Package ‘sfc’

July 23, 2025

Type Package

Title Substance Flow Computation

Version 0.1.0

Description Provides a function sfc() to compute the substance flow with the input files --- ``data" and ``model". If sample.size is set more than 1, uncertainty analysis will be executed while the distributions and parameters are supplied in the file ``data".

Depends R (>= 3.1.0), dplyr, tidyr

Imports stats, utils, triangle, zoo, sna

License GPL

URL https://github.com/ctfysh/sfc

BugReports https://github.com/ctfysh/sfc/issues

LazyData TRUE

RoxygenNote 5.0.1

NeedsCompilation no

Author Hu Sheng [aut, cre]

Maintainer Hu Sheng <shenghu@nju.edu.cn>

Repository CRAN

Date/Publication 2016-08-25 10:01:01

Contents

sfc-package *sfc: A package for substance flow computation.*

Description

The sfc package provides an important functions sfc() to compute the substance flow with the input files — "data" and "model". If sample.size is set more than 1, uncertainty analysis will be executed while the distributions and parameters are supplied in the file "data".

sfc *Substance Flow Computation*

Description

`sfc` is the function for computing the substance flow with the input file "data" and "model", and furtherly analyzing the uncertainty of the computing results which are propagated from the activity data and parameters.

Usage

```
sfc(data, model, sample.size = 1, rand.seed = NULL, check = TRUE,
  inner = FALSE, ...)
```

Arguments

`data`	a path for a ".csv" file or a data frame which contains the basic data for substance flow computation. The details of `data` are given under "Details".
`model`	a path for a text file or a string vector which contains the formulas for substance flow computation. The details of `model` specification are given under "Details".
`sample.size`	a single value, interpreted as an integer. The details of `sample.size` are given under "Details".
`rand.seed`	a single value, interpreted as an integer or NULL, which is the same with `seed` in `set.seed`.
`check`	logical. If FALSE, the summary of uncertainty analysis will use all the samples. IF TRUE, they only use the valid samples. The details of `check` are given under "Details".
`inner`	logical. IF TRUE, the intermediate results will be returnd, such as the input samples and the results for each sample.
`...`	Further arguments to be passed to both `read.csv` and `scan`.

Details

There are two important inputs in `sfc` function: `data` and `model`. "data" is a table in the form of ".csv" which includes a line of the header and the other lines of the specific data. The header can be divided into two categories: special fields and general fields. Special fields represent the fields which are meaningful for computation, while general fields only describe each row in details which will not be used in computation, such as the description of the concept or unit of a variable. So general fields are not as necessary as special fields. Special fields can also be divided into two parts: fixed fields and flexible fields. Fixed fields contains: NAME, DATA, TIME, SITE and DIST. Flexible fields contains: P1, P2, P3, ... and C1, C2, C3, ... NAME means the variable name. DATA means the variable value. TIME means the specific time while SITE means the specific place where the variable observed. DIST is a character of distribution in `Distributions`, such as "norm", "unif", etc. P1, P2, P3, ... are the distribution parameters which are flexible because different distribution has different number or meaning of parameters. C1, C2, C3, ... are supplementary fields for other distinguished information like TIME and SITE. All the letters of field names are in the capital form to distinguish with the normal R variables or functions.

"model" is a special R code fragment which only contains flow computation expressions, like "PF[i, j,] = DX * PX" or "PF[i, j,] <- DX * PX". Here, "i" is the start node and "j" is the end node. "DX" and "PX" represent activity data and parameters, respectively. They also correspond to the "NAME" field in the "data" file. "PF[i, j,]" is an array which means the flow from "i" to "j", while "PF" is the default flow name. If you do not like coding, this package provides another alternative: to write model as into a ".csv" file. This file contains four critical fields: NAME, START, END and FUN. They are required fields while you can add other optional fields to make the model more readable freely. The NAME field is used to define flow label, e.g. "PF_01" labels "PF[i, j,]". The START and END field are used to define the start node "i" and the end node "j" in "PF[i, j,]". The last one, "FUN" is used to define the formula to calculate the flow, such as the righthand expression "DX * PX". To keep the code easy and simple, some auxiliary functions are supplied: `p`, `comp0`, `suma` and `rnone`. `p` is used to keep flow positive while the negative value will be set zero. `comp0` is used to change the value near zero to be zero. `suma` is used to sum an array at the margin `m`. `rnone` is used to sample the same value.

If `sample.size` are greater than 1, the uncertainty of substance flow computation result, i.e. "PF[i, j,]", will be evaluated. However, we will still compute the scenario when `sample.size = 1`, which represent the expected scenario (or most valid scenario), to compare with each result of each sample. If the sign of each flow are the same, the sample are valid, otherwise are invalid. So the `sample.size` in the result list is the valid sample size. To trigger the above comparation, `check` should be set TRUE.

Value

`sfc` returns a list which contains four objects: `result`, `sample.size`, `sample` and `inner.result`. `result` is a data frame of substance flow computation result. `sample.size` is the sample size of valid samples. `sample` is the data frame of the input samples. `inner.result` is the data frame of substance flow computation result for each sample.

Examples

```
library(sfc)

## model as txt
```

```
data <- system.file("extdata", "data_utf8.csv", package = "sfc")
model <- system.file("extdata", "model_utf8.txt", package = "sfc")
sfc(data, model, sample.size = 100, fileEncoding = "UTF-8")

## model as csv
data <- system.file("extdata", "data_utf8.csv", package = "sfc")
model <- system.file("extdata", "model_utf8.csv", package = "sfc")
sfc(data, model, fileEncoding = "UTF-8")
```

Index