Package ‘trifield’

February 20, 2015

Type Package
Title Some basic facilities for ternary fields and plots
Version 1.1
Date 2011-07-24
Author Tim Keitt <tkeitt@gmail.com>
Maintainer Tim Keitt <tkeitt@gmail.com>
Description The package contains routines to 1) project unity-summed
triples to unit-square doubles and vice versa, 2) make a grid
of unity-summed triples paired to doubles, 3) evaluate a
function over the grid and 4) make simple plots including
ternary contour plots over a grid of values.
License GPL-2
LazyLoad yes
Repository CRAN
Date/Publication 2012-10-29 17:24:23
NeedsCompilation no

R topics documented:

trifield-package ......................................................... 2
abc2xy ................................................................. 3
plot.trifield ............................................................ 4
ternary.apply .......................................................... 5
ternary.field ............................................................ 6
ternary.grid ............................................................. 8
ternary.legend .......................................................... 9
triaxes ................................................................. 10
tribox ................................................................. 11
xy2abc ................................................................. 12

Index 14
trifield-package

Assists in creating and plotting ternary fields

Description

The package can project between unity-summed ternary vectors and binary vectors. An "apply" function is provided to apply a function over a ternary field, as is a function for constructing ternary grids. Some rudimentary plotting functions are also provided. This is quick and dirty research code I needed in the course of writing a manuscript. Take it as such.

Details

Package: trifield
Type: Package
Version: 1.0
Date: 2011-07-24
License: GPL 2
LazyLoad: yes

Some simple routines, mainly aimed at making ternary fields and contour plots.

Author(s)

Tim Keitt <tkeitt@gmail.com>
Maintainer: Tim Keitt <tkeitt@gmail.com>

References

http://dx.doi.org/10.1016/j.ecolmodel.2012.05.020

Examples

# See demo(trifield)
## Not run:
grid.size = 128
par(mar = rep(2, 4), oma = rep(0, 4))
tg = ternary.grid(grid.size)
f = function(x)
    sin(2 * pi * x[1]) +
    sin(3 * pi * x[2]) +
    sin(4 * pi * x[3])
z = ternary.apply(tg, f)
tf = ternary.field(tg, z)
plot(tf)
ternary.legend()

## End(Not run)
abc2xy

| abc2xy | Project unity-summed triples \((a, b, c)\) to unit-square doubles \((x, y)\) |

Description

Given a matrix of ternary values, this function will compute their \(x, y\) coordinates.

Usage

```
abc2xy(abc)
```

Arguments

- `abc` A 3-column matrix containing unity-summed triples

Details

The \(x, y\) coordinates are scaled such that the altitude of the equilateral triangle defining the ternary space is equal to one.

Value

A matrix of \(x, y\) values, one per row

Author(s)

Timothy H. Keitt <tkeitt@gmail.com>

References

http://dx.doi.org/10.1016/j.ecolmodel.2012.05.020

See Also

- `xy2abc`

Examples

```r
# See demo(trifield)
## Not run:
grid.size = 128
par(mar = rep(2, 4), oma = rep(0, 4))
tg = ternary.grid(grid.size)
f = function(x)
    sin(2 * pi * x[1]) +
    sin(3 * pi * x[2]) +
    sin(4 * pi * x[3])
z = ternary.apply(tg, f)
tf = ternary.field(tg, z)
```
plot.trifield

Description

A simple wrapper around image and contour for plotting ternary fields

Usage

## S3 method for class 'trifield'
plot(x, contours = TRUE, col = topo.colors(256), lab1 = "A = 0", lab2 = "B = 0", lab3 = "C = 0", tribox

Arguments

x An object of class trifield
contours Make contours?
col Color palette to use
lab1 First axis label
lab2 Second axis label
lab3 Third axis label
tribox Draw a triangle around the plot?
axis.lines Draw internal (altitude) axis lines?
... Additional graphics parameters to be passed to plot

Details

This is a small demo function showing how one can make a ternary contour plot from a trifield object. The input object can be any list with x, y and z fields, but the result will be odd if non-NA z-values fall outside an equilateral triangle with unit altitude.

Value

None

Author(s)

Tim Keitt <tkeitt@gmail.com>

References

http://dx.doi.org/10.1016/j.ecolmodel.2012.05.020
ternary.apply

See Also

plot.default, image.default, contour.default

Examples

# See demo(trifield)
## Not run:
grid.size = 128
par(mar = rep(2, 4), oma = rep(0, 4))
tg = ternary.grid(grid.size)
f = function(x)
    sin(2 * pi * x[1]) +
    sin(3 * pi * x[2]) +
    sin(4 * pi * x[3])
z = ternary.apply(tg, f)
tf = ternary.field(tg, z)
plot(tf)
ternary.legend()
## End(Not run)

ternary.apply  Apply a function over a ternary grid

Description

Given 1) a ternary grid and 2) a function that takes a ternary value as its first argument, this function returns a vector formed by applying the function to each point in the grid.

Usage

ternary.apply(grid, f, ...)

Arguments

grid  A data frame with columns a, b and c specifying points in a ternary field
f    A function that accepts a length 3 vector as its first argument
...  Additional arguments passed to f

Details

Results will be concatenated into a vector; it is most useful for scalar-valued functions.

Value

A vector of results
ternary.field

Constructs a ternary field

Description

Constructs a trifield object, suitable for use with image and contour from a ternary grid

Usage

ternary.field(grid, vals, dim.out = NULL)

Arguments

grid A ternary grid as returned e.g. by the ternary.grid function
vals A vector of values where length(vals) == nrow(grid) as returned e.g. by the ternary.apply function
dim.out Dimensions (rows and columns) of the output

Examples

# See demo(trifield)
## Not run:
grid.size = 128
par(mar = rep(2, 4), oma = rep(0, 4))
tg = ternary.grid(grid.size)
f = function(x)
  sin(2 * pi * x[1]) +
  sin(3 * pi * x[2]) +
  sin(4 * pi * x[3])
z = ternary.apply(tg, f)
tf = ternary.field(tg, z)
plot(tf)
ternary.legend()
## End(Not run)
Details

This function constructs a trifield object suitable for plotting. It may be the case that the dimension indices contained in the ternary grid object may not contain sufficient information to determine the dimensions of the output. In that case, explicit output dimensions can be provided using the dimNout argument. If the dimNout argument is NULL, then the maximum row and column indices (xi, yi) will be extracted from the ternary grid. Note that image plots the transposition of input matrices and that dimNout must be specified with this in mind.

Value

A list with x, y and z elements defining a plotting surface suitable for use with image and contour.

Author(s)

Tim Keitt <tkeitt@gmail.com>

References

http://dx.doi.org/10.1016/j.ecolmodel.2012.05.020

See Also

ternary.grid, image.default

Examples

# See demo(trifield)
## Not run:
grid.size = 128
par(mar = rep(2, 4), oma = rep(0, 4))
tg = ternary.grid(grid.size)
f = function(x)
  sin(2 * pi * x[1]) +
  sin(3 * pi * x[2]) +
  sin(4 * pi * x[3])
z = ternary.apply(tg, f)
tf = ternary.field(tg, z)
plot(tf)
ternary.legend()

## End(Not run)
ternary.grid | Constructs a ternary grid

Description
The function builds a rectangular lattice of points bounded by an equilateral triangle with unit altitude. It can be used with `ternary.apply` and `ternary.field` to create a surface of values for plotting.

Usage
`ternary.grid(grid.size)`

Arguments
- `grid.size`: The number of rows and columns in the grid

Details
The size of the grid is the number of rows and columns in the rectangular region bounding the ternary plot region. The number of output `x`, `y` and corresponding `a`, `b`, `c` points will be less than rows times columns because many `x`, `y` points fall outside the ternary region. In all cases, the sum of `a`, `b` and `c` will be unity.

Value
Data frame with columns
- `x`: x-coordinate
- `y`: y-coordinate
- `xi`: column indices (I think)
- `yi`: row indices (I think)
- `a`: distance along first ternary axis
- `b`: distance along second ternary axis
- `c`: distance along third ternary axis

Author(s)
Tim Keitt <tkeitt@gmail.com>

References
- [http://dx.doi.org/10.1016/j.ecolmodel.2012.05.020](http://dx.doi.org/10.1016/j.ecolmodel.2012.05.020)

See Also
- `ternary.apply`, `ternary.grid`
**ternary.legend**

**Examples**

```r
# See demo(trifield)
## Not run:
grid.size = 128
par(mar = rep(2, 4), oma = rep(0, 4))
tg = ternary.grid(grid.size)
f = function(x)
  sin(2 * pi * x[1]) +
  sin(3 * pi * x[2]) +
  sin(4 * pi * x[3])
z = ternary.apply(tg, f)
tf = ternary.field(tg, z)
plot(tf)
ternary.legend()
## End(Not run)
```

---

**ternary.legend**

*Plots a ternary legend*

**Description**

Plots a legend giving the directions of the ternary axes and labels.

**Usage**

```r
ternary.legend(x0 = 0.2, y0 = 0.8, radius = 0.1, lab1 = "A", lab2 = "B", lab3 = "C", infl = 1.5, cex = 0.7)
```

**Arguments**

- `x0`: x-coordinate of the center point
- `y0`: y-coordinate of the center point
- `radius`: how long to draw the axes
- `lab1`: string giving first axis label
- `lab2`: string giving second axis label
- `lab3`: string giving third axis label
- `infl`: space between axes and labels
- `cex`: size of the labels
- `length`: length of axis lines
- `...`: additional parameters passed to `arrows`

**Value**

None
Author(s)
Tim Keitt <tkeitt@gmail.com>

References
http://dx.doi.org/10.1016/j.ecolmodel.2012.05.020

See Also
plot.trifield

Examples

# See demo(trifield)
## Not run:
grid.size = 128
par(mar = rep(2, 4), oma = rep(0, 4))
tg = ternary.grid(grid.size)
f = function(x)
    sin(2 * pi * x[1]) +
    sin(3 * pi * x[2]) +
    sin(4 * pi * x[3])
z = ternary.apply(tg, f)
tf = ternary.field(tg, z)
plot(tf)
ternary.legend()

## End(Not run)

---

triaxes  Draws ternary axis lines

Description

Draws lines separating adjacent halves of the equilateral triangle with unit altitude

Usage

triaxes(lcol = "darkgrey", lty = 2)

Arguments

lcol  line color
lty   line type

Value

None
tribox

Draws an equilateral triangle with unit altitude

Description

Like the box graphics function, this function draws a "box" around a ternary plot

Usage

tribox(...)  

Arguments

... Additional graphic parameters to be passed to polygon

Value

None

Examples

# See demo(trifield)
## Not run:
grid.size = 128
par(mar = rep(2, 4), oma = rep(0, 4))
tg = ternary.grid(grid.size)
f = function(x)
    sin(2 * pi * x[1]) +
    sin(3 * pi * x[2]) +
    sin(4 * pi * x[3])
z = ternary.apply(tg, f)
tf = ternary.field(tg, z)
plot(tf)
ternary.legend()

## End(Not run)
Author(s)
Tim Keitt <tkeitt@gmail.com>

References
http://dx.doi.org/10.1016/j.ecolmodel.2012.05.020

See Also
plot.default, plot.trifield

Examples

# See demo(trifield)
## Not run:
grid.size = 128
par(mar = rep(2, 4), oma = rep(0, 4))
tg = ternary.grid(grid.size)
f = function(x)
   sin(2 * pi * x[1]) +
   sin(3 * pi * x[2]) +
   sin(4 * pi * x[3])
z = ternary.apply(tg, f)
tr = ternary.field(tg, z)
plot(tr)
ternary.legend()

## End(Not run)

---

xy2abc  Ternary projection of x, y coordinates

Description

Projects x, y coordinates on the unit square to a, b, c ternary ordinates

Usage

xy2abc(xy)

Arguments

xy  A matrix of coordinates, one per row, on the unit square

Value

A matrix of a, b, c triples
Author(s)

Tim Keitt <tkeitt@gmail.com>

References

http://dx.doi.org/10.1016/j.ecolmodel.2012.05.020

See Also

abc2xy

Examples

# See demo(trifield)
## Not run:
grid.size = 128
par(mar = rep(2, 4), oma = rep(0, 4))
tg = ternary.grid(grid.size)
f = function(x)
    sin(2 * pi * x[1]) +
    sin(3 * pi * x[2]) +
    sin(4 * pi * x[3])
z = ternary.apply(tg, f)
tf = ternary.field(tg, z)
plot(tf)
ternary.legend()

## End(Not run)
Index

*Topic **hplot**
  plot.trifield, 4
  ternary.legend, 9
  triaxes, 10
  tribox, 11

*Topic **manip**
  abc2xy, 3
  ternary.apply, 5
  ternary.field, 6
  ternary.grid, 8
  xy2abc, 12

*Topic **package**
  trifield-package, 2

abc2xy, 3, 13
arrows, 9

box, 11

contour, 6, 7
contour.default, 5

image, 6, 7
image.default, 5, 7

plot.default, 5, 11, 12
plot.trifield, 4, 10–12
polygon, 11

ternary.apply, 5, 6, 8
ternary.field, 6, 6, 8
ternary.grid, 6–8, 8
ternary.legend, 9
triaxes, 10
tribox, 11
trifield(trifield-package), 2
trifield-package, 2

xy2abc, 3, 12