Package: phytoclass (via r-universe)

November 14, 2024

Title Estimate Chla Concentrations of Phytoplankton Groups

Version 2.0.0

Description Determine the chlorophyll a (Chl a) concentrations of different phytoplankton groups based on their pigment biomarkers. The method uses non-negative matrix factorisation and simulated annealing to minimise error between the observed and estimated values of pigment concentrations (Hayward et al. (2023) <doi:10.1002/lom3.10541>). The approach is similar to the widely used 'CHEMTAX' program (Mackey et al. 1996) <doi:10.3354/meps144265>, but is more straightforward, accurate, and not reliant on initial guesses for the pigment to Chl a ratios for phytoplankton groups.

Imports bestNormalize, dplyr, dynamicTreeCut, ggplot2, Metrics, RcppML, stats, tidyr

License MIT + file LICENSE

Encoding UTF-8

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.1

Depends R (>= 3.8)

LazyData true

Suggests knitr, rmarkdown, testthat (>= 3.0.0)

VignetteBuilder knitr

URL https://github.com/phytoclass/phytoclass/

BugReports https://github.com/phytoclass/phytoclass/issues/

Config/testthat/edition 3

Config/pak/sysreqs make libicu-dev

Repository https://phytoclass.r-universe.dev

RemoteUrl https://github.com/phytoclass/phytoclass

RemoteRef HEAD

RemoteSha d0e781697e9a47f0b14bc077397434eb2e7b78db

Contents

`Bounded_weights` *Add weights to the data, bound at a maximum.*

Description

Add weights to the data, bound at a maximum.

Usage

```
Bounded_weights(S, weight.upper.bound = 30)
```

Arguments

`S`	Sample data matrix – a matrix of pigment samples
`weight.upper.bound`	Upper bound for weights (default is 30)

Value

A vector with upper bounds for weights

Examples

```
Bounded_weights(Sm, weight.upper.bound = 30)
```

Cluster *Cluster things*

Description

Cluster things

Usage

```
Cluster(Data, min_cluster_size)
```

Arguments

Data	S (sample) matrix
min_cluster_size	the minimum size required for a cluster

Value

A named list of length two. The first element "cluster.list" is a list of clusters, and the second element "cluster.plot" the cluster analysis object (dendogram) that can be plotted.

Examples

```
Cluster.result <- Cluster(Sm, 14)
Cluster.result$cluster.list
plot(Cluster.result$cluster.plot)
```

Fm *Fm data*

Description

Fm data

Usage

```
Fm
```

Format

Fm**:**

A data frame with 9 rows and 15 columns:

chl_c1 XX

Per XX

X19but XX ...

Source

XX

`Fp` *Fp data*

Description

Fp data

Usage

```
Fp
```

Format

`Fp`**:**
A data frame with 9 rows and 15 columns:

chl_c1 XX
Per XX
X19but XX ...

Source

XX

`Matrix_checks` *Remove any column values that average 0. Further to this, also remove phytoplankton groups from the F matrix if their diagnostic pigment isn't present.*

Description

Remove any column values that average 0. Further to this, also remove phytoplankton groups from the F matrix if their diagnostic pigment isn't present.

Usage

```
Matrix_checks(S, Fmat)
```

Arguments

`S`	Sample data matrix – a matrix of pigment samples
`Fmat`	Pigment to Chl a matrix

Value

Named list with new S and Fmat matrices

Examples

```
MC <- Matrix_checks(Sm, Fm)
Snew <- MC$Snew
```

`min_max` *min_max data*

Description

min_max data

Usage

```
min_max
```

Format

`min_max`**:**
A data frame with 76 rows and 4 columns:
class XX
Pig_Abbrev XX
min XX
max max ...

Source

XX

`NNLS_MF` *Performs the non-negative matrix factorisation for given phytoplankton pigments and pigment ratios, to attain an estimate of phytoplankton class abundances.*

Description

Performs the non-negative matrix factorisation for given phytoplankton pigments and pigment ratios, to attain an estimate of phytoplankton class abundances.

Usage

```
NNLS_MF(Fn, S, cm = NULL)
```

Arguments

Fn	Pigment to Chl a matrix
S	Sample data matrix – a matrix of pigment samples
cm	Weights for each column

Value

A list containing

1. The F matrix (pigment: Chl *a*) ratios
2. The root mean square error (RMSE)
3. The C matrix (class abundances for each group)

Examples

```
MC <- Matrix_checks(Sm,Fm)
Snew <- MC$Snew
Fnew <- MC$Fnew
cm <- Bounded_weights(Snew, weight.upper.bound = 30)
NNLS_MF(Fnew, Snew, cm)
```

---

simulated_annealing *Phytoclass - simualted annealing*

Description

This is the main phytoclass algorithm. It performs simulated annealing algorithm for S and F matrices. See the examples (Fm, Sm) for how to set up matrices, and the vignette for more detailed instructions. Different pigments and phytoplankton groups may be used.

Usage

```
simulated_annealing(
  S,
  Fmat = NULL,
  user_defined_min_max = NULL,
  do_matrix_checks = TRUE,
  niter = 500,
  step = 0.009,
  weight.upper.bound = 30,
  verbose = TRUE
)
```

Arguments

S	Sample data matrix – a matrix of pigment samples
Fmat	Pigment to Chl a matrix
user_defined_min_max	data frame with some format as min_max built-in data
do_matrix_checks	This should only be set to TRUE when using the default values. This will remove pigment columns that have column sums of 0. Set to FALSE if using customised names for pigments and phytoplankton groups
niter	Number of iterations (default is 500)
step	Step ratio used (default is 0.009)
weight.upper.bound	Upper limit of the weights applied (default value is 30).
verbose	Logical value. Output error and temperature at each iteration. Default value of TRUE

Value

A list containing

1. Fmat matrix
2. RMSE (Root Mean Square Error)
3. condition number
4. Class abundances
5. Figure (plot of results)
6. MAE (Mean Absolute Error)
7. Error

Examples

```
# Using the built-in matrices Sm and Fm
set.seed(5326)
sa.example <- simulated_annealing(Sm, Fm, niter = 5)
sa.example$Figure

#Using non-default data:
# Set up a new F matrix
Fu <- data.frame(
  Per = c(0, 0, 0, 0, 1, 0, 0, 0),
  X19but = c(0, 0, 0, 0, 0, 1, 1, 0),
  Fuco = c(0, 0, 0, 1, 0, 1, 1, 0),
  Pra = c(1, 0, 0, 0, 0, 0, 0, 0),
  X19hex = c(0, 0, 0, 0, 0, 1, 0, 0),
  Allo = c(0, 0, 1, 0, 0, 0, 0, 0),
  Zea = c(1, 1, 0, 0, 0, 0, 0, 1),
  Chl_b = c(1, 1, 0, 0, 0, 0, 0, 0),
```

```
  Tchla = c(1, 1, 1, 1, 1, 1, 1, 1)
)

rownames(Fu) <- c(
  "Prasinophytes", "Chlorophytes", "Cryptophytes"
  , "Diatoms-2", "Dinoflagellates-1",
  "Haptophytes", "Pelagophytes", "Syn"
)

#Set up a new Min_max file
Min_max <- data.frame(
  Class = c(
    "Syn", "Chlorophytes", "Chlorophytes", "Prasinophytes", "Prasinophytes",
    "Prasinophytes", "Cryptophytes", "Diatoms-2", "Diatoms-2", "Pelagophytes",
    "Pelagophytes", "Pelagophytes", "Dinoflagellates-1", "Haptophytes",
    "Haptophytes", "Haptophytes", "Haptophytes", "Diatoms-2", "Cryptophytes",
    "Prasinophytes", "Chlorophytes", "Syn", "Dinoflagellates-1", "Pelagophytes"
  ),
  Pig_Abbrev = c(
    "Zea", "Zea", "Chl_b", "Pra", "Zea", "Chl_b", "Allo", "Chl_c3",
    "Fuco", "Chl_c3", "X19but", "Fuco", "Per", "X19but", "X19hex",
    "Fuco", "Tchla", "Tchla", "Tchla", "Tchla", "Tchla", "Tchla", "Tchla",
    "Tchla"
  ),
  min = as.numeric(c(
    0.0800, 0.0063, 0.1666, 0.0642, 0.0151, 0.4993, 0.2118, 0.0189,
    0.3315, 0.1471, 0.2457, 0.3092, 0.3421, 0.0819, 0.2107, 0.0090,
    1.0000, 1.0000, 1.0000, 1.0000, 1.0000, 1.0000, 1.0000, 1.0000
  )),
  max = as.numeric(c(
    1.2123, 0.0722, 0.9254, 0.4369, 0.1396, 0.9072, 0.5479, 0.1840,
    0.9332, 0.2967, 1.0339, 1.2366, 0.8650, 0.2872, 1.3766, 0.4689,
    1.0000, 1.0000, 1.0000, 1.0000, 1.0000, 1.0000, 1.0000, 1.0000
  ))
)
#Run the new file with your own set up (make sure all names between your data (S),
#F-marix, and min_max are correct)
Results <- simulated_annealing(
  S = Sm,
  F = Fu,
  user_defined_min_max = Min_max,
  do_matrix_checks = TRUE,
  #You may want to change this to faults if your naming conventions are different.
  niter = 1,
  step = 0.01,
  weight.upper.bound = 30)
```

Sm *Sm data*

Description

Sm data

Usage

```
Sm
```

Format

`Sm`**:**
A data frame with 29 rows and 15 columns:

chl_c1 XX
Per XX
X19but XX ...

Source

XX

`Sp` *Sp data*

Description

Sp data

Usage

```
Sp
```

Format

`Sp`**:**
A data frame with 29 rows and 15 columns:

chl_c1 XX
Per XX
X19but XX ...

Source

XX

Steepest_Desc	*Stand-alone version of steepest descent algorithm. This is similar to the CHEMTAX steepest descent algorithm. It is not required to use this function, and as results are not bound by minimum and maximum, results may be unrealistic.*

Description

Stand-alone version of steepest descent algorithm. This is similar to the CHEMTAX steepest descent algorithm. It is not required to use this function, and as results are not bound by minimum and maximum, results may be unrealistic.

Usage

```
Steepest_Desc(Fmat, S, num.loops)
```

Arguments

Fmat	Pigment to Chl a matrix
S	Sample data matrix – a matrix of pigment samples
num.loops	Number of loops/iterations to perform (no default)

Value

A list containing

1. The F matrix (pigment: Chl *a*) ratios
2. RMSE (Root Mean Square Error)
3. Condition number
4. class abundances
5. Figure (plot of results)
6. MAE (Mean Absolute Error)

Examples

```
MC <- Matrix_checks(Sm,Fm)
Snew <- MC$Snew
Fnew <- MC$Fnew
SDRes <- Steepest_Desc(Fnew,Snew, num.loops = 20)
```

Index