, 29
gause_1936_AnE_t02, 30
gause_1936_AnE_t03, 31
gause_wrapper, 32
gauseR, 3
get_lag, 33
get_logistic, 34

huffaker_1963, 35
lv_interaction, 36
lv_interaction_log, 37
lv_optim, 38

mclaren_1994_f03, 41
ode_prediction, 42
percap_growth, 43
test_goodness_of_fit, 44