Package 'bSi'

January 24, 2024

Title Modeling and Computing Biogenic Silica ('bSi') from Inland and Pelagic Sediments

Version 1.0.0

Description
A collection of integrated tools designed to seamlessly interact with each other for the analysis of biogenic silica 'bSi' in inland and marine sediments. These tools share common data representations and follow a consistent 'API' design. The primary goal of the 'bSi' package is to simplify the installation process, facilitate data loading, and enable the analysis of multiple samples for biogenic silica fluxes. This package is designed to enhance the efficiency and coherence of the entire 'bSi' analytic workflow, from data loading to model construction and visualization tailored towards reconstructing productivity in aquatic ecosystems.

License GPL-3

Encoding UTF-8

RoxygenNote 7.2.3

Imports dplyr, ggplot2, ggpubr

Depends cowplot, tidyr, tibble

Suggests knitr, rmarkdown, testthat (>= 3.0.0)

Config/testthat/edition 3

VignetteBuilder knitr

NeedsCompilation no

Author George Okoko [aut, cre],
Rogers Owit [ctb]

Maintainer George Okoko <goo2112@columbia.edu>

Repository CRAN

Date/Publication 2024-01-24 15:52:57 UTC

R topics documented:

`flux` *Calculate bSi fluxes based on pbSi (percent bSi) and MARS*

Description

Calculate bSi fluxes based on pbSi (percent bSi) and MARS

Usage

```
flux(pbSi = NULL, MARS = NULL, data_file = NULL, output_csv_file = NULL)
```

Arguments

`pbSi`	Numeric vector of percent biogenic silica.
`MARS`	Numeric vector of sediment mass accumulation rates.
`data_file`	Path to a CSV file containing data with columns 'pbSi' and 'MARS'.
`output_csv_file`	Path to save the calculated flux values as a CSV file.

Value

A data frame with input values and calculated flux.

Examples

```
#Example 1: Using vectors
flux_values <- flux(pbSi = c(2, 5, 8), MARS = c(10, 15, 20),
output_csv_file = tempfile("flux_output1.csv"))
#Example 2: Using data from a file
data_file <- system.file("extdata", "example_data.csv", package = "bSi")
flux_values <- flux(data_file = data_file, output_csv_file = tempfile("flux_output2.csv"))
```

pbSi *Calculate %bSi*

Description

Calculate the bSi percent based on the provided formula.

Usage

```
pbSi(
  C0,
  Vol_Na2CO3,
  Molar_mass_silicon,
  sample_dry_weight,
  output_dir = tempdir()
)
```

Arguments

C0	Concentration of silica from biogenic sources (mole/L).
Vol_Na2CO3	Vol. Na2CO3 (L) for samples digested in 40.0 ml Na2CO3.
Molar_mass_silicon	Molar mass of silicon (g/mole).
sample_dry_weight	Dry sample weight, the measured weight of each sample in grams (0.05 +/- 0.005g).
output_dir	The directory where the output CSV file should be saved. Defaults to the temporary directory (tempdir()).

Value

%bSi value calculated using the formula.

Examples

```
C0 <- 0.01# Concentration of silica from biogenic sources mol/L
Vol_Na2CO3 <- 0.04  # Vol. Na2CO3 (L)
Molar_mass_silicon <- 28.09  # Molar mass of silicon (g/mol)
sample_dry_weight <- 0.05  # Sample dry weight (g)
result <- pbSi(C0, Vol_Na2CO3, Molar_mass_silicon, sample_dry_weight, output_dir = tempdir())
print(result)
```

plotStdC *Plot Silica Concentrations vs. Absorbance*

Description

This function plots the Standard Calibration curves from known concentration of silica in standard solutions against absorbance values from spectrophotometer analysis. It takes Silica concentration values as Y argument and their absorbance values from spectrophotometer as X argument then creates a scatter plot, fits a line of best fit, and returns the y-intercept and R-squared values.

Usage

```
plotStdC(
  concentration,
  absorbance,
  title = "Concentration vs. Absorbance",
  xlab = "Absorbance",
  ylab = "Concentration (Millimoles)"
)
```

Arguments

concentration	A numeric vector of concentration values.
absorbance	A numeric vector of absorbance values.
title	A character string for the plot title.
xlab	A character string for the x-axis label.
ylab	A character string for the y-axis label.

Value

A list with components:

- intercept: The y-intercept of the fitted line.
- rsquared: The R-squared value of the fitted line.
- equation : The equation of the fitted line in the form y=mx+C

Examples

```
concentration <- c(1, 2, 3, 4, 5)
absorbance <- c(0.1, 0.3, 0.6, 0.8, 1.2)
plotStdC(concentration, absorbance,
          title = "Concentration vs. Absorbance",
          xlab = "Absorbance",
          ylab = "Concentrations")
```

silco *Calculate concentration(C0) of Silica from samples*

Description

This function calculates C0 values based on the provided slope (m) and y-intercept (y) from the tdgraph function. The sample data is loaded from a CSV file and the intercept (c) from plotStdC function is used.

Usage

```
silco(m, y, c, data, output_dir = tempdir())
```

Arguments

m	The slope value (replace with the actual slope from plotStdC).
y	The y-intercept (replace with the actual intercept from tdgraph).
c	The intercept (replace with the actual intercept from plotStdC).
data	Path to the CSV file containing output values from plotStdC.
output_dir	The directory where the output CSV file should be saved. Defaults to the temporary directory (tempdir()).

Value

A data frame with sample_id and C0 values.

Examples

```
data <- system.file("extdata", "WLO6output.csv", package = "bSi")
m <- 5.6073  # Replace with the actual slope from plotStdC
y <- 0.1234  # Replace with the actual intercept from tdgraph
c <- 0.5678  # Replace with the actual intercept from plotStdC
C0 <- silco(m, y, c, data)
```

tdgraph *Create Time-Dissolution Graphs*

Description

Create Time-Dissolution Graphs

Usage

```
tdgraph(
  data_file,
  output_plot_file,
  output_csv_file,
  label_y1 = 0.055,
  label_y2 = 0.032,
  param,
  value,
  eq.label,
  rr.label,
  .
)
```

Arguments

`data_file`	Path to the data CSV file.
`output_plot_file`	Path to save the graph as a TIFF file.
`output_csv_file`	Path to save the CSV file.
`label_y1`	Y-coordinate for the first equation label.
`label_y2`	Y-coordinate for the second equation label.
`param`	Placeholder for param variable.
`value`	Placeholder for value variable.
`eq.label`	Placeholder for eq.label variable.
`rr.label`	Placeholder for rr.label variable.
`.`	Placeholder for . variable.

Value

A data frame with the equation of the line, R-squared value, and y-intercept for each parameter.

Examples

```
data_file <- system.file("extdata", "mydata.csv", package = "bSi")
output_plot_file <- file.path(tempdir(), "plot1.tiff")
output_csv_file <- file.path(tempdir(), "output.csv")
param <- NULL
value <- NULL
eq.label <- NULL
rr.label <- NULL
. <- NULL
tdgraph(data_file, output_plot_file, output_csv_file,
        label_y1 = 0.055, label_y2 = 0.032, param, value, eq.label, rr.label, .)
```

Index