Package ‘biogeom’

December 22, 2022

Type Package

Title Biological Geometries

Version 1.3.5

Date 2022-12-21

Author Peijian Shi [aut, cre],
Johan Gielis [aut],
Brady K. Quinn [aut]

Maintainer Peijian Shi `<pjshi@njfu.edu.cn>`

Imports spatstat.geom (>= 2.4-0)

Description Is used to simulate and fit biological geometries. 'biogeom' incorporates several novel universal parametric equations that can generate the profiles of bird eggs, flowers, linear and lanceolate leaves, seeds, starfish, and tree-rings (Gielis (2003) <doi:10.3732/ajb.90.3.333>; Shi et al. (2020) <doi:10.3390/sym12040645>), three growth-rate curves representing the ontogenetic growth trajectories of animals and plants against time, and the axially symmetrical and integral forms of all these functions (Shi et al. (2017) <doi:10.1016/j.ecolmodel.2017.01.012>; Shi et al. (2021) <doi:10.3390/sym13081524>). The optimization method proposed by Nelder and Mead (1965) <doi:10.1093/comjnl/7.4.308> was used to estimate model parameters. 'biogeom' includes several real data sets of the boundary coordinates of natural shapes, including avian eggs, fruit, lanceolate and ovate leaves, tree rings, seeds, and sea stars,and can be potentially applied to other natural shapes. 'biogeom' can quantify the conspecific or interspecific similarity of natural outlines, and provides information with important ecological and evolutionary implications for the growth and form of living organisms. Please see Shi et al. (2022) <doi:10.1111/nyas.14862> for details.

Depends R (>= 4.2.0)

License GPL (>= 2)

NeedsCompilation no

Repository CRAN

Date/Publication 2022-12-22 13:10:08 UTC

R topics documented:

adjdata . . . 3
areaGE . . . 5
areaovate . . . 7
bambooleaves . . . 9
bilat . . . 10
biogeom . . . 15
curveEPE . . . 18
curveETE . . . 20
curveGE . . . 22
curveNRGE . . . 25
curveovate . . . 26
DEPE . . . 28
DETE . . . 30
DNRGE . . . 31
DSGE . . . 33
eggs . . . 34
EPE . . . 36
ETE . . . 38
fitEPE . . . 39
fitETE . . . 43
fitGE . . . 46
fitNRGE . . . 51
fitovate . . . 55
fitsigmoid . . . 58
fracdim . . . 61
GE . . . 63
ginkgoseed . . . 66
kp . . . 67
lmPE . . . 68
lmTE . . . 72
MbetaE . . . 76
MBriereE . . . 79
MLRFE . . . 81
Neocinnamomum . . . 83
NRGE . . . 84
PE . . . 86
shoots . . . 89
sigmoid . . . 90
starfish . . . 92
SurfaceAreaEPE . . . 93
SurfaceAreaETE . . . 94
SurfaceAreaNRGE . . . 95
SurfaceAreaSGE . . . 97
TE . . . 98
TGE . . . 100
TSE . . . 102

veins 104
VolumeEPE 105
VolumeETE 107
VolumeNRGE 108
VolumeSGE 110
whitespruce 111

Index **114**

`adjdata` *Boundary Data Adjustment of A Polygon*

Description

`adjdata` adjusts the data points in counterclockwise order based on the shortest distance method.

Usage

```
adjdata(x, y, ub.np = 2000, times = 1.2, len.pro = 1/20, index.sp = 1)
```

Arguments

`x`	the x coordinates of points on a polygon's boundary.
`y`	the y coordinates of points on a polygon's boundary.
`ub.np`	the upper bound of the number of points eventually retained on the polygon's boundary.
`times`	the number of times `un.np` is initially retained as the number of points.
`len.pro`	the proportion of the distance between any two points to the maximum distance between the points on the polygon's boundary, which is used to determine whether the second point needs to be deleted.
`index.sp`	the index of the starting point of a group of indices that regularly divide the number of points on the polygon's boundary into `ub.np` parts.

Details

When `ub.np` > `length(x)`, `length(x)` points on the polygon's boundary are retained. The `quantile` function in package **stats** is used to carry out the regular division of data points. From the starting point, the second point is the one that has the shortest distance from the former. When the distance between the two points is larger than `len.pro` multiplied by the maximum distance between points on the polygon's boundary, the second point is deleted from the coordinates. Then, the third point that has the shortest distance from the first point is defined as the second point. If the distance between the first point and the second point is no more than `len.pro` multiplied by the maximum distance, the first and second points are recorded in a new matrix for the coordinates of the polygon, and the second point is defined as the first point in the old matrix for the coordinates of the polygon. The shortest distance method is then used to look for a third point that meets the requirement.

Value

x	the x coordinates of points eventually retained on the polygon's boundary.
y	the y coordinates of points eventually retained on the polygon's boundary.

Note

The initial boundary data of a polygon can be obtained by running the M-file based on Matlab (version >= 2009a) developed by Shi et al. (2018) and Su et al. (2019) for a .bmp black and white image of the polygon. See references below.

Author(s)

Peijian Shi <`pjshi@njfu.edu.cn`>, Johan Gielis <`johan.gielis@uantwerpen.be`>, Brady K. Quinn <`Brady.Quinn@dfo-mpo.gc.ca`>.

References

Shi, P., Gielis, J., Quinn, B.K., Niklas, K.J., Ratkowsky, D.A., Schrader, J., Ruan, H., Wang, L., Niinemets, Ü. (2022) 'biogeom': An R package for simulating and fitting natural shapes. *Annals of the New York Academy of Sciences* 1516, 123−134. doi:10.1111/nyas.14862

Shi, P., Ratkowsky, D.A., Li, Y., Zhang, L., Lin, S., Gielis, J. (2018) General leaf-area geometric formula exists for plants - Evidence from the simplified Gielis equation. *Forests* 9, 714. doi:10.3390/f9110714

Su, J., Niklas, K.J., Huang, W., Yu, X., Yang, Y., Shi, P. (2019) Lamina shape does not correlate with lamina surface area: An analysis based on the simplified Gielis equation. *Global Ecology and Conservation* 19, e00666. doi:10.1016/j.gecco.2019.e00666

Examples

```
data(eggs)
uni.C1 <- sort( unique(eggs$Code) )
ind1   <- 2
Data1  <- eggs[eggs$Code==uni.C1[ind1], ]
x0     <- Data1$x
y0     <- Data1$y

Res1   <- adjdata(x0, y0, ub.np=2000, times=1.2, len.pro=1/20)
x1     <- Res1$x
y1     <- Res1$y

dev.new()
plot( x1, y1, asp=1, cex.lab=1.5, cex.axis=1.5,
      xlab=expression(italic("x")), ylab=expression(italic("y")),
      pch=1, col=1 )

Res2   <- adjdata(x0, y0, ub.np=40, times=1, len.pro=1/2, index.sp=20)
x2     <- Res2$x
y2     <- Res2$y

Res3   <- adjdata(x0, y0, ub.np=100, times=1, len.pro=1/2, index.sp=100)
```

```
x3     <- Res3$x
y3     <- Res3$y

dev.new()
plot( x2, y2, asp=1, cex.lab=1.5, cex.axis=1.5,
      xlab=expression(italic("x")), ylab=expression(italic("y")),
      pch=1, col=4 )
points( x3, y3, col=2)


data(starfish)

uni.C2 <- sort( unique(starfish$Code) )
ind2   <- 2
Data2  <- starfish[starfish$Code==uni.C2[ind2], ]
x4     <- Data2$x
y4     <- Data2$y

dev.new()
plot( x4, y4, asp=1, type="l", cex.lab=1.5, cex.axis=1.5,
      xlab=expression(italic("x")), ylab=expression(italic("y")) )

Res4 <- adjdata(x4, y4, ub.np=500, times=1.2, len.pro=1/20)
x5   <- Res4$x
y5   <- Res4$y

dev.new()
plot( x5, y5, asp=1, type="l", cex.lab=1.5, cex.axis=1.5,
      xlab=expression(italic("x")), ylab=expression(italic("y")) )

graphics.off()
```

`areaGE` *Area Calculation for the Gielis Curve Within* $[0, 2\pi)$

Description

`areaGE` is used to calculate the area of the polygon generated by the Gielis curve within $[0, 2\pi)$.

Usage

```
areaGE(expr, P, m = 1, simpver = NULL,
       nval = 1, subdivisions = 100L,
       rel.tol = .Machine$double.eps^0.25,
       abs.tol = rel.tol, stop.on.error = TRUE,
       keep.xy = FALSE, aux = NULL)
```

Arguments

`expr`	the original (or twin) Gielis equation or one of its simplified versions.
`P`	the parameters of the original (or twin) Gielis equation or one of its simplified versions.
`m`	the given m value that determines the number of angles of the Gielis curve within $[0, 2\pi)$.
`simpver`	an optional argument to use the simplified version of the original (or twin) Gielis equation.
`nval`	the specified value for n_1 or n_2 or n_3 in the simplified versions.
`subdivisions`	please see the arguments for the `integrate` function in package **stats**.
`rel.tol`	please see the arguments for the `integrate` function in package **stats**.
`abs.tol`	please see the arguments for the `integrate` function in package **stats**.
`stop.on.error`	please see the arguments for the `integrate` function in package **stats**.
`keep.xy`	please see the arguments for the `integrate` function in package **stats**.
`aux`	please see the arguments for the `integrate` function in package **stats**.

Details

The arguments of `P`, `m`, `simpver`, and `nval` should correspond to `expr` (i.e., `GE` or `TGE`). Please note the differences in the simplified version number and the number of parameters between `GE` and `TGE`.

Value

The area of the polygon within $[0, 2\pi)$ generated by the original (or twin) Gielis equation or one of its simplified versions.

Note

`simpver` in `GE` is different from that in `TGE`.

Author(s)

Peijian Shi <`pjshi@njfu.edu.cn`>, Johan Gielis <`johan.gielis@uantwerpen.be`>, Brady K. Quinn <`Brady.Quinn@dfo-mpo.gc.ca`>.

References

Gielis, J. (2003) A generic geometric transformation that unifies a wide range of natural and abstract shapes. *American Journal of Botany* 90, 333−338. doi:10.3732/ajb.90.3.333

Li, Y., Quinn, B.K., Gielis, J., Li, Y., Shi, P. (2022) Evidence that supertriangles exist in nature from the vertical projections of *Koelreuteria paniculata* fruit. *Symmetry* 14, 23. doi:10.3390/sym14010023

Shi, P., Gielis, J., Quinn, B.K., Niklas, K.J., Ratkowsky, D.A., Schrader, J., Ruan, H., Wang, L., Niinemets, Ü. (2022) 'biogeom': An R package for simulating and fitting natural shapes. *Annals of the New York Academy of Sciences* 1516, 123−134. doi:10.1111/nyas.14862

Shi, P., Ratkowsky, D.A., Gielis, J. (2020) The generalized Gielis geometric equation and its application. *Symmetry* 12, 645. doi:10.3390/sym12040645

Shi, P., Xu, Q., Sandhu, H.S., Gielis, J., Ding, Y., Li, H., Dong, X. (2015) Comparison of dwarf bamboos (*Indocalamus* sp.) leaf parameters to determine relationship between spatial density of plants and total leaf area per plant. *Ecology and Evolution* 5, 4578−4589. doi:10.1002/ece3.1728

See Also

`curveGE`, `fitGE`, `GE`, `TGE`

Examples

```
Para1 <- c(1.7170, 5.2258, 7.9802)
areaGE(GE, P = Para1, m=5, simpver=1)

Para2 <- c(2.1066, 3.5449, 0.4619, 10.5697)
areaGE(TGE, P = Para2, m=5, simpver=1)
```

`areaovate` *Area Calculation for an Ovate Polygon*

Description

`areaovate` is used to calculate the area of an ovate polygon made from combing two symmetrical curves generated by a performance equation (e.g., `MLRFE`).

Usage

```
areaovate(expr, P, simpver = NULL,
          subdivisions = 100L,
          rel.tol = .Machine$double.eps^0.25,
          abs.tol = rel.tol, stop.on.error = TRUE,
          keep.xy = FALSE, aux = NULL)
```

Arguments

`expr`	a performance equation or one of its simplified versions.
`P`	the parameters of the performance equation or one of its simplified versions.
`simpver`	an optional argument to use the simplfied version of the performance equation.
`subdivisions`	please see the arguments for the `integrate` function in package **stats**.
`rel.tol`	please see the arguments for the `integrate` function in package **stats**.
`abs.tol`	please see the arguments for the `integrate` function in package **stats**.
`stop.on.error`	please see the arguments for the `integrate` function in package **stats**.
`keep.xy`	please see the arguments for the `integrate` function in package **stats**.
`aux`	please see the arguments for the `integrate` function in package **stats**.

Details

The performance equations denote `MbetaE`, `MBriereE`, `MLRFE`, and their simplified versions. The arguments of `P` and `simpver` should correspond to `expr` (i.e., `MbetaE` or `MBriereE` or `MLRFE`).

Value

The area of two symmetrical curves along the x-axis generated by a performance equation or one of its simplified versions.

Note

Here, the user can define other performance equations, but new equations or their simplified versions should include the lower and upper thresholds in the x-axis corresponding to $y = 0$, whose indices should be the same as those in `MbetaE` or `MBriereE` or `MLRFE`.

Author(s)

Peijian Shi <`pjshi@njfu.edu.cn`>, Johan Gielis <`johan.gielis@uantwerpen.be`>, Brady K. Quinn <`Brady.Quinn@dfo-mpo.gc.ca`>.

References

Jin, J., Quinn, B.K., Shi, P. (2022) The modified Brière equation and its applications. *Plants* 11, 1769. doi:10.3390/plants11131769

Shi, P., Fan, M., Ratkowsky, D.A., Huang, J., Wu, H., Chen, L., Fang, S., Zhang, C. (2017) Comparison of two ontogenetic growth equations for animals and plants. *Ecological Modelling* 349, 1−10. doi:10.1016/j.ecolmodel.2017.01.012

Shi, P., Gielis, J., Quinn, B.K., Niklas, K.J., Ratkowsky, D.A., Schrader, J., Ruan, H., Wang, L., Niinemets, Ü. (2022) 'biogeom': An R package for simulating and fitting natural shapes. *Annals of the New York Academy of Sciences* 1516, 123−134. doi:10.1111/nyas.14862

Shi, P., Yu, K., Niklas, K.J., Schrader, J., Song, Y., Zhu, R., Li, Y., Wei, H., Ratkowsky, D.A. (2021) A general model for describing the ovate leaf shape. *Symmetry* 13, 1524. doi:10.3390/sym13081524

See Also

`curveovate`, `fitovate`, `MbetaE`, `MBriereE`, `MLRFE`, `sigmoid`

Examples

```
Par1 <- c(1.8175, 2.7795, 7.1557, 1.6030)
areaovate(MbetaE, P = Par1, simpver = 1)

Par2 <- c(0.0550, 0.3192, 7.1965, 0.5226)
areaovate(MBriereE, P = Par2, simpver = 1)

Par3 <- c(1.8168, 2.7967, 7.2623, 0.9662)
areaovate(MLRFE, P = Par3, simpver = 1)
```

`bambooleaves` *Leaf Boundary Data of* Phyllostachys incarnata *T. H. Wen (Poaceae: Bambusoideae)*

Description

The data consist of the boundary data of six leaves of *P. incarnata* sampled at Nanjing Forestry University campus in early December 2016.

Usage

```
data(bambooleaves)
```

Details

In the data set, there are three columns of variables: `Code`, `x`, and `y`. `Code` saves the codes of individual leaves; `x` saves the x coordinates of the leaf boundary in the Cartesian coordinate system (cm); and `y` saves the y coordinates of the leaf boundary in the Cartesian coordinate system (cm).

References

Lin, S., Shao, L., Hui, C., Song, Y., Reddy, G.V.P., Gielis, J., Li, F., Ding, Y., Wei, Q., Shi, P. (2018) Why does not the leaf weight-area allometry of bamboos follow the 3/2-power law? *Frontiers in Plant Science* 9, 583. doi:10.3389/fpls.2018.00583

Shi, P., Ratkowsky, D.A., Li, Y., Zhang, L., Lin, S., Gielis, J. (2018) General leaf-area geometric formula exists for plants - Evidence from the simplified Gielis equation. *Forests* 9, 714. doi:10.3390/f9110714

Examples

```
data(bambooleaves)

uni.C <- sort( unique(bambooleaves$Code) )
ind   <- 1
Data  <- bambooleaves[bambooleaves$Code==uni.C[ind], ]
x0    <- Data$x
y0    <- Data$y

dev.new()
plot( x0, y0, asp=1, type="l", cex.lab=1.5, cex.axis=1.5,
      xlab=expression(italic("x")), ylab=expression(italic("y")) )
length(x0)

Res1 <- adjdata(x0, y0, ub.np=600, len.pro=1/20)
dev.new()
plot( Res1$x, Res1$y, asp=1, cex.lab=1.5, cex.axis=1.5, type="l",
      xlab=expression(italic("x")), ylab=expression(italic("y")) )

graphics.off()
```

`bilat` *Measure of the Extent of Bilateral Symmetry of A Polygon*

Description

`bilat` is used to measure the extent of bilateral (a)symmetry and other measures for a polygon (e.g., a leaf).

Usage

```
bilat(x, y, strip.num = 200, peri.np = NULL, n.loop = 60,
      auto.search = TRUE, animation.fig = TRUE, time.interval = 0.001,
      unit = "cm", main = NULL, diff.fig = TRUE, angle = NULL,
      ratiox = 0.02, ratioy = 0.08, fd.opt = TRUE, frac.fig = TRUE,
      denomi.range = seq(8, 30, by = 1))
```

Arguments

`x`	the x coordinates of a polygon's boundary.
`y`	the y coordinates of a polygon's boundary.
`strip.num`	the number of equidistant strips intersecting with the polygon that are horizontally placed. See Shi et al. (2018, 2020) for details.
`peri.np`	the number of data points on the boundary retained for calculating the perimeter of the polygon.
`n.loop`	the number of data points to randomly sample for calculating the mean perimeter of the polygon.
`auto.search`	an optional argument to automatically search the maximum distance between two points on the polygon's boundary.
`animation.fig`	the option of showing the data points on the polygon's boundary in an animation.
`time.interval`	the time interval at which to suspend execution, in seconds.
`unit`	the units of the x-axis and the y-axis when showing the polygon.
`main`	the main title of the figure.
`diff.fig`	an optional argument to draw the differences in areas between the intersections of the strips with the upper part of the polygon and the intersections of the strips with the lower part of the polygon. The polygon is divided into the upper and lower parts by the x-axis. See Shi et al. (2018, 2020) for details.
`angle`	the angle between the major axis (i.e., the leaf length axis) and the x-axis, which can be defined by the user.
`fd.opt`	An optional argument to use the box-counting method to calculate the fractal dimension of the polygon's boundary on a log-log scale.
`ratiox`	the x coordinate of the location parameter for positioning the legend in the plot of the linear fitting.

`ratioy`	the y coordinate of the location parameter for positioning the legend in the plot of the linear fitting.
`frac.fig`	an optional argument to draw the results of the linear fitting using the box-counting method to calculate the fractal dimension of the polygon's boundary on a log-log scale.
`denomi.range`	the number of equidistant segments of the maximum range between the range of the x coordinates and that of the y coordinates.

Details

The data of `x` and `y` should be the coordinates adjusted using the `adjdata` function. If `peri.np = NULL`, the number of `length(x)` is used to calculate the perimeter of the polygon; if `peri.np` is a positive integer, the number of data points retained on the polygon's boundary is equal to `peri.np` and random sampling for retaining `peri.np` data points is carried out `n.loop` times for calculating the mean perimeter of the polygon. That is to say, the final output for the perimeter is the mean of the `n.loop` perimeters (i.e., replicates). If the user wants to get a consistent result for the mean perimeter, the `set.seed` function can be used. In addition, if `length(x) < peri.np`, `peri.np` then becomes `length(x)` rather than the specified value in Arguments. If the polygon apparently has a major axis (e.g., the leaf length axis for an ovate leaf), `auto.search` is appropriate. If the major axis of the polygon is not the straight line through two points on the polygon's boundary having the maximum distance, the user can define the major axis using the `locator` function in **graphic** by clicking two points on or near the polygon's boundary. The location of the first click should be northeast of the location of the second click. This means that the angle between the straight line through the locations of the two clicks and the x-axis should range from 0 to π/2. The locations of the clicks can be on the boundary or be approximate to the boundary. The function will automatically find the nearest data point on the boundary to the location of each click. When `angle = NULL`, the observed polygon will be shown at its initial angle in the scanned image; when `angle` is a numerical value (e.g., $\pi/4$) defined by the user, it indicates that the major axis is rotated $\pi/4$ counterclockwise from the x-axis.

Value

`x`	the x coordinates retained on the polygon's boundary.
`y`	the y coordinates retained on the polygon's boundary.
`phi`	the angle between the length axis (i.e., the major axis) of the polygon and the x-axis.
`n1`	the number of data points on the upper boundary of the polygon.
`n2`	the number of data points on the lower boundary of the polygon.
`n`	the number of data points on the whole polygon's boundary.
`total.poly`	an object of class "ppp" representing a point pattern dataset in the two-dimensional plane, representing the polygon's boundary.
`upper.poly`	an object of class "ppp" representing a point pattern dataset in the two-dimensional plane, representing the upper boundary of the polygon along the x-axis.
`lower.poly`	an object of class "ppp" representing a point pattern dataset in the two-dimensional plane, representing the lower boundary of the polygon along the x-axis.
`D`	the differences in areas between the upper and lower boundaries of the polygon.

`par.upper.area`	the area of the upper boundary of the polygon along the x-axis.
`par.lower.area`	the area of the lower boundary of the polygon along the x-axis.
`SI`	the standardized index for bilateral (a)symmetry for the polygon.
`AR`	the ratio of the areas of the upper to the lower parts of the polygon.
`scan.length`	the length of the polygon. The default is the maximum distance between two points on the polygon's boundary.
`scan.width`	the maximum width of the polygon.
`scan.area`	the area of the polygon.
`scan.perimeter`	the perimeter of the polygon based on all data points or a mean of `n.loop` replicates of perimeters using the `peri.np` data points retained on the polygon's boundary.
`x.width`	distance from the base to a point on the major axis associated with the maximum width of the polygon.
`width.1e`	the width associated with 1/8 of `scan.length` (starting from the base of the polygon).
`width.2e`	the width associated with 2/8 of `scan.length` (starting from the base of the polygon).
`width.4e`	the width associated with 4/8 of `scan.length` (starting from the base of the polygon).
`width.6e`	the width associated with 6/8 of `scan.length` (starting from the base of the polygon).
`width.7e`	the width associated with 7/8 of `scan.length` (starting from the base of the polygon).
`bi.test`	the testing results for `D` using the Wilcoxon signed rank test with continuity correction. See the `wilcox.test` function in **stats**.
`a`	the estimate of the intercept obtained using the box-counting method to calculate the fractal dimension of the polygon's boundary.
`sd.a`	the standard deviation of the estimated intercept.
`lci.a`	the lower bound of the 95% confidence interval of the estimated intercept.
`uci.a`	the upper bound of the 95% confidence interval of the estimated intercept.
`b`	the estimate of the slope obtained using the box-counting method to calculate the fractal dimension of the polygon's boundary.
`sd.b`	the standard deviation of the estimated slope.
`lci.a`	the lower bound of the 95% confidence interval of the estimated slope.
`uci.a`	the upper bound of the 95% confidence interval of the estimated slope.
`r.sq`	the coefficient of determination obtained when using the box-counting method to calculate the fractal dimension of the polygon's boundary.
`delta`	the vector of box sizes used in the box-counting method to calculate the fractal dimension of the polygon's boundary.
`N`	the number of boxes that include at least one pixel of the polygon's boundary.

Note

The polygon is expected to have an apparent major axis (e.g., the straight line through two points on the polygon's boundary having the maximum distance or one that can be clearly defined to pass by two landmarks on the polygon's boundary [i.e., the leaf length axis, the egg length axis, etc.]). The polygon is placed with its major axis overlapping the x-axis; the base of the polygon is located at the origin; the apex of the polygon is located to the right of the base. `phi` is equal to `angle` when `angle` is not null. In theory, `n1` + `n2` = `n`, but in most cases `n1` + `n2` is slightly smaller than `n`. The reason is that very few boundary points fall outside the the lower and upper boundaries of the polygon when using the `intersect.owin` function in **spatstat.geom**. However, this does not considerably affect the results. The log-transformed `SI` and the log-transformed `AR` are demontrated to have a more symmetrical frequency distribution than their original forms. This is important when performing an analysis of variance between (or among) groups to compared their extents of bilateral (a)symmetry. See Shi et al. (2020) for details. The box-counting approach uses a group of boxes (squares for simplicity) with different sizes (δ) to divide the leaf vein image into different parts. Let N represent the number of boxes that include at least one pixel of the polygon's boundary. The maximum of the range of the x coordinates and the range of the y coordinates for the pixels of the polygon's boundary is defined as z. Let δ represent the vector of z/`denomi.range`. We then used the following equation to calculate the fractal dimension of the polygon's boundary:

$$\ln N = a + b \ln \left(\delta^{-1}\right),$$

where b is the theoretical value of the fractal dimension. We can use its estimate as the numerical value of the fractal dimension for the polygon's boundary.

Author(s)

Peijian Shi <`pjshi@njfu.edu.cn`>, Johan Gielis <`johan.gielis@uantwerpen.be`>, Brady K. Quinn <`Brady.Quinn@dfo-mpo.gc.ca`>.

References

Shi, P., Gielis, J., Quinn, B.K., Niklas, K.J., Ratkowsky, D.A., Schrader, J., Ruan, H., Wang, L., Niinemets, Ü. (2022) 'biogeom': An R package for simulating and fitting natural shapes. *Annals of the New York Academy of Sciences* 1516, 123−134. doi:10.1111/nyas.14862

Shi, P., Niinemets, Ü., Hui, C., Niklas, K.J., Yu, X., Hölscher, D. (2020) Leaf bilateral symmetry and the scaling of the perimeter vs. the surface area in 15 vine species. *Forests* 11, 246. doi:10.3390/f11020246

Shi, P., Zheng, X., Ratkowsky, D.A., Li, Y., Wang, P., Cheng, L. (2018) A simple method for measuring the bilateral symmetry of leaves. *Symmetry* 10, 118. doi:10.3390/sym10040118

See Also

`adjdata`, `fracdim`

Examples

```
data(bambooleaves)

uni.C <- sort( unique(bambooleaves$Code) )
```

```
ind   <- 3
Data  <- bambooleaves[bambooleaves$Code==uni.C[ind], ]
x0    <- Data$x
y0    <- Data$y

dev.new()
plot( x0, y0, asp=1, type="l", cex.lab=1.5, cex.axis=1.5,
      xlab=expression(italic("x")), ylab=expression(italic("y")) )

Res1 <- adjdata(x0, y0, ub.np=2000, len.pro=1/20)
x1   <- Res1$x
y1   <- Res1$y



  Res2 <- bilat( x=x1, y=y1, time.interval=0.00045,
                 peri.np=NULL, auto.search=TRUE,
                 fd.opt=TRUE )
  Res2$scan.perimeter

  set.seed(123)
  Res3 <- bilat( x=x1, y=y1, time.interval=0.00045,
                 peri.np=500, n.loop=30,
                 auto.search=TRUE, fd.opt=FALSE )
  Res3$scan.perimeter

  set.seed(123)
  Res4 <- bilat( x=x1, y=y1, time.interval=0.00045,
                 peri.np=500, n.loop=30,
                 auto.search=TRUE, fd.opt=FALSE, angle=pi/4 )
  Res4$scan.perimeter

  set.seed(123)
  Res5 <- bilat( x=x1, y=y1, time.interval=0.00045,
                 peri.np=500, n.loop=30,
                 auto.search=TRUE, fd.opt=FALSE, angle=0 )
  Res5$scan.perimeter

  if(interactive()){
    # The angle between the leaf length axis (namely the straight
    # line through the leaf apex and base) and the horizontal axis
    # should be between 0 and pi/2 for a scanned leaf's profile.
    # Here, the user needs to first click the leaf apex,
    # and then click the leaf base.
    set.seed(123)
    Res6 <- bilat( x=x1, y=y1, time.interval=0.00045,
                 peri.np=500, n.loop=30,
                 auto.search=FALSE, fd.opt=FALSE, angle=NULL )
    Res6$scan.perimeter
  }

  set.seed(NULL)
```

```
graphics.off()
```

`biogeom` *Biological Geometries*

Description

Is used to simulate and fit biological geometries. 'biogeom' incorporates several novel universal parametric equations that can generate the profiles of bird eggs, flowers, linear and lanceolate leaves, seeds, starfish, and tree-rings (Gielis, 2003; Shi et al., 2020), three growth-rate curves representing the ontogenetic growth trajectories of animals and plants against time, and the axially symmetrical and integral forms of all these functions (Shi et al., 2017, 2021). The optimization method proposed by Nelder and Mead (1965) was used to estimate model parameters. 'biogeom' includes several real data sets of the boundary coordinates of natural shapes, including avian eggs, fruit, lanceolate and ovate leaves, tree rings, seeds, and sea stars,and can be potentially applied to other natural shapes. 'biogeom' can quantify the conspecific or interspecific similarity of natural outlines, and provides information with important ecological and evolutionary implications for the growth and form of living organisms. Please see Shi et al. (2022) for details.

Details

The DESCRIPTION file:

Package:	biogeom
Type:	Package
Title:	Biological Geometries
Version:	1.3.5
Date:	2022-12-21
Authors@R:	c(person(given="Peijian", family="Shi", email="pjshi@njfu.edu.cn", role=c("aut", "cre")), person(giver
Author:	Peijian Shi [aut, cre], Johan Gielis [aut], Brady K. Quinn [aut]
Maintainer:	Peijian Shi <pjshi@njfu.edu.cn>
Imports:	spatstat.geom (>= 2.4-0)
Description:	Is used to simulate and fit biological geometries. 'biogeom' incorporates several novel universal parame
Depends:	R (>= 4.2.0)
License:	GPL (>= 2)
NeedsCompilation:	no

Index of help topics:

`DEPE`	`Calculation of the First-Order Derivative of the Explicit Preston Equation`
`DETE`	`Calculation of the First-Order Derivative of the Explicit Troscianko Equation`
`DNRGE`	`Calculation of the First-Order Derivative of the Narushin-Romanov-Griffin Equation`

DSGE	Calculation of the First-Order Derivative of the Simplified Gielis Equation
EPE	Calculation of the Ordinate For an Arbitrary Point on the Preston Curve in the Plane
ETE	Calculation of the Ordinate For an Arbitrary Point on the Troscianko Curve in the Plane
GE	Calculation of the Polar Radius of the Gielis Curve
MBriereE	Modified Briere Equation
MLRFE	Modified Lobry-Rosso-Flandrois (LRF) Equation
MbetaE	Modified Beta Equation
NRGE	The Narushin-Romanov-Griffin Equation (NRGE)
Neocinnamomum	Leaf Boundary Data of Seven Species of _Neocinnamomum_
PE	Calculation of the Abscissa, Ordinate and Distance From the Origin For an Arbitrary Point on the Preston Curve
SurfaceAreaEPE	Calculation of the Surface Area of An Egg Based on the Explicit Preston Equation
SurfaceAreaETE	Calculation of the Surface Area of An Egg Based on the Explicit Troscianko Equation
SurfaceAreaNRGE	Calculation of the Surface Area of An Egg Based on the Narushin-Romanov-Griffin Equation
SurfaceAreaSGE	Calculation of the Surface Area of An Egg Based on the Simplified Gielis Equation
TE	The Troscianko Equation (TE)
TGE	Calculation of the Polar Radius of the Twin Gielis Curve
TSE	The Todd-Smart Equation (TSE)
VolumeEPE	Calculation of the Volume of An Egg Based on the Explicit Preston Equation
VolumeETE	Calculation of the Volume of An Egg Based on the Explicit Troscianko Equation
VolumeNRGE	Calculation of the Volume of An Egg Based on the Narushin-Romanov-Griffin Equation
VolumeSGE	Calculation of the Volume of An Egg Based on the Simplified Gielis Equation
adjdata	Boundary Data Adjustment of A Polygon
areaGE	Area Calculation for the Gielis Curve Within [0, 2pi)
areaovate	Area Calculation for an Ovate Polygon
bambooleaves	Leaf Boundary Data of _Phyllostachys incarnata_ T. H. Wen (Poaceae: Bambusoideae)
bilat	Measure of the Extent of Bilateral Symmetry of A Polygon
biogeom	Biological Geometries
curveEPE	Drawing the Preston Curve Produced by the the Explicit Preston Equation

curveETE	Drawing the Troscianko Curve Produced by the Explicit Troscianko Equation
curveGE	Drawing the Gielis Curve
curveNRGE	Drawing the Egg Shape Predicted by the Narushin-Romanov-Griffin Equation
curveovate	Drawing the Ovate Leaf-Shape Curve
eggs	Egg Boundary Data of Nine Species of Birds
fitEPE	Data-Fitting Function for the Explicit Preston Equation
fitETE	Data-Fitting Function for the Explicit Troscianko Equation
fitGE	Data-Fitting Function for the Gielis Equation
fitNRGE	Parameter Estimation for the Narushin-Romanov-Griffin Equation
fitovate	Data-Fitting Function for the Ovate Leaf-Shape Equation
fitsigmoid	Data-Fitting Function for the Sigmoid Growth Equation
fracdim	Calculation of Fractal Dimension of Lef Veins Based on the Box-Counting Method
ginkgoseed	Boundary Data of the Side Projections of _Ginkgo biloba_ Seeds
kp	Boundary Data of the Vertical Projections of _Koelreuteria paniculata_ Fruit
lmPE	Parameter Estimation for the Todd-Smart Equation
lmTE	Parameter Estimation for the Troscianko Equation
shoots	Height Growth Data of Bamboo Shoots
sigmoid	Sigmoid Growth Equation
starfish	Boundary Data of Eight Sea Stars
veins	Leaf Vein Data of _Michelia compressa_
whitespruce	Planar Coordinates of _Picea glauca_ Tree Rings

Note

We are deeply thankful to Cang Hui, Yang Li, Uwe Ligges, Valeriy G. Narushin, Ülo Niinemets, Karl J. Nikas, Honghua Ruan, David A. Ratkowsky, Julian Schrader, Rolf Turner, Lin Wang, and Victoria Wimmer for their valuable help during creating this package. This work was supported by the National Key Research and Development Program of China (Grant No. 2021YFD02200403) and Simon Stevin Institute for Geometry (Antwerpen, Belguim).

Author(s)

Peijian Shi [aut, cre], Johan Gielis [aut], Brady K. Quinn [aut]

Maintainer: Peijian Shi <pjshi@njfu.edu.cn>

References

Gielis, J. (2003) A generic geometric transformation that unifies a wide range of natural and abstract shapes. *American Journal of Botany* 90, 333−338. doi:10.3732/ajb.90.3.333

Nelder, J.A., Mead, R. (1965). A simplex method for function minimization. *Computer Journal* 7, 308−313. doi:10.1093/comjnl/7.4.308

Shi, P., Fan, M., Ratkowsky, D.A., Huang, J., Wu, H., Chen, L., Fang, S., Zhang, C. (2017) Comparison of two ontogenetic growth equations for animals and plants. *Ecological Modelling* 349, 1−10. doi:10.1016/j.ecolmodel.2017.01.012

Shi, P., Gielis, J., Quinn, B.K., Niklas, K.J., Ratkowsky, D.A., Schrader, J., Ruan, H., Wang, L., Niinemets, Ü. (2022) 'biogeom': An R package for simulating and fitting natural shapes. *Annals of the New York Academy of Sciences* 1516, 123−134. doi:10.1111/nyas.14862

Shi, P., Ratkowsky, D.A., Gielis, J. (2020) The generalized Gielis geometric equation and its application. *Symmetry* 12, 645. doi:10.3390/sym12040645

Shi, P., Yu, K., Niklas, K.J., Schrader, J., Song, Y., Zhu, R., Li, Y., Wei, H., Ratkowsky, D.A. (2021) A general model for describing the ovate leaf shape. *Symmetry*, 13, 1524. doi:10.3390/sym13081524

`curveEPE` *Drawing the Preston Curve Produced by the the Explicit Preston Equation*

Description

`curveEPE` is used to draw the Preston curve that is produced by the explicit Preston equation.

Usage

```
curveEPE(P, np = 5000, simpver = NULL,
        fig.opt = FALSE, deform.fun = NULL, Par = NULL,
        xlim = NULL, ylim = NULL, unit = NULL, main="")
```

Arguments

`P`	the three location parameters and the parameters of the explicit Preston equation or one of its simplified versions.
`np`	the number of data points on the Preston curve.
`simpver`	an optional argument to use the simplfied version of the explicit Preston equation.
`fig.opt`	an optional argument to draw the Preston curve.
`deform.fun`	the deformation function used to describe the deviation from a theoretical Preston curve.
`Par`	the parameter(s) of the deformation function.
`xlim`	the range of the x-axis over which to plot the Preston curve.

`ylim`	the range of the y-axis over which to plot the Preston curve.
`unit`	the units of the x-axis and the y-axis when showing the Preston curve.
`main`	the main title of the figure.

Details

The first three elements of `P` are location parameters. The first two are the planar coordinates of the transferred origin, and the third is the angle between the major axis of the curve and the x-axis. Here, the major axis is a straight line through the two ends of an egg's profile (i.e., the mid-line of the egg's profile). The other arguments in `P` (except these first three location parameters), and `simpver` should correspond to those of `P` in `EPE`. `deform.fun` should take the form as: `deform.fun <- function(Par, z){...}`, where `z` is a two-dimensional matrix related to the x and y values. And the return value of `deform.fun` should be a `list` with two variables `x` and `y`.

Value

`x`	the x coordinates of the Preston curve.
`y`	the y coordinates of the Preston curve.

Note

When the rotation angle is zero (i.e., the third element in `P` is zero), `np` data points are distributed counterclockwise on the Preston curve from the rightmost end of the egg's profile to itself.

Author(s)

Peijian Shi <`pjshi@njfu.edu.cn`>, Johan Gielis <`johan.gielis@uantwerpen.be`>, Brady K. Quinn <`Brady.Quinn@dfo-mpo.gc.ca`>.

References

Preston, F.W. (1953) The shapes of birds' eggs. *The Auk* 70, 160−182.

Shi, P., Chen, L., Quinn, B.K., Yu, K., Miao, Q., Guo, X., Lian, M., Gielis, J., Niklas, K.J. (2022) Can we simply calculate the volume and surface area of an egg? Evidence from six species of poultry. *Annals of the New York Academy of Sciences* Under preparation.

Shi, P., Gielis, J., Quinn, B.K., Niklas, K.J., Ratkowsky, D.A., Schrader, J., Ruan, H., Wang, L., Niinemets, Ü. (2022) 'biogeom': An R package for simulating and fitting natural shapes. *Annals of the New York Academy of Sciences* 1516, 123−134. doi:10.1111/nyas.14862

Shi, P., Wang, L., Quinn, B.K., Gielis, J. (2022) A new program to estimate the parameters of Preston's equation, a general formula for describing the egg shape of birds. *Symmetry* Under review.

Todd, P.H., Smart, I.H.M. (1984) The shape of birds' eggs. *Journal of Theoretical Biology* 106, 239−243. doi:10.1016/00225193(84)900213

See Also

`EPE`, `fitEPE`, `lmPE`, `PE`, `TSE`

Examples

```
Para1 <- c(0, 0, 0, 10, 6, 0.325, -0.0415)
curveEPE(P=Para1, simpver=1, fig.opt=TRUE)
Para2 <- c(0, 0, pi, 10, 6, -0.325, -0.0415)
curveEPE(P=Para2, simpver=1, fig.opt=TRUE)

Para3 <- c(0, 0, 0, 10, 6, 0.325, -0.0415, 0.2)
curveEPE(P=Para3, simpver=NULL, fig.opt=TRUE)
Para4 <- c(0, 0, pi, 10, 6, -0.325, -0.0415, 0.2)
curveEPE(P=Para4, simpver=NULL, fig.opt=TRUE)

Para5 <- c(0, 0, pi/4, 10, 6, 0.325, -0.0415)
curveEPE(P=Para5, simpver=1,
        fig.opt=TRUE, main="A rotated egg shape")

# There is an example that introduces a deformation function in the egg-shape equation
myfun <- function(Par, z){
  x  <- z[,1]
  y  <- z[,2]
  k1 <- Par[1]
  k2 <- Par[2]
  y  <- y - k1*(y+k2)^2
  list(x=x, y=y)
}
deform.op <- curveEPE(P=Para1, np=5000, simpver=1,
                     fig.opt=TRUE, deform.fun=myfun, Par=c(0.05, 8))


graphics.off()
```

`curveETE` *Drawing the Troscianko Curve Produced by the Explicit Troscianko Equation*

Description

`curveETE` is used to draw the Troscianko curve that is produced by the explicit Troscianko equation.

Usage

```
curveETE(P, np = 5000, fig.opt = FALSE, deform.fun = NULL, Par = NULL,
        xlim = NULL, ylim = NULL, unit = NULL, main="")
```

Arguments

`P`	the three location parameters and the parameters of the explicit Troscianko equation.
`np`	the number of data points on the Troscianko curve.

`fig.opt`	an optional argument to draw the Troscianko curve.
`deform.fun`	the deformation function used to describe the deviation from a theoretical Troscianko curve.
`Par`	the parameter(s) of the deformation function.
`xlim`	the range of the x-axis over which to plot the Troscianko curve.
`ylim`	the range of the y-axis over which to plot the Troscianko curve.
`unit`	the units of the x-axis and the y-axis when showing the Troscianko curve.
`main`	the main title of the figure.

Details

The first three elements of `P` are location parameters. The first two are the planar coordinates of the transferred origin, and the third is the angle between the major axis of the curve and the x-axis. Here, the major axis is a straight line through the two ends of an egg's profile (i.e., the mid-line of the egg's profile). The other arguments in `P` (except these first three location parameters) should correspond to those of `P` in `ETE`. `deform.fun` should take the form as: `deform.fun <- function(Par, z){...}`, where `z` is a two-dimensional matrix related to the x and y values. And the return value of `deform.fun` should be a `list` with two variables `x` and `y`.

Value

`x`	the x coordinates of the Troscianko curve.
`y`	the y coordinates of the Troscianko curve.

Note

When the rotation angle is zero (i.e., the third element in `P` is zero), `np` data points are distributed counterclockwise on the Troscianko curve from the rightmost end of the egg's profile to itself.

Author(s)

Peijian Shi <`pjshi@njfu.edu.cn`>, Johan Gielis <`johan.gielis@uantwerpen.be`>, Brady K. Quinn <`Brady.Quinn@dfo-mpo.gc.ca`>.

References

Biggins, J.D., Montgomeries, R.M., Thompson, J.E., Birkhead, T.R. (2022) Preston's universal formula for avian egg shape. *Ornithology* In press. doi:10.1093/ornithology/ukac028

Biggins, J.D., Thompson, J.E., Birkhead, T.R. (2018) Accurately quantifying the shape of birds' eggs. *Ecology and Evolution* 8, 9728−9738. doi:10.1002/ece3.4412

Shi, P., Gielis, J., Quinn, B.K., Niklas, K.J., Ratkowsky, D.A., Schrader, J., Ruan, H., Wang, L., Niinemets, Ü. (2022) 'biogeom': An R package for simulating and fitting natural shapes. *Annals of the New York Academy of Sciences* 1516, 123−134. doi:10.1111/nyas.14862

Shi, P., Wang, L., Quinn, B.K., Gielis, J. (2022) A new program to estimate the parameters of Preston's equation, a general formula for describing the egg shape of birds. *Symmetry* Under review.

See Also

ETE, fitETE

Examples

```
Para1 <- c(0, 0, 0, 2.25, -0.377, -0.29, -0.16)
curveETE(P=Para1, fig.opt=TRUE)

# There is an example that introduces a deformation function in the egg-shape equation
myfun <- function(Par, z){
  x  <- z[,1]
  y  <- z[,2]
  k1 <- Par[1]
  k2 <- Par[2]
  y  <- y - k1*(y+k2)^2
  list(x=x, y=y)
}
deform.op <- curveETE(P=Para1, np=5000, fig.opt=TRUE, deform.fun=myfun, Par=c(0.05, 8))

graphics.off()
```

---

curveGE *Drawing the Gielis Curve*

Description

`curveGE` is used to draw the Gielis curve.

Usage

```
curveGE(expr, P, phi = seq(0, 2*pi, len = 2000),
        m = 1, simpver = NULL, nval = 1,
        fig.opt = FALSE, deform.fun = NULL, Par = NULL,
        xlim = NULL, ylim = NULL, unit = NULL, main="")
```

Arguments

`expr`	the original (or twin) Gielis equation or one of its simplified versions.
`P`	the three location parameters and the parameters of the original (or twin) Gielis equation or one of its simplified versions.
`phi`	the given polar angles at which we want to draw the Gielis curve.
`m`	the given m value that determines the number of angles of the Gielis curve within $[0, 2\pi)$.
`simpver`	an optional argument to use the simplfied version of the original (or twin) Gielis equation.
`nval`	the specified value for n_1 or n_2 or n_3 in the simplified versions.

`fig.opt`	an optional argument to draw the Gielis curve.
`deform.fun`	the deformation function used to describe the deviation from a theoretical Gielis curve.
`Par`	the parameter(s) of the deformation function.
`xlim`	the range of the x-axis over which to plot the Gielis curve.
`ylim`	the range of the y-axis over which to plot the Gielis curve.
`unit`	the units of the x-axis and the y-axis when showing the Gielis curve.
`main`	the main title of the figure.

Details

The first three elements of `P` are location parameters. The first two are the planar coordinates of the transferred polar point, and the third is the angle between the major axis of the curve and the x-axis. The other arguments in `P` (except these first three location parameters), `m`, `simpver`, and `nval` should correspond to `expr` (i.e., `GE` or `TGE`). Please note the differences in the simplified version number and the number of parameters between `GE` and `TGE`. `deform.fun` should take the form as: `deform.fun <- function(Par, z){...}`, where `z` is a two-dimensional matrix related to the x and y values. And the return value of `deform.fun` should be a `list` with two variables `x` and `y`.

Value

`x`	the x coordinates of the Gielis curve corresponding to the given polar angles `phi`.
`y`	the y coordinates of the Gielis curve corresponding to the given polar angles `phi`.
`r`	the polar radii of the Gielis curve corresponding to the given polar angles `phi`.

Note

`simpver` in `GE` is different from that in `TGE`.

Author(s)

Peijian Shi <`pjshi@njfu.edu.cn`>, Johan Gielis <`johan.gielis@uantwerpen.be`>, Brady K. Quinn <`Brady.Quinn@dfo-mpo.gc.ca`>.

References

Gielis, J. (2003) A generic geometric transformation that unifies a wide range of natural and abstract shapes. *American Journal of Botany* 90, 333-338. doi:10.3732/ajb.90.3.333

Li, Y., Quinn, B.K., Gielis, J., Li, Y., Shi, P. (2022) Evidence that supertriangles exist in nature from the vertical projections of *Koelreuteria paniculata* fruit. *Symmetry* 14, 23. doi:10.3390/sym14010023

Shi, P., Gielis, J., Niklas, K.J. (2022) Comparison of a universal (but complex) model for avian egg shape with a simpler model. *Annals of the New York Academy of Sciences* 1514, 34−42. doi:10.1111/nyas.14799

Shi, P., Gielis, J., Quinn, B.K., Niklas, K.J., Ratkowsky, D.A., Schrader, J., Ruan, H., Wang, L., Niinemets, Ü. (2022) 'biogeom': An R package for simulating and fitting natural shapes. *Annals of the New York Academy of Sciences* 1516, 123−134. doi:10.1111/nyas.14862

Shi, P., Ratkowsky, D.A., Gielis, J. (2020) The generalized Gielis geometric equation and its application. *Symmetry* 12, 645. doi:10.3390/sym12040645

Shi, P., Xu, Q., Sandhu, H.S., Gielis, J., Ding, Y., Li, H., Dong, X. (2015) Comparison of dwarf bamboos (*Indocalamus* sp.) leaf parameters to determine relationship between spatial density of plants and total leaf area per plant. *Ecology and Evolution* 5, 4578-4589. doi:10.1002/ece3.1728

See Also

`areaGE`, `fitGE`, `GE`, `TGE`

Examples

```
GE.par  <- c(2, 1, 4, 6, 3)
phi.vec <- seq(0, 2*pi, len=2000)
r.theor <- GE(P=GE.par, phi=phi.vec, m=5)

dev.new()
plot( phi.vec, r.theor, cex.lab=1.5, cex.axis=1.5,
      xlab=expression(italic(phi)), ylab=expression(italic("r")),
      type="l", col=4 )

curve.par <- c(1, 1, pi/4, GE.par)
GE.res    <- curveGE(GE, P=curve.par, fig.opt=TRUE, deform.fun=NULL, Par=NULL, m=5)
# GE.res$r

GE.res    <- curveGE( GE, P=c(0, 0, 0, 2, 4, 20), m=1, simpver=1, fig.opt=TRUE )
# GE.res$r

GE.res    <- curveGE( GE, P=c(1, 1, pi/4, 2, 1, 3), m=5, simpver=1, fig.opt=TRUE )
# GE.res$r

GE.res    <- curveGE( GE, P=c(1, 1, pi/4, 2, 1, 3), m=2, simpver=1, fig.opt=TRUE )
# GE.res$r

GE.res    <- curveGE( GE, P=c(1, 1, pi/4, 2, 0.05), m=1, simpver=2, fig.opt=TRUE )
# GE.res$r

GE.res    <- curveGE( GE, P=c(1, 1, pi/4, 2), m=4, simpver=3, nval=2, fig.opt=TRUE )
# GE.res$r

GE.res    <- curveGE( GE, P=c(1, 1, pi/4, 2, 0.6), m=4, simpver=8, nval=2, fig.opt=TRUE )
# GE.res$r

graphics.off()
```

`curveNRGE` *Drawing the Egg Shape Predicted by the Narushin-Romanov-Griffin Equation*

Description

`curveNRGE` is used to draw the egg shape predicted by the Narushin-Romanov-Griffin equation.

Usage

```
curveNRGE(P, np = 5000, fig.opt = FALSE, deform.fun = NULL, Par = NULL,
    xlim = NULL, ylim = NULL, unit = NULL, main = "")
```

Arguments

`P`	the three location parameters and the four parameters of the Narushin-Romanov-Griffin equation (Narushin et al., 2021).
`np`	the number of data points on the Narushin-Romanov-Griffin curve.
`fig.opt`	an optional argument to draw the Narushin-Romanov-Griffin curve.
`deform.fun`	the deformation function used to describe the deviation from a theoretical Narushin-Romanov-Griffin curve.
`Par`	the parameter(s) of the deformation function.
`xlim`	the range of the x-axis over which to plot the Narushin-Romanov-Griffin curve.
`ylim`	the range of the y-axis over which to plot the Narushin-Romanov-Griffin curve.
`unit`	the units of the x-axis and the y-axis when showing the Narushin-Romanov-Griffin curve.
`main`	the main title of the figure.

Details

The first three elements of `P` are location parameters. The first two are the planar coordinates of the transferred origin, and the third is the angle between the major axis of the curve and the x-axis. The other arguments in `P` should be the same as those in `NRGE`. `deform.fun` should take the form as: `deform.fun <- function(Par, z){...}`, where `z` is a two-dimensional matrix related to the x and y values. And the return value of `deform.fun` should be a `list` with two variables `x` and `y`.

Value

`x`	the x coordinates of the Narushin-Romanov-Griffin curve.
`y`	the y coordinates of the Narushin-Romanov-Griffin curve.

Note

When the rotation angle is zero (i.e., the third element in `P` is zero), `np` data points are distributed counterclockwise on the Narushin-Romanov-Griffin curve from the rightmost end of the egg's profile to itself.

Author(s)

Peijian Shi <`pjshi@njfu.edu.cn`>, Johan Gielis <`johan.gielis@uantwerpen.be`>, Brady K. Quinn <`Brady.Quinn@dfo-mpo.gc.ca`>.

References

Narushin, V.G., Romanov, M.N., Griffin, D.K. (2021) Egg and math: introducing a universal formula for egg shape. *Annals of the New York Academy of Sciences* 1505, 169−177. doi:10.1111/nyas.14680

Shi, P., Gielis, J., Niklas, K.J. (2022) Comparison of a universal (but complex) model for avian egg shape with a simpler model. *Annals of the New York Academy of Sciences* 1514, 34−42. doi:10.1111/nyas.14799

Shi, P., Gielis, J., Quinn, B.K., Niklas, K.J., Ratkowsky, D.A., Schrader, J., Ruan, H., Wang, L., Niinemets, Ü. (2022) 'biogeom': An R package for simulating and fitting natural shapes. *Annals of the New York Academy of Sciences* 1516, 123−134. doi:10.1111/nyas.14862

Tian, F., Wang, Y., Sandhu, H.S., Gielis, J., Shi, P. (2020) Comparison of seed morphology of two ginkgo cultivars. *Journal of Forestry Research* 31, 751−758. doi:10.1007/s116760180770y

See Also

`fitNRGE`, `NRGE`

Examples

```
PA   <- c(1, 1, pi/4, 11.5, 7.8, 1.1, 5.6)
resA <- curveNRGE(PA, np=5000, fig.opt=TRUE)
resB <- curveNRGE(PA, np=5000, fig.opt=TRUE, xlim=c(-6, 6),
                  ylim=c(-6, 6), main="A pear-shaped egg")
cbind(resB$x, resB$y)

graphics.off()
```

`curveovate` *Drawing the Ovate Leaf-Shape Curve*

Description

`curveovate` is used to draw the ovate leaf-shape curve.

Usage

```
curveovate(expr, P, x, fig.opt = FALSE,
           deform.fun = NULL, Par = NULL,
           xlim = NULL, ylim = NULL, unit = NULL, main = NULL)
```

Arguments

`expr`	the simplified version 1 of a performance equation.
`P`	the three location parameters and the parameters of the simplified version 1 of a performance equation.
`x`	the given x values to draw the ovate leaf-shape curve.
`fig.opt`	an optional argument to draw the ovate leaf-shape curve.
`deform.fun`	the deformation function used to describe the deviation from a theoretical ovate leaf-shape curve.
`Par`	the parameter(s) of the deformation function.
`xlim`	the range of the x-axis over which to plot the ovate leaf-shape curve.
`ylim`	the range of the y-axis over which to plot the ovate leaf-shape curve.
`unit`	the units of the x-axis and the y-axis when showing the ovate leaf-shape curve.
`main`	the main title of the figure.

Details

`P` has seven elements: three location parameters, and four model parameters, i.e., y_{opt}, x_{opt}, x_{max}, and δ. This means that `expr` is limited to be the simplified version 1 (where $x_{\mathrm{min}} = 0$) in `MbetaE`, `MBriereE`, and `MLRFE`. The first three elements of `P` are location parameters, among which the first two are the planar coordinates of the transferred origin, and the third is the angle between the major axis of the curve and the x-axis. `deform.fun` should take the form as: `deform.fun <- function(Par, z){...}`, where `z` is a two-dimensional matrix related to the x and y values. And the return value of `deform.fun` should be a `list` with two variables `x` and `y`.

Value

`x`	the x coordinates of the ovate leaf-shape curve.
`y`	the y coordinates of the ovate leaf-shape curve.

Note

The number of elements in `P` here has additional three location parameters than that in `MbetaE`, `MBriereE`, and `MLRFE`.

Author(s)

Peijian Shi <`pjshi@njfu.edu.cn`>, Johan Gielis <`johan.gielis@uantwerpen.be`>, Brady K. Quinn <`Brady.Quinn@dfo-mpo.gc.ca`>.

References

Jin, J., Quinn, B.K., Shi, P. (2022) The modified Brière equation and its applications. *Plants* 11, 1769. doi:10.3390/plants11131769

Li, Y., Zheng, Y., Ratkowsky, D.A., Wei, H., Shi, P. (2022) Application of an ovate leaf shape model to evaluate leaf bilateral asymmetry and calculate lamina centroid location. *Frontiers in Plant Science* 12, 822907. doi:10.3389/fpls.2021.822907

Shi, P., Fan, M., Ratkowsky, D.A., Huang, J., Wu, H., Chen, L., Fang, S., Zhang, C. (2017) Comparison of two ontogenetic growth equations for animals and plants. *Ecological Modelling* 349, 1−10. doi:10.1016/j.ecolmodel.2017.01.012

Shi, P., Gielis, J., Quinn, B.K., Niklas, K.J., Ratkowsky, D.A., Schrader, J., Ruan, H., Wang, L., Niinemets, Ü. (2022) 'biogeom': An R package for simulating and fitting natural shapes. *Annals of the New York Academy of Sciences* 1516, 123−134. doi:10.1111/nyas.14862

Shi, P., Yu, K., Niklas, K.J., Schrader, J., Song, Y., Zhu, R., Li, Y., Wei, H., Ratkowsky, D.A. (2021) A general model for describing the ovate leaf shape. *Symmetry* 13, 1524. doi:10.3390/sym13081524

See Also

`areaovate`, `fitovate`, `MbetaE`, `MBriereE`, `MLRFE`

Examples

```
P1  <- c(1, 1, pi/4, 2, 3, 10, 4)
RE1 <- curveovate(MLRFE, P=P1, x=seq(0, 10, by=0.1), fig.opt=TRUE)
RE2 <- curveovate(MbetaE, P=P1, x=seq(0, 10, by=0.1), fig.opt=TRUE)

dev.new()
plot(RE1$x, RE1$y, cex.lab=1.5, cex.axis=1.5, type="l",
  xlab=expression(italic("x")), ylab=expression(italic("y")))
lines(RE2$x, RE2$y, col=4)

graphics.off()
```

DEPE *Calculation of the First-Order Derivative of the Explicit Preston Equation*

Description

`DEPE` is used to calculate the first-order derivative of the explicit Preston equation at a given x-value.

Usage

```
DEPE(P, x, simpver = NULL)
```

Arguments

`P`	the parameters of the explicit Preston equation or one of its simplified versions.
`x`	the x-value used in the explicit Preston equation.
`simpver`	an optional argument to use the simplified version of the explicit Preston equation.

Details

When `simpver = NULL`, the first-order derivative of the explicit Preston equation at a given *x*-value is selected:

$$f(x) = \frac{b\left[a^4\,c_1 + a^3\,(2\,c_2 - 1)\,x + a^2\,(3\,c_3 - 2\,c_1)\,x^2 - 3\,a\,c_2 x^3 - 4\,c_3\,x^4\right]}{a^4\sqrt{a^2 - x^2}},$$

where `P` has five parameters: a, b, c_1, c_2, and c_3.

When `simpver = 1`, the first-order derivative of the simplified version 1 is selected:

$$f(x) = \frac{b\left[a^4\,c_1 + a^3\,(2\,c_2 - 1)\,x - 2\,a^2\,c_1\,x^2 - 3\,a\,c_2 x^3\right]}{a^4\sqrt{a^2 - x^2}},$$

where `P` has four parameters: a, b, c_1, and c_2.

When `simpver = 2`, the first-order derivative of the simplified version 2 is selected:

$$f(x) = \frac{b\left[a^4\,c_1 - a^3\,x - 2\,a^2\,c_1\,x^2\right]}{a^4\sqrt{a^2 - x^2}},$$

where `P` has three parameters: a, b, and c_1.

When `simpver = 3`, the first-order derivative of the simplified version 3 is selected:

$$f(x) = \frac{b\left[a^3\,(2\,c_2 - 1)\,x - 3\,a\,c_2 x^3\right]}{a^4\sqrt{a^2 - x^2}},$$

where `P` has three parameters: a, b, and c_2.

Note

The argument `P` in the `DEPE` function has the same parameters, as those in the `EPE` function.

Author(s)

Peijian Shi <`pjshi@njfu.edu.cn`>, Johan Gielis <`johan.gielis@uantwerpen.be`>, Brady K. Quinn <`Brady.Quinn@dfo-mpo.gc.ca`>.

References

Shi, P., Chen, L., Quinn, B.K., Yu, K., Miao, Q., Guo, X., Lian, M., Gielis, J., Niklas, K.J. (2022) Can we simply calculate the volume and surface area of an egg? Evidence from six species of poultry. *Annals of the New York Academy of Sciences* Under preparation.

See Also

`EPE`, `fitEPE`, `SurfaceAreaEPE`

Examples

```
Par3 <- c(4.27, 2.90, 0.0868, 0.0224, -0.0287)
xx1  <- seq(-4.27, 4.27, by=0.001)
f1   <- DEPE(P=Par3, x=xx1, simpver=NULL)
f2   <- -DEPE(P=Par3, x=xx1, simpver=NULL)

dev.new()
plot(xx1, f1, type="l", col=4, cex.lab=1.5, cex.axis=1.5,
     xlim=c(-5, 5), ylim=c(-35, 35), xlab=expression(italic(x)),
     ylab=expression(paste(italic(f), "(", italic(x), ")", sep="")))
lines(xx1, f2, col=2)

graphics.off()
```

`DETE` *Calculation of the First-Order Derivative of the Explicit Troscianko Equation*

Description

`DETE` is used to calculate the first-order derivative of the explicit Troscianko equation at a given *x*-value.

Usage

```
DETE(P, x)
```

Arguments

`P`	the parameters of the explicit Troscianko equation.
`x`	the *x*-value used in the explicit Troscianko equation.

Details

The first-order derivative of the explicit Troscianko equation at a given *x*-value is:

$$h(x) = \left\{\alpha_1 + \frac{2\,\alpha_2}{a}\,x - \frac{x}{a}\left[1-\left(\frac{x}{a}\right)^2\right]^{-1}\right\}\exp\left\{\alpha_0 + \alpha_1\left(\frac{x}{a}\right) + \alpha_2\left(\frac{x}{a}\right)^2\right\}\sqrt{1-\left(\frac{x}{a}\right)^2},$$

where `P` has five parameters: a, α_0, α_1, and α_2.

Note

The argument `P` in the `DETE` function has the same parameters, as those in the `ETE` function.

Author(s)

Peijian Shi <`pjshi@njfu.edu.cn`>, Johan Gielis <`johan.gielis@uantwerpen.be`>, Brady K. Quinn <`Brady.Quinn@dfo-mpo.gc.ca`>.

References

Shi, P., Chen, L., Quinn, B.K., Yu, K., Miao, Q., Guo, X., Lian, M., Gielis, J., Niklas, K.J. (2022) Can we simply calculate the volume and surface area of an egg? Evidence from six species of poultry. *Annals of the New York Academy of Sciences* Under preparation.

See Also

`ETE`, `fitETE`, `SurfaceAreaETE`

Examples

```
Par5 <- c(2.25, -0.38, -0.29, -0.16)
xx2  <- seq(-2.25, 2.25, by=0.001)
h1   <- DETE(P=Par5, x=xx2)
h2   <- -DETE(P=Par5, x=xx2)
ind  <- which(is.na(h1) | is.na(h2))
xx2  <- xx2[-ind]
h1   <- h1[-ind]
h2  <- h2[-ind]

dev.new()
plot(xx2, h1, type="l", col=4, cex.lab=1.5, cex.axis=1.5,
     xlim=c(-2.25, 2.25), ylim=c(-30, 30), xlab=expression(italic(x)),
     ylab=expression(paste(italic(h), "(", italic(x), ")", sep="")))
lines(xx2, h2, col=2)

graphics.off()
```

`DNRGE` *Calculation of the First-Order Derivative of the Narushin-Romanov-Griffin Equation*

Description

`DNRGE` is used to calculate the first-order derivative of the Narushin-Romanov-Griffin equation at a given x-value.

Usage

```
DNRGE(P, x)
```

Arguments

`P`	the parameters of the Narushin-Romanov-Griffin equation.
`x`	the x-value used in the Narushin-Romanov-Griffin equation.

Details

Let us define:

$$
\begin{gathered}
f_1(x) = \frac{B}{2}\sqrt{\frac{A^2-4x^2}{A^2+8Cx+4C^2}},\\
f_2(x) = \sqrt{\frac{A\left(A^2+8Cx+4C^2\right)}{2(A-2C)x^2+\left(A^2+8AC-4C^2\right)x+2AC^2+A^2C+A^3}},\\
f_3(x) = A^2-4x,\\
f_4(x) = A^2+8Cx+4C^2,\\
E = \frac{\sqrt{5.5A^2+11AC+4C^2}\cdot\left(\sqrt{3}AB-2D\sqrt{A^2+2AC+4C^2}\right)}{\sqrt{3}AB\left(\sqrt{5.5A^2+11AC+4C^2}-2\sqrt{A^2+2AC+4C^2}\right)},\\
F = 2\left(A-2C\right),\\
G = A^2+8AC-4C^2,\\
H = 2AC^2+A^2C+A^3,
\end{gathered}
$$

and then the first-order derivative of the Narushin-Romanov-Griffin equation at a given x-value is:

$$
J(x) = -\frac{4\,f_1(x)\left[C\,f_3(x)+x\,f_4(x)\right]}{f_3(x)\cdot f_4(x)}\left\{1-E\cdot\left[1-f_2(x)\right]\right\}-\frac{AE}{2}\,\frac{f_1(x)}{f_2(x)}\,\frac{f_4(x)\cdot(2Fx+G)}{\left(Fx^2+Gx+H\right)^2},
$$

where `P` has four parameters: A, B, C, and D.

Note

The argument `P` in the `DNRGE` function has the same parameters, as those in the `NRGE` function.

Author(s)

Peijian Shi <`pjshi@njfu.edu.cn`>, Johan Gielis <`johan.gielis@uantwerpen.be`>, Brady K. Quinn <`Brady.Quinn@dfo-mpo.gc.ca`>.

References

Narushin, V.G., Romanov, M.N., Griffin, D.K. (2021) Egg and math: introducing a universal formula for egg shape. *Annals of the New York Academy of Sciences* 1505, 169−177. doi:10.1111/nyas.14680

Narushin, V.G., Romanov, M.N., Mishra, B., Griffin, D.K. (2022) Mathematical progression of avian egg shape with associated area and volume determinations. *Annals of the New York Academy of Sciences* 1513, 65−78. doi:10.1111/nyas.14771

Shi, P., Chen, L., Quinn, B.K., Yu, K., Miao, Q., Guo, X., Lian, M., Gielis, J., Niklas, K.J. (2022) Can we simply calculate the volume and surface area of an egg? Evidence from six species of poultry. *Annals of the New York Academy of Sciences* Under preparation.

See Also

`fitNRGE`, `NRGE`, `SurfaceAreaNRGE`

Examples

```
Par6 <- c(4.51, 3.18, 0.1227, 2.2284)
xx3  <- seq(-4.51/2, 4.51/2, len=2000)
J1   <- DNRGE(P=Par6, x=xx3)
J2   <- -DNRGE(P=Par6, x=xx3)
ind  <- which(is.na(J1) | is.na(J2))
xx3  <- xx3[-ind]
J1   <- J1[-ind]
J2   <- J2[-ind]

dev.new()
plot(xx3, J1, type=”l”, col=4, cex.lab=1.5, cex.axis=1.5,
     xlim=c(-4.51/2, 4.51/2), ylim=c(-20, 20), xlab=expression(italic(x)),
     ylab=expression(paste(italic(J), ”(”, italic(x), ”)”, sep=””)))
lines(xx3, J2, col=2)

graphics.off()
```

DSGE *Calculation of the First-Order Derivative of the Simplified Gielis Equation*

Description

`DSGE` is used to calculate the first-order derivative of the simplified Gielis equation at a given φ-value.

Usage

```
DSGE(P, phi)
```

Arguments

`P`	the parameters of the simplified Gielis equation, including a, n_1, and n_2.
`phi`	the φ-value used in the simplified Gielis equation.

Details

The first-order derivative of the simplified Gielis equation with arguments `simpver` = 1 and `m` = 1 at a given φ-value is:

$$g(x) = \frac{a}{4}\,\frac{n_2}{n_1}\left[\left(\cos\frac{\varphi}{4}\right)^{n_2-1}\left(\sin\frac{\varphi}{4}\right)-\left(\sin\frac{\varphi}{4}\right)^{n_2-1}\left(\cos\frac{\varphi}{4}\right)\right]\left[\left(\cos\frac{\varphi}{4}\right)^{n_2}+\left(\sin\frac{\varphi}{4}\right)^{n_2}\right]^{-\frac{1}{n_1}-1},$$

where `P` has three parameters: a, n_1, and n_2.

Note

The argument `P` in the `DSGE` function only has the three parameters: a, n_1, and n_2.

Author(s)

Peijian Shi <`pjshi@njfu.edu.cn`>, Johan Gielis <`johan.gielis@uantwerpen.be`>, Brady K. Quinn <`Brady.Quinn@dfo-mpo.gc.ca`>.

References

Chen, Z. (2012) Volume and area of revolution under polar coordinate system. *Studies in College Mathematics* 15(6), 9−11.

Shi, P., Chen, L., Quinn, B.K., Yu, K., Miao, Q., Guo, X., Lian, M., Gielis, J., Niklas, K.J. (2022) Can we simply calculate the volume and surface area of an egg? Evidence from six species of poultry. *Annals of the New York Academy of Sciences* Under preparation.

See Also

`GE`, `fitGE`, `SurfaceAreaSGE`

Examples

```
Par7 <- c(1.124, 14.86, 49.43)
phi1 <- seq(0, pi, len=2000)
g1   <- DSGE(P=Par7, phi=phi1)

dev.new()
plot(phi1, g1, type="l", col=4, cex.lab=1.5, cex.axis=1.5,
     xlab=expression(italic(varphi)),
     ylab=expression(paste(italic(g), "(", italic(varphi), ")", sep="")))

graphics.off()
```

`eggs` *Egg Boundary Data of Nine Species of Birds*

Description

The data consist of the egg boundary data of nine species of birds.

Usage

```
data(eggs)
```

Details

In the data set, there are four columns of variables: `Code`, `LatinName`, `x`, and `y`. `Code` saves the codes of individual eggs; `LatinName` saves the Latin names of the nine species of birds; `x` saves the x coordinates of the egg boundary in the Cartesian coordinate system (cm); and `y` saves the y coordinates of the egg boundary in the Cartesian coordinate system (cm). In `Code`, codes 1-9 represent *Strix uralensis*, *Dromaius novaehollandiae*, *Turdus philomelos*, *Gallus gallus*, *Pandion haliaetus*, *Uria aalge*, *Uria lomvia*, *Gallinago media*, and *Aptenodytes patagonicus*, respectively.

References

Narushin, V.G., Romanov, M.N., Griffin, D.K. (2021) Egg and math: introducing a universal formula for egg shape. *Annals of the New York Academy of Sciences* 1505, 169−177. doi:10.1111/nyas.14680

Shi, P., Gielis, J., Niklas, K.J. (2022) Comparison of a universal (but complex) model for avian egg shape with a simpler model. *Annals of the New York Academy of Sciences* 1514, 34−42. doi:10.1111/nyas.14799

Tian, F., Wang, Y., Sandhu, H.S., Gielis, J., Shi, P. (2020) Comparison of seed morphology of two ginkgo cultivars. *Journal of Forestry Research* 31, 751−758. doi:10.1007/s116760180770y

Examples

```
data(eggs)

uni.C <- sort( unique(eggs$Code) )
ind   <- 8
Data  <- eggs[eggs$Code==uni.C[ind], ]
x0    <- Data$x
y0    <- Data$y

Res1 <- adjdata(x0, y0, ub.np=1000, times=1.2, len.pro=1/20)
x1   <- Res1$x
y1   <- Res1$y

dev.new()
plot( x1, y1, asp=1, cex.lab=1.5, cex.axis=1.5, type="l",
      xlab=expression(italic("x")), ylab=expression(italic("y")),
      pch=1, col=1 )

Res2 <- adjdata(x0, y0, ub.np=60, times=1, len.pro=1/2, index.sp=20)
x2   <- Res2$x
y2   <- Res2$y

Res3 <- adjdata(x0, y0, ub.np=60, times=1, len.pro=1/2, index.sp=100)
x3   <- Res3$x
y3   <- Res3$y

dev.new()
plot( x2, y2, asp=1, cex.lab=1.5, cex.axis=1.5,
      xlab=expression(italic("x")), ylab=expression(italic("y")),
      pch=1, col=4 )
```

```
points( x3, y3, col=2)

graphics.off()
```

`EPE` *Calculation of the Ordinate For an Arbitrary Point on the Preston Curve in the Plane*

Description

`EPE` is used to calculate the *y*-value for an arbitrary point on the Preston curve that was generated by the explicit Preston equation or one of its simplified versions for a given *x*-value.

Usage

```
EPE(P, x, simpver = NULL)
```

Arguments

`P`	the parameters of the explicit Preston equation or one of its simplified versions.
`x`	the *x*-value used in the explicit Preston equation.
`simpver`	an optional argument to use the simplified version of the explicit Preston equation.

Details

When `simpver = NULL`, the explicit Preston equation is selected:

$$y = b\,\sqrt{1-\left(\frac{x}{a}\right)^2}\,\left(1+c_1\,\frac{x}{a}+c_2\left(\frac{x}{a}\right)^2+c_3\left(\frac{x}{a}\right)^3\right),$$

where `P` has five parameters: a, b, c_1, c_2, and c_3.

When `simpver = 1`, the simplified version 1 is selected:

$$y = b\,\sqrt{1-\left(\frac{x}{a}\right)^2}\,\left(1+c_1\,\frac{x}{a}+c_2\left(\frac{x}{a}\right)^2\right),$$

where `P` has four parameters: a, b, c_1, and c_2.

When `simpver = 2`, the simplified version 2 is selected:

$$y = b\,\sqrt{1-\left(\frac{x}{a}\right)^2}\,\left(1+c_1\,\frac{x}{a}\right),$$

where `P` has three parameters: a, b, and c_1.

When `simpver = 3`, the simplified version 3 is selected:

$$y = b\,\sqrt{1-\left(\frac{x}{a}\right)^2}\,\left(1+c_2\left(\frac{x}{a}\right)^2\right),$$

where `P` has three parameters: a, b, and c_2.

Value

The y values predicted by the explicit Preston equation.

Note

We only considered the upper part of the egg-shape curve in the above expressions because the lower part is symmetrical to the upper part around the *x*-axis. The mid-line of an egg's profile in `EPE` is aligned to the *x*-axis, while the mid-line of an egg's profile in `PE` is aligned to the *y*-axis. The `EPE` function has the same parameters, `P`, as those in the `PE` function. The explicit Preston equation is used for calculating an egg's volume and surface area, when the parameters, which are saved in the `P` vector, are obtained using the `fitEPE` function or the `lmPE` function based on the `TSE` function. In addition, the values in `x` > a (i.e., the first element in `P`) are forced to be a, and the values in `x` < $-a$ will be forced to be $-a$.

Author(s)

Peijian Shi <`pjshi@njfu.edu.cn`>, Johan Gielis <`johan.gielis@uantwerpen.be`>, Brady K. Quinn <`Brady.Quinn@dfo-mpo.gc.ca`>.

References

Shi, P., Chen, L., Quinn, B.K., Yu, K., Miao, Q., Guo, X., Lian, M., Gielis, J., Niklas, K.J. (2022) Can we simply calculate the volume and surface area of an egg? Evidence from six species of poultry. *Annals of the New York Academy of Sciences* Under preparation.

See Also

`curveEPE`, `fitEPE`, `PE`, `SurfaceAreaEPE`, `VolumeEPE`

Examples

```
Par3 <- c(4.27, 2.90, 0.0868, 0.0224, -0.0287)
xx1  <- seq(-4.27, 4.27, by=0.001)
yy1  <- EPE(P=Par3, x=xx1, simpver=NULL)
yy2  <- -EPE(P=Par3, x=xx1, simpver=NULL)

dev.new()
plot(xx1, yy1, asp=1, type="l", col=4, cex.lab=1.5, cex.axis=1.5,
     xlim=c(-5, 5), ylim=c(-5, 5),
     xlab=expression(italic(x)), ylab=expression(italic(y)))
lines(xx1, yy2, col=2)

graphics.off()
```

`ETE` *Calculation of the Ordinate For an Arbitrary Point on the Troscianko Curve in the Plane*

Description

`ETE` is used to calculate the *y*-value for an arbitrary point on the Troscianko curve that was generated by the explicit Troscianko equation.

Usage

```
ETE(P, x)
```

Arguments

`P`	the parameters of the explicit Troscianko equation, including a, α_0, α_1, and α_2.
`x`	the *x*-value used in the explicit Troscianko equation.

Details

The explicit Troscianko equation is recommended as:

$$y = a\,\exp\left\{\alpha_0 + \alpha_1\left(\frac{x}{a}\right) + \alpha_2\left(\frac{x}{a}\right)^2\right\}\sqrt{1-\left(\frac{x}{a}\right)^2},$$

where x and y represent the abscissa and ordinate of an arbitrary point on the explicit Troscianko curve; a, α_0, α_1, and α_2 are parameters to be estimated.

Value

The y values predicted by the explicit Troscianko equation.

Note

We only considered the upper part of the egg-shape curve in the above expressions because the lower part is symmetrical to the upper part around the *x*-axis. The mid-line of an egg's profile in `ETE` is aligned to the *x*-axis. The argument, `P`, in the `ETE` function has the same parameters, α_0, α_1, and α_2, as those in the `TE` function. However, the former has an additional parameter a than the latter, which represents half the egg's length. The `lmTE` function is based on the `TE` function, while the `fitETE` function is based on the `ETE` function here. In addition, the values in `x` > a (i.e., the first element in `P`) are forced to be a, and the values in `x` < $-a$ will be forced to be $-a$.

Author(s)

Peijian Shi <`pjshi@njfu.edu.cn`>, Johan Gielis <`johan.gielis@uantwerpen.be`>, Brady K. Quinn <`Brady.Quinn@dfo-mpo.gc.ca`>.

References

Biggins, J.D., Montgomeries, R.M., Thompson, J.E., Birkhead, T.R. (2022) Preston's universal formula for avian egg shape. *Ornithology* In press. doi:10.1093/ornithology/ukac028

Biggins, J.D., Thompson, J.E., Birkhead, T.R. (2018) Accurately quantifying the shape of birds' eggs. *Ecology and Evolution* 8, 9728–9738. doi:10.1002/ece3.4412

Shi, P., Gielis, J., Quinn, B.K., Niklas, K.J., Ratkowsky, D.A., Schrader, J., Ruan, H., Wang, L., Niinemets, Ü. (2022) 'biogeom': An R package for simulating and fitting natural shapes. *Annals of the New York Academy of Sciences* 1516, 123–134. doi:10.1111/nyas.14862

Shi, P., Wang, L., Quinn, B.K., Gielis, J. (2022) A new program to estimate the parameters of Preston's equation, a general formula for describing the egg shape of birds. *Symmetry* Under review.

Troscianko, J. (2014). A simple tool for calculating egg shape, volume and surface area from digital images. *Ibis*, 156, 874–878. doi:10.1111/ibi.12177

See Also

`curveETE`, `fitETE`, `SurfaceAreaETE`, `VolumeETE`

Examples

```
Par5 <- c(2.25, -0.38, -0.29, -0.16)
xx2  <- seq(-2.25, 2.25, len=2000)
yy3  <- ETE(P=Par5, x=xx2)
yy4  <- -ETE(P=Par5, x=xx2)

dev.new()
plot(xx2, yy3, asp=1, type="l", col=4, cex.lab=1.5, cex.axis=1.5,
     xlim=c(-3, 3), ylim=c(-3, 3),
     xlab=expression(italic(x)), ylab=expression(italic(y)))
lines(xx2, yy4, col=2)

graphics.off()
```

`fitEPE` *Data-Fitting Function for the Explicit Preston Equation*

Description

`fitEPE` is used to estimate the parameters of the explicit Preston equation or one of its simplified versions.

Usage

```
fitEPE(x, y, ini.val, simpver = NULL,
      control = list(), par.list = FALSE,
      stand.fig = TRUE, angle = NULL, fig.opt = FALSE, np = 2000,
      xlim = NULL, ylim = NULL, unit = NULL, main = NULL)
```

Arguments

`x`	the x coordinates of an egg's profile.
`y`	the y coordinates of an egg's profile.
`ini.val`	the list of initial values for the model parameters.
`simpver`	an optional argument to use the simplified version of the explicit Preston equation.
`control`	the list of control parameters for using the `optim` function in package **stats**.
`par.list`	the option of showing the list of parameters on the screen.
`stand.fig`	the option of drawing the observed and predicted profiles of an egg at the standard state (i.e., the egg's centre is located at (0, 0), and the mid-line is aligned to the x-axis).
`angle`	the angle between the mid-line and the x-axis, which can be defined by the user.
`fig.opt`	an optional argument of drawing the observed and predicted profiles of an egg at arbitrary angle between the major axis and the x-axis.
`np`	the number of data points on the predicted explicit Preston curve.
`xlim`	the range of the x-axis over which to plot the Preston curve.
`ylim`	the range of the y-axis over which to plot the Preston curve.
`unit`	the unit of the x-axis and the y-axis when showing the Preston curve.
`main`	the main title of the figure.

Details

The `simpver` argument should correspond to `EPE`. Here, the major axis (i.e., the mid-line of an egg's profile) is the straight line trhough the two ends of the egg's length. The Nelder-Mead algorithm (Nelder and Mead, 1965) is used to carry out the optimization of minimizing the residual sum of squares (RSS) between the observed and predicted y values. The `optim` function in package **stats** was used to carry out the Nelder-Mead algorithm. When `angle = NULL`, the observed egg's profile will be shown at its initial angle in the scanned image; when `angle` is a numerical value (e.g., $\pi/4$) defined by the user, it indicates that the major axis is rotated by the amount ($\pi/4$) counterclockwise from the x-axis.

Value

`par`	the estimates of the model parameters.
`scan.length`	the observed length of the egg's profile.
`scan.width`	the observed width of the egg's profile.
`scan.area`	the observed area of the egg's profile.
`scan.perimeter`	the observed perimeter of the egg's profile.
`r.sq`	the coefficient of determination between the observed and predicted y values on the Preston curve.
`RSS`	the residual sum of squares between the observed and predicted y values on the Preston curve.

`sample.size`	the number of data points used in the data fitting.
`x.stand.obs`	the observed x coordinates of the points on the Preston curve at the standard state.
`y.stand.obs`	the observed y coordinates of the points on the Preston curve at the standard state.
`y.stand.pred`	the predicted y coordinates of the points on the Preston curve at the standard state.
`x.obs`	the observed x coordinates of the points on the Preston curve at the transferred polar angles as defined by the user.
`y.obs`	the observed y coordinates of the points on the Preston curve at the transferred polar angles as defined by the user.
`y.pred`	the predicted y coordinates of the points on the Preston curve at the transferred polar angles as defined by the user.

Note

In the outputs, there are no `x.stand.pred` and `x.pred`, because `y.stand.obs` and `y.stand.pred` share the same x values (i.e., `x.stand.obs`), and `y.obs` and `y.pred` share the same x values (i.e., `x.obs`).

Author(s)

Peijian Shi <`pjshi@njfu.edu.cn`>, Johan Gielis <`johan.gielis@uantwerpen.be`>, Brady K. Quinn <`Brady.Quinn@dfo-mpo.gc.ca`>.

References

Nelder, J.A., Mead, R. (1965) A simplex method for function minimization. *Computer Journal* 7, 308−313. doi:10.1093/comjnl/7.4.308

Preston, F.W. (1953) The shapes of birds' eggs. *The Auk* 70, 160−182.

Shi, P., Chen, L., Quinn, B.K., Yu, K., Miao, Q., Guo, X., Lian, M., Gielis, J., Niklas, K.J. (2022) Can we simply calculate the volume and surface area of an egg? Evidence from six species of poultry. *Annals of the New York Academy of Sciences* Under preparation.

Shi, P., Gielis, J., Quinn, B.K., Niklas, K.J., Ratkowsky, D.A., Schrader, J., Ruan, H., Wang, L., Niinemets, Ü. (2022) 'biogeom': An R package for simulating and fitting natural shapes. *Annals of the New York Academy of Sciences* 1516, 123−134. doi:10.1111/nyas.14862

Shi, P., Wang, L., Quinn, B.K., Gielis, J. (2022) A new program to estimate the parameters of Preston's equation, a general formula for describing the egg shape of birds. *Symmetry* Under review.

Todd, P.H., Smart, I.H.M. (1984) The shape of birds' eggs. *Journal of Theoretical Biology* 106, 239−243. doi:10.1016/00225193(84)900213

See Also

`curveEPE`, `PE`, `lmPE`, `TSE`

Examples

```
data(eggs)

uni.C <- sort( unique(eggs$Code) )
ind   <- 8
Data  <- eggs[eggs$Code==uni.C[ind], ]
x0    <- Data$x
y0    <- Data$y

Res1 <- adjdata(x0, y0, ub.np=2000, times=1.2, len.pro=1/20)
x1   <- Res1$x
y1   <- Res1$y

dev.new()
plot( x1, y1, asp=1, cex.lab=1.5, cex.axis=1.5,  type="l", col=4,
      xlab=expression(italic("x")), ylab=expression(italic("y")) )


  simpver   <- NULL
  res1      <- lmPE( x1, y1, simpver=simpver, dev.angle=seq(-0.05, 0.05, by=0.0001),
                     unit="cm", fig.opt=FALSE )
  x0.ini    <- mean( x1 )
  y0.ini    <- mean( y1 )
  theta.ini <- res1$theta
  a.ini     <- res1$scan.length / 2
  b.ini     <- res1$scan.width / 2
  c1.ini    <- res1$par[2] / res1$par[1]
  c2.ini    <- res1$par[3] / res1$par[1]
  c3.ini    <- res1$par[4] / res1$par[1]

  ini.val <- list(x0.ini, y0.ini, theta.ini, a.ini, b.ini, c1.ini, c2.ini, c3.ini)

  res0 <- fitEPE( x=x1, y=y1, ini.val=ini.val,
                 simpver=simpver, unit="cm", par.list=FALSE,
                 stand.fig=FALSE, angle=NULL, fig.opt=FALSE,
                 control=list(reltol=1e-30, maxit=50000),
                 np=2000 )

  n.loop <- 12
  Show   <- FALSE
  for(i in 1:n.loop){
    ini.val <- res0$par
    if(i==n.loop) Show <- TRUE
    print(paste(i, "/", n.loop, sep=""))
    res0 <- fitEPE( x=x1, y=y1, ini.val=ini.val,
                   simpver=simpver, unit="cm", par.list=FALSE,
                   stand.fig=Show, angle=pi/4, fig.opt=Show,
                   control=list(reltol=1e-30, maxit=50000),
                   np=2000 )
  }

  # The numerical values of the location and model parameters
```

```
  res0$par

  # The root-mean-square error (RMSE) between
  #   the observed and predicted y values
  sqrt(res0$RSS/res0$sample.size)

  sqrt(sum((res0$y.stand.obs-res0$y.stand.pred)^2)/length(res0$y.stand.obs))

  # To calculate the volume of the egg
  VolumeEPE(P=res0$par[4:8])

  # To calculate the surface area of the egg
  SurfaceAreaEPE(P=res0$par[4:8])


graphics.off()
```

`fitETE` *Data-Fitting Function for the Explicit Troscianko Equation*

Description

`fitETE` is used to estimate the parameters of the explicit Troscianko equation.

Usage

```
fitETE(x, y, ini.val, control = list(), par.list = FALSE,
      stand.fig = TRUE, angle = NULL, fig.opt = FALSE, np = 2000,
      xlim = NULL, ylim = NULL, unit = NULL, main = NULL)
```

Arguments

`x`	the x coordinates of an egg's profile.
`y`	the y coordinates of an egg's profile.
`ini.val`	the list of initial values for the model parameters.
`control`	the list of control parameters for using the `optim` function in package **stats**.
`par.list`	the option of showing the list of parameters on the screen.
`stand.fig`	the option of drawing the observed and predicted profiles of an egg at the standard state (i.e., the egg's centre is located at (0, 0), and the mid-line is aligned to the x-axis).
`angle`	the angle between the mid-line and the x-axis, which can be defined by the user.
`fig.opt`	an optional argument of drawing the observed and predicted profiles of an egg at arbitrary angle between the major axis and the x-axis.
`np`	the number of data points on the predicted explicit Troscianko curve.
`xlim`	the range of the x-axis over which to plot the Troscianko curve.
`ylim`	the range of the y-axis over which to plot the Troscianko curve.
`unit`	the unit of the x-axis and the y-axis when showing the Troscianko curve.
`main`	the main title of the figure.

Details

Here, the major axis (i.e., the mid-line of an egg's profile) is the straight line trhough the two ends of the egg's length. The Nelder-Mead algorithm (Nelder and Mead, 1965) is used to carry out the optimization of minimizing the residual sum of squares (RSS) between the observed and predicted y values. The `optim` function in package **stats** was used to carry out the Nelder-Mead algorithm. When `angle = NULL`, the observed egg's profile will be shown at its initial angle in the scanned image; when `angle` is a numerical value (e.g., $\pi/4$) defined by the user, it indicates that the major axis is rotated by the amount ($\pi/4$) counterclockwise from the x-axis.

Value

`par`	the estimates of the model parameters.
`scan.length`	the observed length of the egg's profile.
`scan.width`	the observed width of the egg's profile.
`scan.area`	the observed area of the egg's profile.
`scan.perimeter`	the observed perimeter of the egg's profile.
`r.sq`	the coefficient of determination between the observed and predicted y values on the Troscianko curve.
`RSS`	the residual sum of squares between the observed and predicted y values on the Troscianko curve.
`sample.size`	the number of data points used in the data fitting.
`x.stand.obs`	the observed x coordinates of the points on the Troscianko curve at the standard state.
`y.stand.obs`	the observed y coordinates of the points on the Troscianko curve at the standard state.
`y.stand.pred`	the predicted y coordinates of the points on the Troscianko curve at the standard state.
`x.obs`	the observed x coordinates of the points on the Troscianko curve at the transferred polar angles as defined by the user.
`y.obs`	the observed y coordinates of the points on the Troscianko curve at the transferred polar angles as defined by the user.
`y.pred`	the predicted y coordinates of the points on the Troscianko curve at the transferred polar angles as defined by the user.

Note

In the outputs, there are no `x.stand.pred` and `x.pred`, because `y.stand.obs` and `y.stand.pred` share the same x values (i.e., `x.stand.obs`), and `y.obs` and `y.pred` share the same x values (i.e., `x.obs`).

Author(s)

Peijian Shi `<pjshi@njfu.edu.cn>`, Johan Gielis `<johan.gielis@uantwerpen.be>`, Brady K. Quinn `<Brady.Quinn@dfo-mpo.gc.ca>`.

References

Biggins, J.D., Montgomeries, R.M., Thompson, J.E., Birkhead, T.R. (2022) Preston's universal formula for avian egg shape. *Ornithology* In press. doi:10.1093/ornithology/ukac028

Biggins, J.D., Thompson, J.E., Birkhead, T.R. (2018) Accurately quantifying the shape of birds' eggs. *Ecology and Evolution* 8, 9728–9738. doi:10.1002/ece3.4412

Nelder, J.A., Mead, R. (1965) A simplex method for function minimization. *Computer Journal* 7, 308–313. doi:10.1093/comjnl/7.4.308

Shi, P., Gielis, J., Quinn, B.K., Niklas, K.J., Ratkowsky, D.A., Schrader, J., Ruan, H., Wang, L., Niinemets, Ü. (2022) 'biogeom': An R package for simulating and fitting natural shapes. *Annals of the New York Academy of Sciences* 1516, 123–134. doi:10.1111/nyas.14862

Shi, P., Wang, L., Quinn, B.K., Gielis, J. (2022) A new program to estimate the parameters of Preston's equation, a general formula for describing the egg shape of birds. *Symmetry* Under review.

Troscianko, J. (2014). A simple tool for calculating egg shape, volume and surface area from digital images. *Ibis*, 156, 874–878. doi:10.1111/ibi.12177

See Also

`curveETE`, `TE`, `lmTE`

Examples

```
data(eggs)

uni.C <- sort( unique(eggs$Code) )
ind   <- 8
Data  <- eggs[eggs$Code==uni.C[ind], ]
x0    <- Data$x
y0    <- Data$y

Res1 <- adjdata(x0, y0, ub.np=2000, times=1.2, len.pro=1/20)
x1   <- Res1$x
y1   <- Res1$y

dev.new()
plot( x1, y1, asp=1, cex.lab=1.5, cex.axis=1.5,  type="l", col=4,
      xlab=expression(italic("x")), ylab=expression(italic("y")) )


  res1      <- lmTE( x1, y1, unit="cm", fig.opt=FALSE )

  if(FALSE){
    P0  <- c(res1$scan.length/2, res1$par)
    xx  <- seq(-res1$scan.length/2, res1$scan.length/2, len=2000)
    yy1 <- ETE(P0, xx)
    yy2 <- -ETE(P0, xx)
    dev.new()
    plot( xx, yy1, cex.lab=1.5, cex.axis=1.5, asp=1, col=2,
          ylim=c(-res1$scan.length/2, res1$scan.length/2),
          type="l", xlab=expression(x), ylab=expression(y) )
```

```
    lines( xx, yy2, col=4 )
  }

  x0.ini    <- mean( x1 )
  y0.ini    <- mean( y1 )
  theta.ini <- res1$theta
  a.ini     <- res1$scan.length / 2
  alpha0.ini <- res1$par[1]
  alpha1.ini <- res1$par[2]
  alpha2.ini <- res1$par[3]

  ini.val <- list(x0.ini, y0.ini, theta.ini, a.ini, alpha0.ini, alpha1.ini, alpha2.ini)

  res0 <- fitETE( x=x1, y=y1, ini.val=ini.val,
                 unit="cm", par.list=FALSE,
                 stand.fig=FALSE, angle=NULL, fig.opt=FALSE,
                 control=list(reltol=1e-30, maxit=50000),
                 np=2000 )

  n.loop <- 12
  Show   <- FALSE
  for(i in 1:n.loop){
    ini.val <- res0$par
    if(i==n.loop) Show <- TRUE
    print(paste(i, "/", n.loop, sep=""))
    res0 <- fitETE( x=x1, y=y1, ini.val=ini.val,
                   unit="cm", par.list=FALSE,
                   stand.fig=Show, angle=pi/4, fig.opt=Show,
                   control=list(reltol=1e-30, maxit=50000),
                   np=2000 )
  }

  # The numerical values of the location and model parameters
  res0$par

  # The root-mean-square error (RMSE) between
  #   the observed and predicted y values
  sqrt(res0$RSS/res0$sample.size)

  sqrt(sum((res0$y.stand.obs-res0$y.stand.pred)^2)/length(res0$y.stand.obs))

  # To calculate the volume of the egg
  VolumeETE(P=res0$par[4:7])

  # To calculate the surface area of the egg
  SurfaceAreaETE(P=res0$par[4:7])


graphics.off()
```

`fitGE` *Data-Fitting Function for the Gielis Equation*

Description

`fitGE` is used to estimate the parameters of the original (or twin) Gielis equation or one of its simplified versions.

Usage

```
fitGE(expr, x, y, ini.val, m = 1, simpver = NULL,
      nval = nval, control = list(), par.list = FALSE,
      stand.fig = TRUE, angle = NULL, fig.opt = FALSE, np = 2000,
      xlim = NULL, ylim = NULL, unit = NULL, main = NULL)
```

Arguments

`expr`	the original (or twin) Gielis equation or one of its simplified versions.
`x`	the x coordinates of a polygon's boundary.
`y`	the y coordinates of a polygon's boundary.
`ini.val`	the list of initial values for the model parameters.
`m`	the given m value that determines the number of angles of the Gielis curve within $[0, 2\pi)$.
`simpver`	an optional argument to use the simplified version of the original (or twin) Gielis equation.
`nval`	the specified value for n_1 or n_2 or n_3 in the simplified versions.
`control`	the list of control parameters for using the `optim` function in package **stats**.
`par.list`	the option of showing the list of parameters on the screen.
`stand.fig`	the option of drawing the observed and predicted polygons at the standard state (i.e., the polar point is located at (0, 0), and the major axis overlaps with the x-axis).
`angle`	the angle between the major axis and the x-axis, which can be defined by the user.
`fig.opt`	an optional argument of drawing the observed and predicted polygons at arbitrary angle between the major axis and the x-axis.
`np`	the number of data points on the predicted Gielis curve.
`xlim`	the range of the x-axis over which to plot the Gielis curve.
`ylim`	the range of the y-axis over which to plot the Gielis curve.
`unit`	the unit of the x-axis and the y-axis when showing the Gielis curve.
`main`	the main title of the figure.

Details

The arguments of `m`, `simpver`, and `nval` should correspond to `expr` (i.e., `GE` or `TGE`). Please note the differences in the simplified version number and the number of parameters between `GE` and `TGE`. The Nelder-Mead algorithm (Nelder and Mead, 1965) is used to carry out the optimization of minimizing the residual sum of squares (RSS) between the observed and predicted radii. The `optim` function in package **stats** was used to carry out the Nelder-Mead algorithm. When `angle = NULL`, the observed polygon will be shown at its initial angle in the scanned image; when `angle` is a numerical value (e.g., $\pi/4$) defined by the user, it indicates that the major axis is rotated by the amount ($\pi/4$) counterclockwise from the x-axis.

Value

`par`	the estimates of the model parameters.
`scan.length`	the observed length of the polygon.
`scan.width`	the observed width of the polygon.
`scan.area`	the observed area of the polygon.
`r.sq`	the coefficient of determination between the observed and predicted polar radii.
`RSS`	the residual sum of squares between the observed and predicted polar radii.
`sample.size`	the number of data points used in the data fitting.
`phi.stand.obs`	the polar angles at the standard state.
`phi.trans`	the transferred polar angles rotated as defined by the user.
`r.stand.obs`	the observed polar radii at the standard state.
`r.stand.pred`	the predicted polar radii at the standard state.
`x.stand.obs`	the observed x coordinates at the standard state.
`x.stand.pred`	the predicted x coordinates at the standard state.
`y.stand.obs`	the observed y coordinates at the standard state.
`y.stand.pred`	the predicted y coordinates at the standard state.
`r.obs`	the observed polar radii at the transferred polar angles as defined by the user.
`r.pred`	the predicted polar radii at the transferred polar angles as defined by the user.
`x.obs`	the observed x coordinates at the transferred polar angles as defined by the user.
`x.pred`	the predicted x coordinates at the transferred polar angles as defined by the user.
`y.obs`	the observed y coordinates at the transferred polar angles as defined by the user.
`y.pred`	the predicted y coordinates at the transferred polar angles as defined by the user.

Note

`simpver` in `GE` is different from that in `TGE`.

Author(s)

Peijian Shi <`pjshi@njfu.edu.cn`>, Johan Gielis <`johan.gielis@uantwerpen.be`>, Brady K. Quinn <`Brady.Quinn@dfo-mpo.gc.ca`>.

References

Gielis, J. (2003) A generic geometric transformation that unifies a wide range of natural and abstract shapes. *American Journal of Botany* 90, 333−338. doi:10.3732/ajb.90.3.333

Li, Y., Quinn, B.K., Gielis, J., Li, Y., Shi, P. (2022) Evidence that supertriangles exist in nature from the vertical projections of *Koelreuteria paniculata* fruit. *Symmetry* 14, 23. doi:10.3390/sym14010023

Nelder, J.A., Mead, R. (1965) A simplex method for function minimization. *Computer Journal* 7, 308−313. doi:10.1093/comjnl/7.4.308

Shi, P., Gielis, J., Quinn, B.K., Niklas, K.J., Ratkowsky, D.A., Schrader, J., Ruan, H., Wang, L., Niinemets, Ü. (2022) 'biogeom': An R package for simulating and fitting natural shapes. *Annals of the New York Academy of Sciences* 1516, 123−134. doi:10.1111/nyas.14862

Shi, P., Ratkowsky, D.A., Gielis, J. (2020) The generalized Gielis geometric equation and its application. *Symmetry* 12, 645. doi:10.3390/sym12040645

Shi, P., Xu, Q., Sandhu, H.S., Gielis, J., Ding, Y., Li, H., Dong, X. (2015) Comparison of dwarf bamboos (*Indocalamus* sp.) leaf parameters to determine relationship between spatial density of plants and total leaf area per plant. *Ecology and Evolution* 5, 4578-4589. doi:10.1002/ece3.1728

See Also

`areaGE`, `curveGE`, `DSGE`, `GE`, `SurfaceAreaSGE`, `TGE`, `VolumeSGE`

Examples

```
data(eggs)

uni.C <- sort( unique(eggs$Code) )
ind   <- 1
Data  <- eggs[eggs$Code==uni.C[ind], ]
x0    <- Data$x
y0    <- Data$y

Res1 <- adjdata(x0, y0, ub.np=200, times=1.2, len.pro=1/20)
x1   <- Res1$x
y1   <- Res1$y
Res2 <- adjdata(x0, y0, ub.np=40, times=1, len.pro=1/2, index.sp=20)
x2   <- Res2$x
y2   <- Res2$y
Res3 <- adjdata(x0, y0, ub.np=100, times=1, len.pro=1/2, index.sp=100)
x3   <- Res3$x
y3   <- Res3$y

dev.new()
plot( x2, y2, asp=1, cex.lab=1.5, cex.axis=1.5,
      xlab=expression(italic("x")), ylab=expression(italic("y")),
      pch=1, col=4 )
points( x3, y3, col=2)


  x0.ini    <- mean( x1 )
```

```
y0.ini    <- mean( y1 )
theta.ini <- pi
a.ini     <- sqrt(2) * max( y0.ini-min(y1), x0.ini-min(x1) )
n1.ini <- c(5, 25)
n2.ini <- c(15, 25)
if(ind == 2){
  n1.ini <- c(0.5, 1)
  n2.ini <- c(6, 12)
}
ini.val <- list(x0.ini, y0.ini, theta.ini, a.ini, n1.ini, n2.ini)

Res4 <- fitGE( GE, x=x1, y=y1, ini.val=ini.val,
               m=1, simpver=1, nval=1, unit="cm",
               par.list=FALSE, fig.opt=TRUE, angle=NULL,
               control=list(reltol=1e-20, maxit=20000),
               np=2000 )
Res4$par
sqrt(sum((Res4$y.stand.obs-Res4$y.stand.pred)^2)/Res4$sample.size)

xx    <- Res4$x.stand.obs
yy    <- Res4$y.stand.obs

library(spatstat.geom)
poly0 <- as.polygonal(owin(poly=list(x=xx, y=yy)))
area(poly0)

areaGE(GE, P = Res4$par[4:6],
       m=1, simpver=1)


# The following code is used to
#   calculate the root-mean-square error (RMSE) in the y-coordinates
ind1  <- which(yy >= 0)
ind2  <- which(yy < 0)
xx1   <- xx[ind1] # The upper part of the egg
yy1   <- yy[ind1]
xx2   <- xx[ind2] # The lower part of the egg
yy2   <- yy[ind2]
Para  <- c(0, 0, 0, Res4$par[4:length(Res4$par)])
PartU <- curveGE(GE, P=Para, phi=seq(0, pi, len=100000), m=1, simpver=1, fig.opt=FALSE)
xv1   <- PartU$x
yv1   <- PartU$y
PartL <- curveGE(GE, P=Para, phi=seq(pi, 2*pi, len=100000), m=1, simpver=1, fig.opt=FALSE)
xv2   <- PartL$x
yv2   <- PartL$y
ind3  <- c()
for(q in 1:length(xx1)){
  ind.temp <- which.min(abs(xx1[q]-xv1))
  ind3     <- c(ind3, ind.temp)
}
ind4  <- c()
for(q in 1:length(xx2)){
  ind.temp <- which.min(abs(xx2[q]-xv2))
```

```
    ind4     <- c(ind4, ind.temp)
  }
  RSS   <- sum((yy1-yv1[ind3])^2) + sum((yy2-yv2[ind4])^2)
  RMSE  <- sqrt( RSS/length(yy) )


  # To calculate the volume of the Gielis egg when simpver=1 & m=1
  VolumeSGE(P=Res4$par[4:6])

  # To calculate the surface area of the Gielis egg when simpver=1 & m=1
  SurfaceAreaSGE(P=Res4$par[4:6])


graphics.off()
```

`fitNRGE` *Parameter Estimation for the Narushin-Romanov-Griffin Equation*

Description

`fitNRGE` is used to estimate the parameters of the Narushin-Romanov-Griffin equation.

Usage

```
fitNRGE(x, y, dev.angle = NULL, ini.C = c(-1, 0.1, 0.5, 1),
        strip.num = 2000, control = list(), fig.opt = TRUE, np = 2000,
        xlim = NULL, ylim = NULL, unit = NULL, main = NULL)
```

Arguments

`x`	the x coordinates of the edge of an egg's boundary.
`y`	the y coordinates of the edge of an egg's boundary.
`dev.angle`	the angle of deviation for the axis associated with the maximum distance between two points on an egg's profile from the mid-line of the egg's profile.
`ini.C`	the initial value(s) of parameter C in the Narushin-Romanov-Griffin equation.
`strip.num`	the number of equidistant strips intersecting with the egg's boundary that are horizontally placed. See Shi et al. (2018, 2020) for details.
`control`	the list of control parameters for using the `optim` function in package **stats**.
`fig.opt`	an optional argument to draw the observed and predicted egg's boundaries.
`np`	the number of data points on the predicted Narushin-Romanov-Griffin curve.
`xlim`	the range of the x-axis over which to plot the Narushin-Romanov-Griffin curve.
`ylim`	the range of the y-axis over which to plot the Narushin-Romanov-Griffin curve.
`unit`	the units of the x-axis and the y-axis when showing the Narushin-Romanov-Griffin curve.
`main`	the main title of the figure.

Details

The NRGE (see `NRGE`) has a complex model structure with four parameters (i.e., A, B, C, and D). Because three out of four parameters of NRGE have clear biological and geometric meanings (i.e., A, B, and D), their values could be estimated by means of numerical calculation. After obtaining the numerical values of the three parameters, the Nelder-Mead algorithm (Nelder and Mead, 1965) was used to estimate C. Because of the failure of the optimization method to estimate the major axis (i.e., the mid-line) and model parameters of NRGE, it was difficult to define the egg length axis, although it is essential for calculating A, B, and D. For this reason, two methods were used to obtain the major axis: the maximum distance method, and the longest axis adjustment method. In the first method, the straight line through two points forming the maximum distance on the egg's profile is defined as the major axis (i.e., the mid-line). In the second method, we assume that there is an angle of deviation for the longest axis (i.e., the axis associated with the maximum distance between two points on an egg's profile) from the mid-line of the egg's profile. That is to say, the mid-line of an egg's profile is not the axis associated with the maximum distance between two points on the egg's profile. When `angle = NULL`, the maximum distance method is used; when `angle` is a numerical value or a numerical vector, the longest axis adjustment method is used. Here, the numerical value of `dev.angle` is not the angle of deviation for the major axis of an egg's profile from the x-axis, and instead it is the angle of deviation for the longest axis (associated with the maximum distance between two points on the egg's profile) from the mid-line of the egg's profile. Once the major axis is established, the distance of the major axis can be calculated as the estimate of A. Using the maximum distance method, A equals the maximum distance. Using the longest axis adjustment method, A may be slightly smaller than the maximum distance. After rotating the major axis to make it overlap with the x-axis, a large number of equidistant strips can be used (e.g., 2000) from the egg base to egg tip to intersect the egg's boundary. This methodology makes it easy to obtain the maximum egg's breadth (i.e., B) and D. The residual sum of squares (RSS) between the observed and predicted y values can be minimized using an optimization method (Nelder and Mead, 1965) to estimate C. Despite the complex structure of NRGE (see `NRGE`), the optimization method for estimating the remaining parameter C becomes feasible after the other three parameters have been numerically estimated. Please see Shi et al. (2022) for details.

Value

`theta`	the angle between the longest axis of an egg's profile (i.e., the axis associated with the maximum distance between two points on the egg's profile) and the x-axis.
`epsilon`	the optimal angle of deviation for the longest axis (associated with the maximum distance between two points on an egg's profile) from the mid-line of the egg's profile, when `dev.angle` is not `NULL`.
`RSS.vector`	the vector of residual sum of squares corresponding to `dev.angle`, when `dev.angle` is not `NULL`.
`x.obs`	the observed x coordinates.
`y.obs`	the observed y coordinates.
`y.pred`	the predicted y coordinates corresponding to the the observed x coordinates.
`par`	the estimates of the four model parameters in the Narushin-Romanov-Griffin equation.
`scan.length`	the length of the egg's boundary. The default is the maximum distance between two points on the egg's boundary.

`scan.width`	the maximum width of the egg's boundary.
`scan.area`	the area of the egg's boundary.
`scan.perimeter`	the perimeter of the egg's boundary based on all data points on the egg's boundary.
`RSS`	the residual sum of squares between the observed and predicted y values.
`sample.size`	the number of data points used in the numerical calculation.
`RMSE`	the root-mean-square errors between the observed and predicted y values.

Note

`theta` is the calculated angle between the longest axis (i.e., the axis associated with the maximum distance between two points on an egg's profile) and the x-axis, and `epsilon` is the calculated angle of deviation for the longest axis from the mid-line of the egg's profile. This means that the angle between the mid-line and the x-axis is equal to `theta` + `epsilon`. In the outputs, there is no `x.pred`, because `y.obs` and `y.pred` share the same x values (i.e., `x.obs`).

Author(s)

Peijian Shi <`pjshi@njfu.edu.cn`>, Johan Gielis <`johan.gielis@uantwerpen.be`>, Brady K. Quinn <`Brady.Quinn@dfo-mpo.gc.ca`>.

References

Narushin, V.G., Romanov, M.N., Griffin, D.K. (2021) Egg and math: introducing a universal formula for egg shape. *Annals of the New York Academy of Sciences* 1505, 169−177. doi:10.1111/nyas.14680

Nelder, J.A., Mead, R. (1965). A simplex method for function minimization. *Computer Journal* 7, 308−313. doi:10.1093/comjnl/7.4.308

Shi, P., Gielis, J., Niklas, K.J. (2022) Comparison of a universal (but complex) model for avian egg shape with a simpler model. *Annals of the New York Academy of Sciences* 1514, 34−42. doi:10.1111/nyas.14799

Shi, P., Gielis, J., Quinn, B.K., Niklas, K.J., Ratkowsky, D.A., Schrader, J., Ruan, H., Wang, L., Niinemets, Ü. (2022) 'biogeom': An R package for simulating and fitting natural shapes. *Annals of the New York Academy of Sciences* 1516, 123−134. doi:10.1111/nyas.14862

Shi, P., Niinemets, Ü., Hui, C., Niklas, K.J., Yu, X., Hölscher, D. (2020) Leaf bilateral symmetry and the scaling of the perimeter vs. the surface area in 15 vine species. *Forests* 11, 246. doi:10.3390/f11020246

Shi, P., Zheng, X., Ratkowsky, D.A., Li, Y., Wang, P., Cheng, L. (2018) A simple method for measuring the bilateral symmetry of leaves. *Symmetry* 10, 118. doi:10.3390/sym10040118

Tian, F., Wang, Y., Sandhu, H.S., Gielis, J., Shi, P. (2020) Comparison of seed morphology of two ginkgo cultivars. *Journal of Forestry Research* 31, 751−758. doi:10.1007/s116760180770y

See Also

`curveNRGE`, `NRGE`

Examples

```
data(eggs)

uni.C <- sort( unique(eggs$Code) )
ind   <- 8
Data  <- eggs[eggs$Code==uni.C[ind], ]
x0    <- Data$x
y0    <- Data$y

Res1  <- adjdata(x0, y0, ub.np=3000, len.pro=1/20)
x1    <- Res1$x
y1    <- Res1$y

dev.new()
plot( Res1$x, Res1$y, asp=1, cex.lab=1.5, cex.axis=1.5,
      xlab=expression(italic("x")), ylab=expression(italic("y")) )


  Res2 <- fitNRGE(x1, y1, dev.angle=NULL, ini.C=c(-1, -0.1, seq(0.1, 1, by=0.05)),
                  strip.num=2000, fig.opt=TRUE)

  dev.new()
  plot(Res2$x.obs, Res2$y.obs, asp=1, cex.lab=1.5, cex.axis=1.5,
       xlab=expression(italic("x")), ylab=expression(italic("y")),
       type="l", col=4)
  lines( Res2$x.obs, Res2$y.pred, col=2)

  Res3 <- fitNRGE(x1, y1, dev.angle=seq(-0.05, 0.05, by=0.01),
                  ini.C=c(-1, -0.1, seq(0.1, 1, by=0.05)),
                  strip.num=2000, fig.opt=TRUE)

  zeta <- Res3$theta + Res3$epsilon
  x2   <- x1*cos(zeta) + y1*sin(zeta)
  y2   <- y1*cos(zeta) - x1*sin(zeta)
  plot( x2-min(x2), y2-min(y2), asp=1, col="grey70", cex=1,
        xlab=expression(italic("x")), ylab=expression(italic("y")) )
  lines(Res3$x.obs-min(Res3$x.obs), Res3$y.obs-min(Res3$y.obs), col=4)
  lines(Res3$x.obs-min(Res3$x.obs), Res3$y.pred-min(Res3$y.obs), col=2)

  RMSE <- sqrt( Res3$RSS / Res3$sample.size )
  RMSE

  # To calculate the volume of the egg
  VolumeNRGE(P=Res3$par)

  # To calculate the surface area of the egg
  SurfaceAreaNRGE(P=Res3$par)


graphics.off()
```

`fitovate` *Data-Fitting Function for the Ovate Leaf-Shape Equation*

Description

`fitovate` is used to estimate the parameters of a simplified performance equation.

Usage

```
fitovate(expr, x, y, ini.val,
         par.list = FALSE, stand.fig = TRUE, control = list(),
         angle = NULL, fig.opt = FALSE, index.xmax = 3, np = 2000,
         xlim = NULL, ylim = NULL, unit = NULL, main = NULL)
```

Arguments

`expr`	the simplified version 1 of the performance equations.
`x`	the x coordinates of a polygon's boundary.
`y`	the y coordinates of a polygon's boundary.
`ini.val`	the initial values of the simplified version 1 of a performance equation.
`par.list`	an optional argument to show the list of parameters on the screen.
`stand.fig`	an optional argument to draw the observed and predicted polygons' boundaries at the standard state (i.e., the origin is located at (0, 0), and the major axis overlaps with the x-axis).
`control`	the list of control parameters for using the `optim` function in package **stats**.
`angle`	the angle between the major axis of the polygon and the x-axis, which can be defined by the user.
`fig.opt`	an optional argument to draw the observed and predicted polygons at an arbitrary angle between the major axis and the x-axis.
`index.xmax`	the specified index in parameters representing $x_{\max}$.
`np`	the number of data points on the predicted ovate leaf-shape curve.
`xlim`	the range of the x-axis over which to plot the ovate leaf-shape curve.
`ylim`	the range of the y-axis over which to plot the ovate leaf-shape curve.
`unit`	the units of the x-axis and the y-axis when showing the ovate leaf-shape curve.
`main`	the main title of the figure.

Details

`ini.val` is a list for seven parameters: three location parameters, and four model parameters, i.e., y_{opt}, x_{opt}, $x_{\max}$, and δ. This means that `expr` is limited to being the simplified version 1 (where $x_{\min} = 0$) in `MbetaE`, `MBriereE`, and `MLRFE`. The initial values for the first three parameters in `ini.val` are location parameters, among which the first two are the planar coordinates of the transferred origin, and the third is the angle between the major axis of the polygon and the x-axis. The

Nelder-Mead algorithm (Nelder and Mead, 1965) is used to carry out the optimization of minimizing the residual sum of squares (RSS) between the observed and predicted radii. The `optim` function in package **stats** was used to carry out the Nelder-Mead algorithm. When `angle = NULL`, the observed polygon will be shown at its initial angle in the scanned image; when `angle` is a numerical value (e.g., $\pi/4$) defined by the user, it indicates that the major axis is rotated by the amount $(\pi/4)$ counterclockwise from the x-axis.

Value

`par`	the estimates of the model parameters.
`r.sq`	the coefficient of determination between the observed and predicted y values.
`RSS`	the residual sum of squares between the observed and predicted y values.
`sample.size`	the number of data points on the polygon's boundary in the data fitting.
`scan.length`	the observed length of the polygon's boundary.
`scan.width`	the observed width of the polygon's boundary.
`scan.perimeter`	the observed perimeter of the polygon's boundary.
`scan.area`	the observed area of the polygon's boundary.
`pred.length`	the predicted length of the polygon's boundary.
`pred.width`	the predicted width of the polygon's boundary.
`pred.perimeter`	the predicted perimeter of the polygon's boundary.
`pred.area`	the predicted area of the polygon's boundary.
`x.stand.obs`	the observed x coordinates at the standard state.
`x.stand.pred`	the predicted x coordinates at the standard state.
`y.stand.obs`	the observed y coordinates at the standard state.
`y.stand.pred`	the predicted y coordinates at the standard state.
`x.obs`	the observed x coordinates at the transferred angles defined by the user.
`x.pred`	the predicted x coordinates at the transferred angles defined by the user.
`y.obs`	the observed y coordinates at the transferred angles defined by the user.
`y.pred`	the predicted y coordinates at the transferred angles defined by the user.

Note

There are seven parameters in total for the value of `par`. The transferred angle denotes the angle between the major axis and the x-axis.

Author(s)

Peijian Shi <`pjshi@njfu.edu.cn`>, Johan Gielis <`johan.gielis@uantwerpen.be`>, Brady K. Quinn <`Brady.Quinn@dfo-mpo.gc.ca`>.

References

Jin, J., Quinn, B.K., Shi, P. (2022) The modified Brière equation and its applications. *Plants* 11, 1769. doi:10.3390/plants11131769

Li, Y., Zheng, Y., Ratkowsky, D.A., Wei, H., Shi, P. (2022) Application of an ovate leaf shape model to evaluate leaf bilateral asymmetry and calculate lamina centroid location. *Frontiers in Plant Science* 12, 822907. doi:10.3389/fpls.2021.822907

Nelder, J.A., Mead, R. (1965) A simplex method for function minimization. *Computer Journal* 7, 308−313. doi:10.1093/comjnl/7.4.308

Shi, P., Fan, M., Ratkowsky, D.A., Huang, J., Wu, H., Chen, L., Fang, S., Zhang, C. (2017) Comparison of two ontogenetic growth equations for animals and plants. *Ecological Modelling* 349, 1−10. doi:10.1016/j.ecolmodel.2017.01.012

Shi, P., Gielis, J., Quinn, B.K., Niklas, K.J., Ratkowsky, D.A., Schrader, J., Ruan, H., Wang, L., Niinemets, Ü. (2022) 'biogeom': An R package for simulating and fitting natural shapes. *Annals of the New York Academy of Sciences* 1516, 123−134. doi:10.1111/nyas.14862

Shi, P., Yu, K., Niklas, K.J., Schrader, J., Song, Y., Zhu, R., Li, Y., Wei, H., Ratkowsky, D.A. (2021) A general model for describing the ovate leaf shape. *Symmetry* 13, 1524. doi:10.3390/sym13081524

See Also

`areaovate`, `curveovate`, `MbetaE`, `MBriereE`, `MLRFE`

Examples

```
data(Neocinnamomum)

uni.C <- sort( unique(Neocinnamomum$Code) )
ind   <- 2
Data  <- Neocinnamomum[Neocinnamomum$Code==uni.C[ind], ]
x0    <- Data$x
y0    <- Data$y

Res1 <- adjdata(x0, y0, ub.np=200, len.pro=1/20)
x1   <- Res1$x
y1   <- Res1$y

dev.new()
plot( Res1$x, Res1$y, asp=1, cex.lab=1.5, cex.axis=1.5,
      xlab=expression(italic("x")), ylab=expression(italic("y")) )


  x0.ini    <- min( x1 )
  y0.ini    <- min( y1 )
  theta.ini <- pi/4
  len.max   <- max( max(y1)-min(y1), max(x1)-min(x1) ) *2/sqrt(2)
  a.ini     <- c(0.1, 0.01, 0.001, 0.0001)
  m.ini     <- c(0.1, 0.5, 1, 2)
  x2.ini    <- len.max
  delta.ini <- c(0.5, 1)
```

```
  ini.val   <- list(x0.ini, y0.ini, theta.ini, a.ini, m.ini, x2.ini, delta.ini)

  Res2 <- fitovate(MBriereE, x=x1, y=y1, ini.val=ini.val,
            par.list=FALSE, fig.opt=TRUE, angle=pi/6,
            control=list(reltol=1e-20, maxit=20000),
            np=2000, unit=NULL)
  Res2$RSS

  x0.ini    <- min( x1 )
  y0.ini    <- min( y1 )
  theta.ini <- pi/4
  len.max   <- max( max(y1)-min(y1), max(x1)-min(x1) ) *2/sqrt(2)
  yc.ini    <- len.max/3
  xc.ini    <- 1/4*len.max
  x2.ini    <- len.max
  delta.ini <- c(0.5, seq(1, 5, by=5))
  ini.val   <- list(x0.ini, y0.ini, theta.ini, yc.ini, xc.ini, x2.ini, delta.ini)

  Res3 <- fitovate( MbetaE, x=x1, y=y1, ini.val=ini.val,
                    par.list=TRUE, fig.opt=TRUE, angle=pi/3,
                    control=list(reltol=1e-20, maxit=20000),
                    np=2000, unit=NULL )
  Res3$RSS

  Res4 <- fitovate( MLRFE, x=x1, y=y1, ini.val=ini.val,
                    unit=NULL, par.list=FALSE, fig.opt=TRUE,
                    angle=NULL, control=list(reltol=1e-20,
                    maxit=20000), np=2000)
  Res4$RSS


graphics.off()
```

`fitsigmoid` *Data-Fitting Function for the Sigmoid Growth Equation*

Description

`fitsigmoid` is used to estimate the parameters of a sigmoid growth equation based on the integral of a performance equation or one of its simplified versions.

Usage

```
fitsigmoid(expr, x, y, ini.val, simpver = 1,
           control = list(), par.list = FALSE, fig.opt = FALSE,
           xlim = NULL, ylim = NULL, xlab = NULL, ylab = NULL,
           main = NULL, subdivisions = 100L,
           rel.tol = .Machine$double.eps^0.25,
           abs.tol = rel.tol, stop.on.error = TRUE,
           keep.xy = FALSE, aux = NULL)
```

Arguments

`expr`	a performance equation or one of its simplified versions that is used to build a sigmoid growth equation.
`x`	the observed investigation times.
`y`	the observed y values (i.e., biomass, height, body length, etc.).
`ini.val`	the initial values of the model parameters.
`simpver`	an optional argument to use the simplified version of the performance equation.
`control`	the list of control parameters for using the `optim` function in package **stats**.
`par.list`	the option of showing the list of parameters on the screen.
`fig.opt`	an optional argument to draw the observations and the predicted sigmoid curve.
`xlim`	the range of the x-axis over which to plot a sigmoid growth curve.
`ylim`	the range of the y-axis over which to plot a sigmoid growth curve.
`xlab`	the label of the x-axis when showing a sigmoid growth curve.
`ylab`	the label of the y-axis when showing a sigmoid growth curve.
`main`	the main title of the figure.
`subdivisions`	please see the arguments for the `integrate` function in package **stats**.
`rel.tol`	please see the arguments for the `integrate` function in package **stats**.
`abs.tol`	please see the arguments for the `integrate` function in package **stats**.
`stop.on.error`	please see the arguments for the `integrate` function in package **stats**.
`keep.xy`	please see the arguments for the `integrate` function in package **stats**.
`aux`	please see the arguments for the `integrate` function in package **stats**.

Details

Here, `ini.val` only includes the initial values of the model parameters as a list. The Nelder-Mead algorithm (Nelder and Mead, 1965) is used to carry out the optimization of minimizing the residual sum of squares (RSS) between the observed and predicted y values. The `optim` function in package **stats** was used to carry out the Nelder-Mead algorithm. The performance equations denote `MbetaE`, `MBriereE`, `MLRFE` and their simplified versions. The arguments of `P` and `simpver` should correspond to `expr` (i.e., `MbetaE` or `MBriereE` or `MLRFE`). The sigmoid equation is the integral of a performance equation or one of its simplified versions.

Value

`par`	the estimates of the model parameters.
`r.sq`	the coefficient of determination between the observed and predicted y values.
`RSS`	the residual sum of squares between the observed and predicted y values.
`sample.size`	the number of data points used in the data fitting.
`x`	the observed x values.
`y`	the observed y values.
`y.pred`	the predicted y values.

Note

Here, the user can define other performance equations, but new equations or their simplified versions should include the lower and upper thresholds on the x-axis corresponding to $y = 0$, whose indices should be the same as those in `MbetaE` or `MBriereE` or `MLRFE`.

Author(s)

Peijian Shi <`pjshi@njfu.edu.cn`>, Johan Gielis <`johan.gielis@uantwerpen.be`>, Brady K. Quinn <`Brady.Quinn@dfo-mpo.gc.ca`>.

References

Jin, J., Quinn, B.K., Shi, P. (2022) The modified Brière equation and its applications. *Plants* 11, 1769. doi:10.3390/plants11131769

Nelder, J.A., Mead, R. (1965) A simplex method for function minimization. *Computer Journal* 7, 308−313. doi:10.1093/comjnl/7.4.308

Shi, P., Fan, M., Ratkowsky, D.A., Huang, J., Wu, H., Chen, L., Fang, S., Zhang, C. (2017) Comparison of two ontogenetic growth equations for animals and plants. *Ecological Modelling* 349, 1−10. doi:10.1016/j.ecolmodel.2017.01.012

Shi, P., Gielis, J., Quinn, B.K., Niklas, K.J., Ratkowsky, D.A., Schrader, J., Ruan, H., Wang, L., Niinemets, Ü. (2022) 'biogeom': An R package for simulating and fitting natural shapes. *Annals of the New York Academy of Sciences* 1516, 123−134. doi:10.1111/nyas.14862

See Also

`areaovate`, `MbetaE`, `MBriereE`, `MLRFE`, `sigmoid`

Examples

```
# The shrimp growth data(See the supplementary table in West et al., 2001)
# West, G.B., Brown, J.H., Enquist, B.J. (2001) A general model for ontogenetic growth.
#     Nature 413, 628-631.
t0 <- c(3, 60, 90, 120, 150, 180, 384)
m0 <- c(0.001, 0.005, 0.018, 0.037, 0.06, 0.067, 0.07)

dev.new()
plot( t0, m0, cex.lab=1.5, cex.axis=1.5, col=4,
      xlab=expression(italic(x)), ylab=expression(italic(y)) )

xopt0   <- seq(100, 150, by=5)
ini.val <- list(0.035, xopt0, 200, 1)
resu1  <- fitsigmoid(MLRFE, x=t0, y=m0, ini.val=ini.val, simpver=1, fig.opt=TRUE, par.list=TRUE)


  delta0  <- c(0.5, 1, 2, 5, 10, 20)
  ini.val <- list(0.035, 150, -100, 200, delta0)
  resu2   <- fitsigmoid(MLRFE, x=t0, y=m0, ini.val=ini.val, simpver=NULL,
               fig.opt=TRUE, control=list(reltol=1e-20, maxit=20000),
               subdivisions = 100L, rel.tol=.Machine$double.eps^0.25,
```

```
                abs.tol=.Machine$double.eps^0.25, stop.on.error=TRUE,
                keep.xy=FALSE, aux=NULL)

  xopt0   <- seq(100, 150, by=5)
  ini.val <- list(0.035, xopt0, 200, 1)
  resu3   <- fitsigmoid(MbetaE, x=t0, y=m0, ini.val=ini.val, simpver=1, fig.opt=TRUE)

  m.ini   <- c(0.5, 1, 2, 3, 4, 5, 10, 20)
  ini.val <- list(1e-8, m.ini, 200, 1)
  resu3   <- fitsigmoid(MBriereE, x=t0, y=m0, ini.val=ini.val, simpver=1,
              fig.opt=TRUE, control=list(reltol=1e-20, maxit=20000, trace=FALSE),
              subdivisions=100L, rel.tol=.Machine$double.eps^0.25,
              abs.tol=.Machine$double.eps^0.25, stop.on.error=TRUE,
              keep.xy=FALSE, aux=NULL)


graphics.off()
```

`fracdim` *Calculation of Fractal Dimension of Lef Veins Based on the Box-Counting Method*

Description

`fracdim` is used to calculate the fractal dimension of leaf veins based on the box-counting method.

Usage

```
fracdim(x, y, frac.fig = TRUE, denomi.range = seq(8, 30, by=1),
        ratiox = 0.02, ratioy = 0.08, main = NULL)
```

Arguments

`x`	the x coordinates of leaf-vein pixels.
`y`	the y coordinates of leaf-vein pixels.
`frac.fig`	the option of drawing the results of the linear fitting.
`denomi.range`	the number of equidistant segments of the maximum range between the range of the x coordinates and that of the y coordinates.
`ratiox`	the the x coordinate of the location parameter for positioning the legend.
`ratioy`	the the y coordinate of the location parameter for positioning the legend.
`main`	the main title of the figure.

Details

The box-counting approach uses a group of boxes (squares for simplicity) with different sizes (δ) to divide the leaf vein image into different parts. Let N represent the number of boxes that include at least one pixel of leaf vein. The maximum of the range of the x coordinates and the range of the y coordinates for leaf-vein pixels is defined as z. Let δ represent the vector of z/`denomi.range`. Then, we used the following equation to calculate the fractal dimension of leaf veins:

$$\ln N = a + b \ln\left(\delta^{-1}\right),$$

where b is the theoretical value of the fractal dimension. We can use its estimate as the numerical value of the fractal dimension for a leaf venation network.

Value

`a`	the estimate of the intercept.
`sd.a`	the standard deviation of the estimated intercept.
`lci.a`	the lower bound of the 95% confidence interval of the estimated intercept.
`uci.a`	the upper bound of the 95% confidence interval of the estimated intercept.
`b`	the estimate of the slope.
`sd.b`	the standard deviation of the estimated slope.
`lci.a`	the lower bound of the 95% confidence interval of the estimated slope.
`uci.a`	the upper bound of the 95% confidence interval of the estimated slope.
`r.sq`	the coefficient of determination.
`delta`	the vector of box sizes.
`N`	the number of boxes that include at least one pixel of leaf vein.

Note

Here, `x` and `y` cannot be adjusted by the `adjdata` function because the leaf veins are not the leaf's boundary data.

Author(s)

Peijian Shi <`pjshi@njfu.edu.cn`>, Johan Gielis <`johan.gielis@uantwerpen.be`>, Brady K. Quinn <`Brady.Quinn@dfo-mpo.gc.ca`>.

References

Shi, P., Gielis, J., Quinn, B.K., Niklas, K.J., Ratkowsky, D.A., Schrader, J., Ruan, H., Wang, L., Niinemets, Ü. (2022) 'biogeom': An R package for simulating and fitting natural shapes. *Annals of the New York Academy of Sciences* 1516, 123−134. doi:10.1111/nyas.14862

Shi, P., Yu, K., Niinemets, Ü., Gielis, J. (2021) Can leaf shape be represented by the ratio of leaf width to length? Evidence from nine species of *Magnolia* and *Michelia* (Magnoliaceae). *Forests* 12, 41. doi:10.3390/f12010041

Vico, P.G., Kyriacos, S., Heymans, O., Louryan, S., Cartilier, L. (1998) Dynamic study of the extraembryonic vascular network of the chick embryo by fractal analysis. *Journal of Theoretical Biology* 195, 525−532. doi:10.1006/jtbi.1998.0810

See Also

veins

Examples

```
data(veins)

dev.new()
plot(veins$x, veins$y, cex=0.01, asp=1, cex.lab=1.5, cex.axis=1.5,
     xlab=expression(italic("x")), ylab=expression(italic("y")))

fracdim(veins$x, veins$y)

graphics.off()
```

GE *Calculation of the Polar Radius of the Gielis Curve*

Description

`GE` is used to calculate polar radii of the original Gielis equation or one of its simplified versions at given polar angles.

Usage

```
GE(P, phi, m = 1, simpver = NULL, nval = 1)
```

Arguments

`P`	the parameters of the original Gielis equation or one of its simplified versions.
`phi`	the polar angle(s).
`m`	the given m value that determines the number of angles of the Gielis curve within $[0, 2\pi)$.
`simpver`	an optional argument to use the simplified version of the original Gielis equation.
`nval`	the specified value for n_1 or n_2 or n_3 in the simplified versions.

Details

When `simpver = NULL`, the original Gielis equation is selected:

$$r\left(\varphi\right) = a\left(\left|\cos\left(\frac{m}{4}\varphi\right)\right|^{n_2} + \left|\frac{1}{k}\sin\left(\frac{m}{4}\varphi\right)\right|^{n_3}\right)^{-\frac{1}{n_1}},$$

where r represents the polar radius at the polar angle φ; m determines the number of angles within $[0, 2\pi)$; and a, k, n_1, n_2, and n_3 need to be provided in `P`.

When `simpver = 1`, the simplified version 1 is selected:

$$r\left(\varphi\right)=a\left(\left|\cos\left(\frac{m}{4}\varphi\right)\right|^{n_2}+\left|\sin\left(\frac{m}{4}\varphi\right)\right|^{n_2}\right)^{-\frac{1}{n_1}},$$

where a, n_1, and n_2 need to be provided in `P`.

When `simpver = 2`, the simplified version 2 is selected:

$$r\left(\varphi\right)=a\left(\left|\cos\left(\frac{m}{4}\varphi\right)\right|^{n_2}+\left|\sin\left(\frac{m}{4}\varphi\right)\right|^{n_2}\right)^{-\frac{1}{n_1}},$$

where a and n_1 need to be provided in `P`, and n_2 should be specified in `nval`.

When `simpver = 3`, the simplified version 3 is selected:

$$r\left(\varphi\right)=a\left(\left|\cos\left(\frac{m}{4}\varphi\right)\right|^{n_1}+\left|\sin\left(\frac{m}{4}\varphi\right)\right|^{n_1}\right)^{-\frac{1}{n_1}},$$

where a needs to be provided in `P`, and n_1 should be specified in `nval`.

When `simpver = 4`, the simplified version 4 is selected:

$$r\left(\varphi\right)=a\left(\left|\cos\left(\frac{m}{4}\varphi\right)\right|^{n_1}+\left|\sin\left(\frac{m}{4}\varphi\right)\right|^{n_1}\right)^{-\frac{1}{n_1}},$$

where a and n_1 need to be provided in `P`.

When `simpver = 5`, the simplified version 5 is selected:

$$r\left(\varphi\right)=a\left(\left|\cos\left(\frac{m}{4}\varphi\right)\right|^{n_2}+\left|\sin\left(\frac{m}{4}\varphi\right)\right|^{n_3}\right)^{-\frac{1}{n_1}},$$

where a, n_1, n_2, and n_3 need to be provided in `P`.

When `simpver = 6`, the simplified version 6 is selected:

$$r\left(\varphi\right)=a\left(\left|\cos\left(\frac{m}{4}\varphi\right)\right|^{n_2}+\left|\frac{1}{k}\sin\left(\frac{m}{4}\varphi\right)\right|^{n_2}\right)^{-\frac{1}{n_1}},$$

where a, k, n_1, and n_2 need to be provided in `P`.

When `simpver = 7`, the simplified version 7 is selected:

$$r\left(\varphi\right)=a\left(\left|\cos\left(\frac{m}{4}\varphi\right)\right|^{n_2}+\left|\frac{1}{k}\sin\left(\frac{m}{4}\varphi\right)\right|^{n_2}\right)^{-\frac{1}{n_1}},$$

where a, k, and n_1 need to be provided in `P`, and n_2 should be specified in `nval`.

When `simpver = 8`, the simplified version 8 is selected:

$$r\left(\varphi\right)=a\left(\left|\cos\left(\frac{m}{4}\varphi\right)\right|^{n_1}+\left|\frac{1}{k}\sin\left(\frac{m}{4}\varphi\right)\right|^{n_1}\right)^{-\frac{1}{n_1}},$$

where a and k are parameters that need to be provided in `P`, and n_1 should be specified in `nval`.

When `simpver = 9`, the simplified version 9 is selected:

$$r\left(\varphi\right)=a\left(\left|\cos\left(\frac{m}{4}\varphi\right)\right|^{n_1}+\left|\frac{1}{k}\sin\left(\frac{m}{4}\varphi\right)\right|^{n_1}\right)^{-\frac{1}{n_1}},$$

where a, k, and n_1 need to be provided in `P`.

Value

The polar radii predicted by the original Gielis equation or one of its simplified versions.

Note

`simpver` here is different from that in the `TGE` function.

Author(s)

Peijian Shi <`pjshi@njfu.edu.cn`>, Johan Gielis <`johan.gielis@uantwerpen.be`>, Brady K. Quinn <`Brady.Quinn@dfo-mpo.gc.ca`>.

References

Gielis, J. (2003) A generic geometric transformation that unifies a wide range of natural and abstract shapes. *American Journal of Botany* 90, 333−338. doi:10.3732/ajb.90.3.333

Li, Y., Quinn, B.K., Gielis, J., Li, Y., Shi, P. (2022) Evidence that supertriangles exist in nature from the vertical projections of *Koelreuteria paniculata* fruit. *Symmetry* 14, 23. doi:10.3390/sym14010023

Shi, P., Gielis, J., Quinn, B.K., Niklas, K.J., Ratkowsky, D.A., Schrader, J., Ruan, H., Wang, L., Niinemets, Ü. (2022) 'biogeom': An R package for simulating and fitting natural shapes. *Annals of the New York Academy of Sciences* 1516, 123−134. doi:10.1111/nyas.14862

Shi, P., Ratkowsky, D.A., Gielis, J. (2020) The generalized Gielis geometric equation and its application. *Symmetry* 12, 645. doi:10.3390/sym12040645

Shi, P., Xu, Q., Sandhu, H.S., Gielis, J., Ding, Y., Li, H., Dong, X. (2015) Comparison of dwarf bamboos (*Indocalamus* sp.) leaf parameters to determine relationship between spatial density of plants and total leaf area per plant. *Ecology and Evolution* 5, 4578−4589. doi:10.1002/ece3.1728

See Also

`areaGE`, `curveGE`, `DSGE`, `fitGE`, `SurfaceAreaSGE`, `TGE`, `VolumeSGE`

Examples

```
GE.par  <- c(2, 1, 4, 6, 3)
varphi.vec <- seq(0, 2*pi, len=2000)
r.theor <- GE(P=GE.par, phi=varphi.vec, m=5)

dev.new()
plot( varphi.vec, r.theor, cex.lab=1.5, cex.axis=1.5,
      xlab=expression(italic(varphi)), ylab=expression(italic("r")),
      type="l", col=4 )

graphics.off()
```

`ginkgoseed` *Boundary Data of the Side Projections of* Ginkgo biloba *Seeds*

Description

The data consist of the boundary data of four side projections of *G. biloba* (Cultivar 'Fozhi') seeds sampled at Nanjing Forestry University campus on September 23, 2021.

Usage

```
data(ginkgoseed)
```

Details

In the data set, there are three columns of variables: `Code`, `x`, and `y`. `Code` saves the codes of individual fruit; `x` saves the x coordinates of the side projections of seeds in the Cartesian coordinate system (cm); and `y` saves the y coordinates of the side projections of seeds in the Cartesian coordinate system (cm).

References

Tian, F., Wang, Y., Sandhu, H.S., Gielis, J., Shi, P. (2020) Comparison of seed morphology of two ginkgo cultivars. *Journal of Forestry Research* 31, 751–758. doi:10.1007/s116760180770y

Examples

```
data(ginkgoseed)

uni.C <- sort( unique(ginkgoseed$Code) )
ind   <- 1
Data  <- ginkgoseed[ginkgoseed$Code==uni.C[ind], ]
x0    <- Data$x
y0    <- Data$y

Res1 <- adjdata(x0, y0, ub.np=2000, len.pro=1/20)
dev.new()
plot( Res1$x, Res1$y, asp=1, cex.lab=1.5, cex.axis=1.5, type="l",
      xlab=expression(italic("x")), ylab=expression(italic("y")) )


  x1        <- Res1$x
  y1        <- Res1$y
  x0.ini    <- mean( x1 )
  y0.ini    <- mean( y1 )
  theta.ini <- pi/4
  a.ini     <- 1
  n1.ini    <- seq(0.6, 1, by=0.1)
  n2.ini    <- 1
  n3.ini    <- 1
```

```
  ini.val   <- list(x0.ini, y0.ini, theta.ini,
                    a.ini, n1.ini, n2.ini, n3.ini)

  Res2 <- fitGE( GE, x=x1, y=y1, ini.val=ini.val,
                 m=2, simpver=5, nval=1, unit="cm",
                 par.list=FALSE, fig.opt=TRUE, angle=NULL,
                 control=list(reltol=1e-20, maxit=20000),
                 np=2000 )


graphics.off()
```

`kp` *Boundary Data of the Vertical Projections of* Koelreuteria paniculata *Fruit*

Description

The data consist of the boundary data of four vertical projections of *K. paniculata* fruit sampled at Nanjing Forestry University campus in early October 2021.

Usage

```
data(kp)
```

Details

In the data set, there are three columns of variables: `Code`, `x`, and `y`. `Code` saves the codes of individual fruit; `x` saves the x coordinates of the vertical projections of fruit in the Cartesian coordinate system (cm); and `y` saves the y coordinates of the vertical projections of fruit in the Cartesian coordinate system (cm).

References

Li, Y., Quinn, B.K., Gielis, J., Li, Y., Shi, P. (2022) Evidence that supertriangles exist in nature from the vertical projections of *Koelreuteria paniculata* fruit. *Symmetry* 14, 23. doi:10.3390/sym14010023

Examples

```
data(kp)

uni.C <- sort( unique(kp$Code) )
ind   <- 1
Data  <- kp[kp$Code==uni.C[ind], ]
x0    <- Data$x
y0    <- Data$y

Res1 <- adjdata(x0, y0, ub.np=200, len.pro=1/20)
```

```
dev.new()
plot( Res1$x, Res1$y, asp=1, cex.lab=1.5, cex.axis=1.5, type="l",
      xlab=expression(italic("x")), ylab=expression(italic("y")) )


  x1        <- Res1$x
  y1        <- Res1$y
  x0.ini    <- mean( x1 )
  y0.ini    <- mean( y1 )
  theta.ini <- pi
  a.ini     <- 0.9
  n1.ini    <- c(1, 4)
  n2.ini    <- 5
  n3.ini    <- c(5, 10, 15)
  ini.val   <- list(x0.ini, y0.ini, theta.ini,
                    a.ini, n1.ini, n2.ini, n3.ini)

  Res2 <- fitGE( GE, x=x1, y=y1, ini.val=ini.val,
                 m=3, simpver=5, nval=1, unit="cm",
                 par.list=FALSE, fig.opt=TRUE, angle=NULL,
                 control=list(reltol=1e-20, maxit=20000),
                 np=2000 )


graphics.off()
```

`lmPE` *Parameter Estimation for the Todd-Smart Equation*

Description

`lmPE` is used to estimate the parameters of the Todd-Smart equation using the multiple linear regression.

Usage

```
lmPE(x, y, simpver = NULL, dev.angle = NULL, weights = NULL, fig.opt = TRUE,
     prog.opt = TRUE, xlim = NULL, ylim = NULL, unit = NULL, main = NULL,
     extr.method = "Shi")
```

Arguments

`x`	the x coordinates of the edge of an egg's boundary.
`y`	the y coordinates of the edge of an egg's boundary.
`simpver`	an optional argument to use the simplified version of the original Todd-Smart equation.
`dev.angle`	the angle of deviation for the axis associated with the maximum distance between two points on an egg's profile from the mid-line of the egg's profile.

`weights`	the weights for the multiple linear regression.
`fig.opt`	an optional argument to draw the observed and predicted egg's boundaries.
`prog.opt`	an optional argument to show the running progress for different values of `dev.angle`, when `dev.angle` is not `NULL`.
`xlim`	the range of the x-axis over which to plot the Todd-Smart curve.
`ylim`	the range of the y-axis over which to plot the Todd-Smart curve.
`unit`	the units of the x-axis and the y-axis when showing the Todd-Smart curve.
`main`	the main title of the figure.
`extr.method`	an optional argument to fit the planar coordinate data of an egg's profile extracted using different methods.

Details

There are two methods to obtain the major axis (i.e., the mid-line) of an egg's profile: the maximum distance method, and the longest axis adjustment method. In the first method, the straight line through two points forming the maximum distance on the egg's boundary is defined as the major axis. In the second method, we assume that there is an angle of deviation for the longest axis (i.e., the axis associated with the maximum distance between two points on an egg's profile) from the mid-line of the egg's profile. That is to say, the mid-line of an egg's profile is not the axis associated with the maximum distance between two points on the egg's profile. When `dev.angle = NULL`, the maximum distance method is used; when `dev.angle` is a numerical value or a numerical vector, the longest axis adjustment method is used. Here, the numerical value of `dev.angle` is not the angle of deviation for the major axis of an egg's profile from the x-axis, and instead it is the angle of deviation for the longest axis (associated with the maximum distance between two points on the egg's profile) from the mid-line of the egg's profile. It is better to take the option of `extr.method = "Shi"` for correctly fitting the planar coordinate data of an egg's profile extracted using the protocols proposed by Shi et al. (2015, 2018) (and also see Su et al. (2019)), while it is better to take the option of `extr.method = "Biggins"` for correctly fitting the planar coordinate data of an egg's profile extracted using the protocols proposed by Biggins et al. (2018). For the planar coordinate data extracted using the protocols of Biggins et al. (2018), there are fewer data points on the two ends of the mid-line than other parts of the egg's profile, which means that the range of the observed x values might be smaller than the actual egg's length. A group of equidistant x values are set along the mid-line, and each x value corresponds to two y values that are respectively located at the upper and lower sides of the egg's profile. Because of the difference in the curvature for differnt parts of the egg's profile, the equidistant x values cannot render the extracted data points on the egg's profile to be regular. For the planar coordinate data extracted using the protocols of Shi et al. (2015, 2018), the data points are more regularly distributed on the egg's profile (perimeter) than those of Biggins et al. (2018), although the x values of the data points along the mid-line are not equidistant.

Value

`lm.tse`	the fitted results of the multiple linear regression.
`par`	the estimates of the four model parameters in the Todd-Smart equation.
`theta`	the angle between the longest axis of an egg's profile (i.e., the axis associated with the maximum distance between two points on the egg's profile) and the x-axis.

`epsilon`	the optimal angle of deviation for the longest axis (associated with the maximum distance between two points on an egg's profile) from the mid-line of the egg's profile, when `dev.angle` is not `NULL`.
`RSS.vector`	the vector of residual sum of squares corresponding to `dev.angle`, when `dev.angle` is not `NULL`.
`x.obs`	the observed x coordinates.
`y.obs`	the observed y coordinates.
`y.pred`	the predicted y coordinates corresponding to the the observed x coordinates.
`x.stand.obs`	the observed x coordinates when the egg length is fixed to be 2 ranging from -1 to 1.
`y.stand.obs`	the observed y coordinates when the egg length is fixed to be 2 ranging from -1 to 1.
`y.stand.pred`	the predicted y coordinates corresponding to the the observed x coordinates, when the egg length is fixed to be 2 ranging from -1 to 1.
`scan.length`	the length of the egg's boundary. The default is the maximum distance between two points on the egg's boundary.
`scan.width`	the maximum width of the egg's boundary.
`scan.area`	the area of the egg's boundary.
`scan.perimeter`	the perimeter of the egg's boundary based on all data points on the egg's boundary.
`RSS.scaled`	the residual sum of squares between the observed and predicted y values for a scaled egg's profile whose length equals 2.
`RSS`	the residual sum of squares between the observed and predicted y values.
`sample.size`	the number of data points used in the numerical calculation.
`RMSE.scaled`	the root-mean-square errors between the observed and predicted y values for a scaled egg's profile whose length equals 2.
`RMSE`	the root-mean-square errors between the observed and predicted y values.

Note

`theta` is the calculated angle between the longest axis (i.e., the axis associated with the maximum distance between two points on an egg's profile) and the x-axis, and `epsilon` is the calculated angle of deviation for the longest axis from the mid-line of the egg's profile. This means that the angle between the mid-line and the x-axis is equal to `theta + epsilon`. Here, `RSS`, and `RMSE` are for the observed and predicted y coordinates of the egg's profile, not for those when the egg's length is scaled to 2. There are two figures when `fig.opt = TRUE`: (i) the observed and predicted egg's boundaries when the egg's length is scaled to 2, and (ii) the observed and predicted egg's boundaries at their actual scales.

Author(s)

Peijian Shi <`pjshi@njfu.edu.cn`>, Johan Gielis <`johan.gielis@uantwerpen.be`>, Brady K. Quinn <`Brady.Quinn@dfo-mpo.gc.ca`>.

References

Biggins, J.D., Montgomeries, R.M., Thompson, J.E., Birkhead, T.R. (2022) Preston's universal formula for avian egg shape. *Ornithology* In press. doi:10.1093/ornithology/ukac028

Biggins, J.D., Thompson, J.E., Birkhead, T.R. (2018) Accurately quantifying the shape of birds' eggs. *Ecology and Evolution* 8, 9728−9738. doi:10.1002/ece3.4412

Nelder, J.A., Mead, R. (1965). A simplex method for function minimization. *Computer Journal* 7, 308−313. doi:10.1093/comjnl/7.4.308

Preston, F.W. (1953) The shapes of birds' eggs. *The Auk* 70, 160−182.

Shi, P., Gielis, J., Quinn, B.K., Niklas, K.J., Ratkowsky, D.A., Schrader, J., Ruan, H., Wang, L., Niinemets, Ü. (2022) 'biogeom': An R package for simulating and fitting natural shapes. *Annals of the New York Academy of Sciences* 1516, 123−134. doi:10.1111/nyas.14862

Shi, P., Huang, J., Hui, C., Grissino-Mayer, H.D., Tardif, J., Zhai, L., Wang, F., Li, B. (2015) Capturing spiral radial growth of conifers using the superellipse to model tree-ring geometric shape. *Frontiers in Plant Science* 6, 856. doi:10.3389/fpls.2015.00856

Shi, P., Niinemets, Ü., Hui, C., Niklas, K.J., Yu, X., Hölscher, D. (2020) Leaf bilateral symmetry and the scaling of the perimeter vs. the surface area in 15 vine species. *Forests* 11, 246. doi:10.3390/f11020246

Shi, P., Ratkowsky, D.A., Li, Y., Zhang, L., Lin, S., Gielis, J. (2018) General leaf-area geometric formula exists for plants - Evidence from the simplified Gielis equation. *Forests* 9, 714. doi:10.3390/f9110714

Shi, P., Wang, L., Quinn, B.K., Gielis, J. (2022) A new program to estimate the parameters of Preston's equation, a general formula for describing the egg shape of birds. *Symmetry* Under review.

Su, J., Niklas, K.J., Huang, W., Yu, X., Yang, Y., Shi, P. (2019) Lamina shape does not correlate with lamina surface area: An analysis based on the simplified Gielis equation. *Global Ecology and Conservation* 19, e00666. doi:10.1016/j.gecco.2019.e00666

Todd, P.H., Smart, I.H.M. (1984) The shape of birds' eggs. *Journal of Theoretical Biology* 106, 239−243. doi:10.1016/00225193(84)900213

See Also

`curveEPE`, `fitEPE`, `PE`, `TSE`

Examples

```
data(eggs)

uni.C <- sort( unique(eggs$Code) )
ind   <- 8
Data  <- eggs[eggs$Code==uni.C[ind], ]
x0    <- Data$x
y0    <- Data$y

Res1  <- adjdata(x0, y0, ub.np=3000, len.pro=1/20)
x1    <- Res1$x
y1    <- Res1$y
```

```
dev.new()
plot( Res1$x, Res1$y, asp=1, cex.lab=1.5, cex.axis=1.5,
      xlab=expression(italic("x")), ylab=expression(italic("y")) )


  Res2 <- lmPE(x1, y1, simpver=NULL, dev.angle=NULL, unit="cm")
  summary( Res2$lm.tse )
  Res2$RMSE.scaled / 2

  if(FALSE){
    dev.new()
    xg1 <- seq(-1, 1, len=1000)
    yg1 <- TSE(P=Res2$par, x=xg1, simpver=NULL)
    xg2 <- seq(1, -1, len=1000)
    yg2 <- -TSE(P=Res2$par, x=xg2, simpver=NULL)
    plot(xg1, yg1, asp=1, type="l", col=2, ylim=c(-1,1), cex.lab=1.5, cex.axis=1.5,
      xlab=expression(italic(x)), ylab=expression(italic(y)))
    lines(xg2, yg2, col=4)

    dev.new()
    plot(Res2$x.obs, Res2$y.obs, asp=1, cex.lab=1.5, cex.axis=1.5,
      xlab=expression(italic(x)), ylab=expression(italic(y)), type="l")
    lines(Res2$x.obs, Res2$y.pred, col=2)

    dev.new()
    plot(Res2$x.stand.obs, Res2$y.stand.obs, asp=1, cex.lab=1.5, cex.axis=1.5,
      xlab=expression(italic(x)), ylab=expression(italic(y)), type="l")
    lines(Res2$x.stand.obs, Res2$y.stand.pred, col=2)
  }

  Res3 <- lmPE(x1, y1, simpver=NULL, dev.angle=seq(-0.05, 0.05, by=0.0001), unit="cm")
  summary( Res3$lm.tse )
  Res3$epsilon
  Res3$RMSE.scaled / 2

  Res4 <- lmPE(x1, y1, simpver=1, dev.angle=NULL, unit="cm")
  summary( Res4$lm.tse )



graphics.off()
```

`lmTE` *Parameter Estimation for the Troscianko Equation*

Description

`lmTE` is used to estimate the parameters of the Troscianko equation using the multiple linear regression, and the estimated values of the parameters are only used as the initial values for using the `fitETE` function

Usage

```
lmTE(x, y, dev.angle = NULL, weights = NULL, fig.opt = TRUE,
     prog.opt = TRUE, xlim = NULL, ylim = NULL, unit = NULL, main = NULL)
```

Arguments

`x`	the x coordinates of the edge of an egg's boundary.
`y`	the y coordinates of the edge of an egg's boundary.
`dev.angle`	the angle of deviation for the axis associated with the maximum distance between two points on an egg's profile from the mid-line of the egg's profile.
`weights`	the weights for the multiple linear regression.
`fig.opt`	an optional argument to draw the observed and predicted egg's boundaries.
`prog.opt`	an optional argument to show the running progress for different values of `dev.angle`, when `dev.angle` is not `NULL`.
`xlim`	the range of the x-axis over which to plot the Troscianko curve.
`ylim`	the range of the y-axis over which to plot the Troscianko curve.
`unit`	the units of the x-axis and the y-axis when showing the Troscianko curve.
`main`	the main title of the figure.

Details

The estimated values of the parameters using the `lmTE` function tend to be NOT globally optimal, and the values are only used as the initial values for using the `fitETE` function. There are two methods to obtain the major axis (i.e., the mid-line) of an egg's profile: the maximum distance method, and the longest axis adjustment method. In the first method, the straight line through two points forming the maximum distance on the egg's boundary is defined as the major axis. In the second method, we assume that there is an angle of deviation for the longest axis (i.e., the axis associated with the maximum distance between two points on an egg's profile) from the mid-line of the egg's profile. That is to say, the mid-line of an egg's profile is not the axis associated with the maximum distance between two points on the egg's profile. When `dev.angle = NULL`, the maximum distance method is used; when `dev.angle` is a numerical value or a numerical vector, the longest axis adjustment method is used. Here, the numerical value of `dev.angle` is not the angle of deviation for the major axis of an egg's profile from the x-axis, and instead it is the angle of deviation for the longest axis (associated with the maximum distance between two points on the egg's profile) from the mid-line of the egg's profile. The planar coordinate data of an egg's profile are extracted using the protocols proposed by Shi et al. (2015, 2018) (and also see Su et al. (2019)). For the planar coordinate data extracted using the protocols of Shi et al. (2015, 2018), the data points are more regularly distributed on the egg's profile (perimeter), although the x values of the data points along the mid-line are not equidistant.

Value

`lm.te`	the fitted results of the multiple linear regression.
`par`	the estimates of the four model parameters in the Troscianko equation.

`theta`	the angle between the longest axis of an egg's profile (i.e., the axis associated with the maximum distance between two points on the egg's profile) and the x-axis.
`epsilon`	the optimal angle of deviation for the longest axis (associated with the maximum distance between two points on an egg's profile) from the mid-line of the egg's profile, when `dev.angle` is not `NULL`.
`RSS.vector`	the vector of residual sum of squares corresponding to `dev.angle`, when `dev.angle` is not `NULL`.
`x.obs`	the observed x coordinates.
`y.obs`	the observed y coordinates.
`y.pred`	the predicted y coordinates corresponding to the the observed x coordinates.
`x.stand.obs`	the observed x coordinates when the egg length is fixed to be 2 ranging from -1 to 1.
`y.stand.obs`	the observed y coordinates when the egg length is fixed to be 2 ranging from -1 to 1.
`y.stand.pred`	the predicted y coordinates corresponding to the the observed x coordinates, when the egg length is fixed to be 2 ranging from -1 to 1.
`scan.length`	the length of the egg's boundary. The default is the maximum distance between two points on the egg's boundary.
`scan.width`	the maximum width of the egg's boundary.
`scan.area`	the area of the egg's boundary.
`scan.perimeter`	the perimeter of the egg's boundary based on all data points on the egg's boundary.
`RSS.scaled`	the residual sum of squares between the observed and predicted y values for a scaled egg's profile whose length equals 2.
`RSS`	the residual sum of squares between the observed and predicted y values.
`sample.size`	the number of data points used in the numerical calculation.
`RMSE.scaled`	the root-mean-square errors between the observed and predicted y values for a scaled egg's profile whose length equals 2.
`RMSE`	the root-mean-square errors between the observed and predicted y values.

Note

`theta` is the calculated angle between the longest axis (i.e., the axis associated with the maximum distance between two points on an egg's profile) and the x-axis, and `epsilon` is the calculated angle of deviation for the longest axis from the mid-line of the egg's profile. This means that the angle between the mid-line and the x-axis is equal to `theta + epsilon`. Here, `RSS`, and `RMSE` are for the observed and predicted y coordinates of the egg's profile, not for those when the egg's length is scaled to 2. There are two figures when `fig.opt = TRUE`: (i) the observed and predicted egg's boundaries when the egg's length is scaled to 2, and (ii) the observed and predicted egg's boundaries at their actual scales.

Author(s)

Peijian Shi <`pjshi@njfu.edu.cn`>, Johan Gielis <`johan.gielis@uantwerpen.be`>, Brady K. Quinn <`Brady.Quinn@dfo-mpo.gc.ca`>.

References

Biggins, J.D., Montgomeries, R.M., Thompson, J.E., Birkhead, T.R. (2022) Preston's universal formula for avian egg shape. *Ornithology* In press. doi:10.1093/ornithology/ukac028

Biggins, J.D., Thompson, J.E., Birkhead, T.R. (2018) Accurately quantifying the shape of birds' eggs. *Ecology and Evolution* 8, 9728−9738. doi:10.1002/ece3.4412

Nelder, J.A., Mead, R. (1965). A simplex method for function minimization. *Computer Journal* 7, 308−313. doi:10.1093/comjnl/7.4.308

Shi, P., Huang, J., Hui, C., Grissino-Mayer, H.D., Tardif, J., Zhai, L., Wang, F., Li, B. (2015) Capturing spiral radial growth of conifers using the superellipse to model tree-ring geometric shape. *Frontiers in Plant Science* 6, 856. doi:10.3389/fpls.2015.00856

Shi, P., Ratkowsky, D.A., Li, Y., Zhang, L., Lin, S., Gielis, J. (2018) General leaf-area geometric formula exists for plants - Evidence from the simplified Gielis equation. *Forests* 9, 714. doi:10.3390/f9110714

Shi, P., Wang, L., Quinn, B.K., Gielis, J. (2022) A new program to estimate the parameters of Preston's equation, a general formula for describing the egg shape of birds. *Symmetry* Under review.

Su, J., Niklas, K.J., Huang, W., Yu, X., Yang, Y., Shi, P. (2019) Lamina shape does not correlate with lamina surface area: An analysis based on the simplified Gielis equation. *Global Ecology and Conservation* 19, e00666. doi:10.1016/j.gecco.2019.e00666

Troscianko, J. (2014). A simple tool for calculating egg shape, volume and surface area from digital images. *Ibis*, 156, 874−878. doi:10.1111/ibi.12177

See Also

`fitETE`, `TE`

Examples

```
data(eggs)

uni.C <- sort( unique(eggs$Code) )
ind   <- 8
Data  <- eggs[eggs$Code==uni.C[ind], ]
x0    <- Data$x
y0    <- Data$y

Res1  <- adjdata(x0, y0, ub.np=3000, len.pro=1/20)
x1    <- Res1$x
y1    <- Res1$y

dev.new()
plot( Res1$x, Res1$y, asp=1, cex.lab=1.5, cex.axis=1.5,
      xlab=expression(italic("x")), ylab=expression(italic("y")) )
```

```
  Res2 <- lmTE(x1, y1, dev.angle=NULL, unit="cm")
  summary( Res2$lm.te )
  Res2$RMSE.scaled / 2

  if(FALSE){
    dev.new()
    xg1 <- seq(-1, 1, len=1000)
    yg1 <- TE(P=Res2$par, x=xg1)
    xg2 <- seq(1, -1, len=1000)
    yg2 <- -TE(P=Res2$par, x=xg2)
    plot(xg1, yg1, asp=1, type="l", col=2, ylim=c(-1,1), cex.lab=1.5, cex.axis=1.5,
      xlab=expression(italic(x)), ylab=expression(italic(y)))
    lines(xg2, yg2, col=4)

    dev.new()
    plot(Res2$x.obs, Res2$y.obs, asp=1, cex.lab=1.5, cex.axis=1.5,
      xlab=expression(italic(x)), ylab=expression(italic(y)), type="l")
    lines(Res2$x.obs, Res2$y.pred, col=2)

    dev.new()
    plot(Res2$x.stand.obs, Res2$y.stand.obs, asp=1, cex.lab=1.5, cex.axis=1.5,
      xlab=expression(italic(x)), ylab=expression(italic(y)), type="l")
    lines(Res2$x.stand.obs, Res2$y.stand.pred, col=2)
  }

  Res3 <- lmTE(x1, y1, dev.angle=seq(-0.05, 0.05, by=0.0001), unit="cm")
  summary( Res3$lm.te )
  Res3$epsilon
  Res3$RMSE.scaled / 2


graphics.off()
```

`MbetaE` *Modified Beta Equation*

Description

`MbetaE` is used to calculate y values at given x values using the modified beta equation or one of its simplified versions.

Usage

```
MbetaE(P, x, simpver = 1)
```

Arguments

`P`	the parameters of the modified beta equation or one of its simplified versions.
`x`	the given x values.
`simpver`	an optional argument to use the simplified version of the modified beta equation.

Details

When `simpver = NULL`, the modified beta equation is selected:

$$\text{if } x \in (x_{\mathrm{min}}, x_{\mathrm{max}}),$$

$$y = y_{\mathrm{opt}}\left[\left(\frac{x_{\mathrm{max}} - x}{x_{\mathrm{max}} - x_{\mathrm{opt}}}\right)\left(\frac{x - x_{\mathrm{min}}}{x_{\mathrm{opt}} - x_{\mathrm{min}}}\right)^{\frac{x_{\mathrm{opt}} - x_{\mathrm{min}}}{x_{\mathrm{max}} - x_{\mathrm{opt}}}}\right]^{\delta};$$

$$\text{if } x \notin (x_{\mathrm{min}}, x_{\mathrm{max}}),$$

$$y = 0.$$

Here, x and y represent the independent and dependent variables, respectively; y_{opt}, x_{opt}, x_{min}, and x_{max} are constants to be estimated; y_{opt} represents the maximum y, and x_{opt} is the x value associated with the maximum y (i.e., y_{opt}); and x_{min} and x_{max} represent the lower and upper intersections between the curve and the x-axis. y is defined as 0 when $x < x_{\mathrm{min}}$ or $x > x_{\mathrm{max}}$. There are five elements in `P`, representing the values of y_{opt}, x_{opt}, x_{min}, x_{max}, and δ, respectively.

When `simpver = 1`, the simplified version 1 is selected:

$$\text{if } x \in (0, x_{\mathrm{max}}),$$

$$y = y_{\mathrm{opt}}\left[\left(\frac{x_{\mathrm{max}} - x}{x_{\mathrm{max}} - x_{\mathrm{opt}}}\right)\left(\frac{x}{x_{\mathrm{opt}}}\right)^{\frac{x_{\mathrm{opt}}}{x_{\mathrm{max}} - x_{\mathrm{opt}}}}\right]^{\delta};$$

$$\text{if } x \notin (0, x_{\mathrm{max}}),$$

$$y = 0.$$

There are four elements in `P`, representing the values of y_{opt}, x_{opt}, x_{max}, and δ, respectively.

When `simpver = 2`, the simplified version 2 is selected:

$$\text{if } x \in (x_{\mathrm{min}}, x_{\mathrm{max}}),$$

$$y = y_{\mathrm{opt}}\left(\frac{x_{\mathrm{max}} - x}{x_{\mathrm{max}} - x_{\mathrm{opt}}}\right)\left(\frac{x - x_{\mathrm{min}}}{x_{\mathrm{opt}} - x_{\mathrm{min}}}\right)^{\frac{x_{\mathrm{opt}} - x_{\mathrm{min}}}{x_{\mathrm{max}} - x_{\mathrm{opt}}}};$$

$$\text{if } x \notin (x_{\mathrm{min}}, x_{\mathrm{max}}),$$

$$y = 0.$$

There are four elements in `P`, representing the values of y_{opt}, x_{opt}, x_{min}, and x_{max}, respectively.

When `simpver = 3`, the simplified version 3 is selected:

$$\text{if } x \in (0, x_{\mathrm{max}}),$$

$$y = y_{\text{opt}} \left(\frac{x_{\text{max}} - x}{x_{\text{max}} - x_{\text{opt}}} \right) \left(\frac{x}{x_{\text{opt}}} \right)^{\frac{x_{\text{opt}}}{x_{\text{max}} - x_{\text{opt}}}};$$

$$\text{if } x \notin (0, x_{\text{max}}),$$

$$y = 0.$$

There are three elements in `P`, representing the values of y_{opt}, x_{opt}, and x_{max}, respectively.

Value

The y values predicted by the modified beta equation or one of its simplified versions.

Note

We have added a parameter δ in the original beta equation (i.e., `simpver = 2`) to increase the flexibility for data fitting.

Author(s)

Peijian Shi <`pjshi@njfu.edu.cn`>, Johan Gielis <`johan.gielis@uantwerpen.be`>, Brady K. Quinn <`Brady.Quinn@dfo-mpo.gc.ca`>.

References

Shi, P., Fan, M., Ratkowsky, D.A., Huang, J., Wu, H., Chen, L., Fang, S., Zhang, C. (2017) Comparison of two ontogenetic growth equations for animals and plants. *Ecological Modelling* 349, 1−10. doi:10.1016/j.ecolmodel.2017.01.012

Shi, P., Gielis, J., Quinn, B.K., Niklas, K.J., Ratkowsky, D.A., Schrader, J., Ruan, H., Wang, L., Niinemets, Ü. (2022) 'biogeom': An R package for simulating and fitting natural shapes. *Annals of the New York Academy of Sciences* 1516, 123−134. doi:10.1111/nyas.14862

See Also

`areaovate`, `curveovate`, `fitovate`, `MBriereE`, `MLRFE`, `sigmoid`

Examples

```
x1   <- seq(-5, 15, len=2000)
Par1 <- c(3, 3, 10, 2)
y1   <- MbetaE(P=Par1, x=x1, simpver=1)

dev.new()
plot( x1, y1,cex.lab=1.5, cex.axis=1.5, type="l",
      xlab=expression(italic(x)), ylab=expression(italic(y)) )

graphics.off()
```

`MBriereE` *Modified Briere Equation*

Description

`MBriereE` is used to calculate y values at given x values using the modified Brière equation or one of its simplified versions.

Usage

```
MBriereE(P, x, simpver = 1)
```

Arguments

`P`	the parameters of the modified Brière equation or one of its simplified versions.
`x`	the given x values.
`simpver`	an optional argument to use the simplified version of the modified Brière equation.

Details

When `simpver = NULL`, the modified Brière equation is selected:

$$\text{if } x \in (x_{\text{min}}, x_{\text{max}}),$$

$$y = a\left|x(x-x_{\text{min}})(x_{\text{max}}-x)^{1/m}\right|^{\delta};$$

$$\text{if } x \notin (x_{\text{min}}, x_{\text{max}}),$$

$$y = 0.$$

Here, x and y represent the independent and dependent variables, respectively; and a, m, x_{min}, and x_{max} are constants to be estimated, where x_{min} and x_{max} represents the lower and upper intersections between the curve and the x-axis. y is defined as 0 when $x < x_{\text{min}}$ or $x > x_{\text{max}}$. There are five elements in `P`, representing the values of a, m, x_{min}, x_{max}, and δ, respectively.

When `simpver = 1`, the simplified version 1 is selected:

$$\text{if } x \in (0, x_{\text{max}}),$$

$$y = a\left|x^{2}(x_{\text{max}}-x)^{1/m}\right|^{\delta};$$

$$\text{if } x \notin (0, x_{\text{max}}),$$

$$y = 0.$$

There are four elements in `P`, representing the values of a, m, x_{max}, and δ, respectively.

When `simpver = 2`, the simplified version 2 is selected:

$$\text{if } x \in (x_{\text{min}}, x_{\text{max}}),$$

$$y = ax(x - x_{\min})(x_{\max} - x)^{1/m};$$

$$\text{if } x \notin (x_{\min}, x_{\max}),$$

$$y = 0.$$

There are four elements in `P` representing the values of a, m, $x_{\min}$, and $x_{\max}$, respectively.

When `simpver = 3`, the simplified version 3 is selected:

$$\text{if } x \in (0, x_{\max}),$$

$$y = ax^2(x_{\max} - x)^{1/m};$$

$$\text{if } x \notin (0, x_{\max}),$$

$$y = 0.$$

There are three elements in `P` representing the values of a, m, and $x_{\max}$, respectively.

Value

The y values predicted by the modified Brière equation or one of its simplified versions.

Note

We have added a parameter δ in the original Brière equation (i.e., `simpver = 2`) to increase the flexibility for data fitting.

Author(s)

Peijian Shi <`pjshi@njfu.edu.cn`>, Johan Gielis <`johan.gielis@uantwerpen.be`>, Brady K. Quinn <`Brady.Quinn@dfo-mpo.gc.ca`>.

References

Brière, J.-F., Pracros, P, Le Roux, A.-Y., Pierre, J.-S. (1999) A novel rate model of temperature-dependent development for arthropods. *Environmental Entomology* 28, 22−29. doi:10.1093/ee/28.1.22

Cao, L., Shi, P., Li, L., Chen, G. (2019) A new flexible sigmoidal growth model. *Symmetry* 11, 204. doi:10.3390/sym11020204

Jin, J., Quinn, B.K., Shi, P. (2022) The modified Brière equation and its applications. *Plants* 11, 1769. doi:10.3390/plants11131769

Shi, P., Gielis, J., Quinn, B.K., Niklas, K.J., Ratkowsky, D.A., Schrader, J., Ruan, H., Wang, L., Niinemets, Ü. (2022) 'biogeom': An R package for simulating and fitting natural shapes. *Annals of the New York Academy of Sciences* 1516, 123−134. doi:10.1111/nyas.14862

See Also

`areaovate`, `curveovate`, `fitovate`, `MbetaE`, `MLRFE`, `sigmoid`

Examples

```
x2   <- seq(-5, 15, len=2000)
Par2 <- c(0.01, 3, 0, 10, 1)
y2   <- MBriereE(P=Par2, x=x2, simpver=NULL)

dev.new()
plot( x2, y2, cex.lab=1.5, cex.axis=1.5, type="l",
      xlab=expression(italic(x)), ylab=expression(italic(y)) )

graphics.off()
```

MLRFE *Modified Lobry-Rosso-Flandrois (LRF) Equation*

Description

`MLRFE` is used to calculate y values at given x values using the modified LRF equation or one of its simplified versions.

Usage

```
MLRFE(P, x, simpver = 1)
```

Arguments

`P`	the parameters of the modified LRF equation or one of its simplified versions.
`x`	the given x values.
`simpver`	an optional argument to use the simplified version of the modified LRF equation.

Details

When `simpver = NULL`, the modified LRF equation is selected:

$$\text{if } x \in \left(x_{\min}, \frac{x_{\min}+x_{\max}}{2}\right),$$

$$y = y_{\text{opt}} \left\{ \frac{(x-x_{\min})\,(x-x_{\max})^2}{(x_{\max}-x_{\text{opt}})\,[(x_{\max}-x_{\text{opt}})(x-x_{\text{opt}})-(x_{\min}-x_{\text{opt}})(x_{\text{opt}}+x_{\max}-2x)]} \right\}^{\delta};$$

$$\text{if } x \in \left[\frac{x_{\min}+x_{\max}}{2}, x_{\max}\right),$$

$$y = y_{\text{opt}} \left\{ \frac{(x-x_{\max})\,(x-x_{\min})^2}{(x_{\text{opt}}-x_{\min})\,[(x_{\text{opt}}-x_{\min})(x-x_{\text{opt}})-(x_{\text{opt}}-x_{\max})(x_{\text{opt}}+x_{\min}-2x)]} \right\}^{\delta};$$

$$\text{if } x \notin (x_{\min}, x_{\max}),$$

$$y = 0.$$

Here, x and y represent the independent and dependent variables, respectively; y_{opt}, x_{opt}, $x_{\min}$, and $x_{\max}$ are constants to be estimated; y_{opt} represents the maximum y, and x_{opt} is the x value associated with the maximum y (i.e., y_{opt}); and $x_{\min}$ and $x_{\max}$ represents the lower and upper intersections between the curve and the x-axis. There are five elements in `P`, representing the values of y_{opt}, x_{opt}, $x_{\min}$, $x_{\max}$, and δ, respectively.

When `simpver = 1`, the simplified version 1 is selected:

$$\text{if } x \in \left(0, \frac{x_{\max}}{2}\right),$$

$$y = y_{\text{opt}} \left\{ \frac{x\left(x - x_{\max}\right)^2}{\left(x_{\max} - x_{\text{opt}}\right)\left[(x_{\max} - x_{\text{opt}})(x - x_{\text{opt}}) + x_{\text{opt}}(x_{\text{opt}} + x_{\max} - 2x)\right]} \right\}^{\delta};$$

$$\text{if } x \in \left[\frac{x_{\max}}{2}, x_{\max}\right),$$

$$y = y_{\text{opt}} \left\{ \frac{\left(x - x_{\max}\right)x^2}{x_{\text{opt}}\left[x_{\text{opt}}(x - x_{\text{opt}}) - (x_{\text{opt}} - x_{\max})(x_{\text{opt}} - 2x)\right]} \right\}^{\delta};$$

$$\text{if } x \notin (0, x_{\max}),$$

$$y = 0.$$

There are four elements in `P`, representing the values of y_{opt}, x_{opt}, $x_{\max}$, and δ, respectively.

When `simpver = 2`, the simplified version 2 is selected:

$$\text{if } x \in \left(x_{\min}, \frac{x_{\min} + x_{\max}}{2}\right),$$

$$y = \frac{y_{\text{opt}}\left(x - x_{\min}\right)\left(x - x_{\max}\right)^2}{\left(x_{\max} - x_{\text{opt}}\right)\left[(x_{\max} - x_{\text{opt}})(x - x_{\text{opt}}) - (x_{\min} - x_{\text{opt}})(x_{\text{opt}} + x_{\max} - 2x)\right]};$$

$$\text{if } x \in \left[\frac{x_{\min} + x_{\max}}{2}, x_{\max}\right),$$

$$y = \frac{y_{\text{opt}}\left(x - x_{\max}\right)\left(x - x_{\min}\right)^2}{\left(x_{\text{opt}} - x_{\min}\right)\left[(x_{\text{opt}} - x_{\min})(x - x_{\text{opt}}) - (x_{\text{opt}} - x_{\max})(x_{\text{opt}} + x_{\min} - 2x)\right]};$$

$$\text{if } x \notin (x_{\min}, x_{\max}),$$

$$y = 0.$$

There are four elements in `P`, representing the values of y_{opt}, x_{opt}, $x_{\min}$, and $x_{\max}$, respectively.

When `simpver = 3`, the simplified version 3 is selected:

$$\text{if } x \in \left(0, \frac{x_{\max}}{2}\right),$$

$$y = \frac{y_{\text{opt}}x\left(x - x_{\max}\right)^2}{\left(x_{\max} - x_{\text{opt}}\right)\left[(x_{\max} - x_{\text{opt}})(x - x_{\text{opt}}) + x_{\text{opt}}(x_{\text{opt}} + x_{\max} - 2x)\right]};$$

$$\text{if } x \in \left[\frac{x_{\max}}{2}, x_{\max}\right),$$

$$y = \frac{y_{\text{opt}}\left(x - x_{\max}\right)x^2}{x_{\text{opt}}\left[x_{\text{opt}}(x - x_{\text{opt}}) - (x_{\text{opt}} - x_{\max})(x_{\text{opt}} - 2x)\right]};$$

$$\text{if } x \notin (0, x_{\max}),$$

$$y = 0.$$

There are three elements in `P`, representing the values of y_{opt}, x_{opt}, and $x_{\max}$, respectively.

Value

The y values predicted by the modified LRF equation or one of its simplified versions.

Note

We have added n parameter δ in the original LRF equation (i.e., `simpver = 2`) to increase the flexibility for data fitting.

Author(s)

Peijian Shi `<pjshi@njfu.edu.cn>`, Johan Gielis `<johan.gielis@uantwerpen.be>`, Brady K. Quinn `<Brady.Quinn@dfo-mpo.gc.ca>`.

References

Shi, P., Fan, M., Ratkowsky, D.A., Huang, J., Wu, H., Chen, L., Fang, S., Zhang, C. (2017) Comparison of two ontogenetic growth equations for animals and plants. *Ecological Modelling* 349, 1−10. doi:10.1016/j.ecolmodel.2017.01.012

Shi, P., Gielis, J., Quinn, B.K., Niklas, K.J., Ratkowsky, D.A., Schrader, J., Ruan, H., Wang, L., Niinemets, Ü. (2022) 'biogeom': An R package for simulating and fitting natural shapes. *Annals of the New York Academy of Sciences* 1516, 123−134. doi:10.1111/nyas.14862

See Also

`areaovate`, `curveovate`, `fitovate`, `MbetaE`, `MBriereE`, `sigmoid`

Examples

```
x3   <- seq(-5, 15, len=2000)
Par3 <- c(3, 3, 10, 2)
y3   <- MbetaE(P=Par3, x=x3, simpver=1)

dev.new()
plot( x3, y3,cex.lab=1.5, cex.axis=1.5, type="l",
      xlab=expression(italic(x)), ylab=expression(italic(y)) )

graphics.off()
```

`Neocinnamomum` *Leaf Boundary Data of Seven Species of* Neocinnamomum

Description

The data consist of the leaf boundary data of seven species of *Neocinnamomum*.

Usage

```
data(Neocinnamomum)
```

Details

In the data set, there are four columns of variables: `Code`, `LatinName`, `x`, and `y`. `Code` saves the codes of individual leaves; `LatinName` saves the Latin names of the seven species of *Neocinnamomum*; `x` saves the x coordinates of the leaf boundary in the Cartesian coordinate system (cm); and `y` saves the y coordinates of the leaf boundary in the Cartesian coordinate system (cm).

References

Shi, P., Fan, M., Ratkowsky, D.A., Huang, J., Wu, H., Chen, L., Fang, S., Zhang, C. (2017) Comparison of two ontogenetic growth equations for animals and plants. *Ecological Modelling* 349, 1−10. doi:10.1016/j.ecolmodel.2017.01.012

Shi, P., Yu, K., Niklas, K.J., Schrader, J., Song, Y., Zhu, R., Li, Y., Wei, H., Ratkowsky, D.A. (2021) A general model for describing the ovate leaf shape. *Symmetry*, 13, 1524. doi:10.3390/sym13081524

Examples

```
data(Neocinnamomum)

uni.C <- sort( unique(Neocinnamomum$Code) )
ind   <- 2
Data  <- Neocinnamomum[Neocinnamomum$Code==uni.C[ind], ]
x0    <- Data$x
y0    <- Data$y
length(x0)

Res1  <- adjdata(x0, y0, ub.np=200, len.pro=1/20)
x1    <- Res1$x
y1    <- Res1$y
length(x1)

dev.new()
plot( Res1$x, Res1$y, asp=1, cex.lab=1.5, cex.axis=1.5,
      xlab=expression(italic("x")), ylab=expression(italic("y")) )

graphics.off()
```

NRGE *The Narushin-Romanov-Griffin Equation (NRGE)*

Description

`NRGE` is used to calculate y values at given x values using the Narushin-Romanov-Griffin equation (NRGE).

Usage

```
NRGE(P, x)
```

Arguments

`P`	the four parameters (i.e., A, B, C, and D) of the Narushin-Romanov-Griffin equation.
`x`	the given x values.

Details

The Narushin-Romanov-Griffin equation (Narushin et al., 2021) has four parameters in total, among which three parameters have clear geometric meanings.

$$f_1(x) = \frac{B}{2}\sqrt{\frac{A^2 - 4x^2}{A^2 + 8Cx + 4C^2}},$$

$$E = \frac{\sqrt{5.5A^2 + 11AC + 4C^2} \cdot \left(\sqrt{3}AB - 2D\sqrt{A^2 + 2AC + 4C^2}\right)}{\sqrt{3}AB\left(\sqrt{5.5A^2 + 11AC + 4C^2} - 2\sqrt{A^2 + 2AC + 4C^2}\right)},$$

$$f_2(x) = \sqrt{\frac{A\left(A^2 + 8Cx + 4C^2\right)}{2(A - 2C)x^2 + \left(A^2 + 8AC - 4C^2\right)x + 2AC^2 + A^2C + A^3}},$$

$$f(x) = \pm f_1(x) \cdot \left\{1 - E \cdot \left[1 - f_2(x)\right]\right\}.$$

Here, $f(x)$ is the Narushin-Romanov-Griffin equation, which is used to predict the y coordinates at the given x coordinates; A represents the egg's length; B represents the egg's maximum breadth; C is a parameter to be estimated, and it can be expressed as $(A - B)/(2q)$, where q is a parameter to be estimated; D represents the egg's breadth associated with $(3/4)L$ from the egg base (to the egg tip) on the egg length axis (which can be regarded as the major axis of the egg shape).

Value

The y values predicted by the Narushin-Romanov-Griffin equation.

Note

Here, parameter `C` is a parameter to be estimated, which can be directly calculated numerically based on the egg-shape data.

Author(s)

Peijian Shi <`pjshi@njfu.edu.cn`>, Johan Gielis <`johan.gielis@uantwerpen.be`>, Brady K. Quinn <`Brady.Quinn@dfo-mpo.gc.ca`>.

References

Narushin, V.G., Romanov, M.N., Griffin, D.K. (2021) Egg and math: introducing a universal formula for egg shape. *Annals of the New York Academy of Sciences* 1505, 169−177. doi:10.1111/nyas.14680

Shi, P., Gielis, J., Niklas, K.J. (2022) Comparison of a universal (but complex) model for avian egg shape with a simpler model. *Annals of the New York Academy of Sciences* 1514, 34−42. doi:10.1111/nyas.14799

Tian, F., Wang, Y., Sandhu, H.S., Gielis, J., Shi, P. (2020) Comparison of seed morphology of two ginkgo cultivars. *Journal of Forestry Research* 31, 751−758. doi:10.1007/s116760180770y

See Also

`curveNRGE`, `fitNRGE`, `SurfaceAreaNRGE`, `VolumeNRGE`

Examples

```
P0 <- c(11.5, 7.8, 1.1, 5.6)
x  <- seq(-11.5/2, 11.5/2, len=2000)
y1 <- NRGE(P=P0, x=x)
y2 <- -NRGE(P=P0, x=x)

dev.new()
plot(x, y1, cex.lab=1.5, cex.axis=1.5, type="l",
     col=4, ylim=c(-4, 4), asp=1,
     xlab=expression(italic(x)), ylab=expression(italic(y)) )
lines(x, y2, col=2)

graphics.off()
```

PE *Calculation of the Abscissa, Ordinate and Distance From the Origin For an Arbitrary Point on the Preston Curve*

Description

`PE` is used to calculate the abscissa, ordinate and distance from the origin for an arbitrary point on the Preston curve that was generated by the original Preston equation or one of its simplified versions at a given angle.

Usage

```
PE(P, zeta, simpver = NULL)
```

Arguments

`P`	the parameters of the original Preston equation or one of its simplified versions.
`zeta`	the angle(s) used in the Preston equation.
`simpver`	an optional argument to use the simplified version of the original Preston equation.

Details

When `simpver = NULL`, the original Preston equation is selected:

$$y = a\,\sin\zeta,$$

$$x = b\,\cos\zeta\,\left(1 + c_1\sin\zeta + c_2\sin^2\zeta + c_3\sin^3\zeta\right),$$

$$r = \sqrt{x^2 + y^2},$$

where x and y represent the abscissa and ordinate of an arbitrary point on the Preston curve corresponding to an angle ζ; r represents the distance of the point from the origin; a, b, c_1, c_2, and c_3 are parameters to be estimated.

When `simpver = 1`, the simplified version 1 is selected:

$$y = a \sin \zeta,$$
$$x = b \cos \zeta \left(1 + c_1 \sin \zeta + c_2 \sin^2 \zeta\right),$$
$$r = \sqrt{x^2 + y^2},$$

where x and y represent the abscissa and ordinate of an arbitrary point on the Preston curve corresponding to an angle ζ; r represents the distance of the point from the origin; a, b, c_1, and c_2 are parameters to be estimated.

When `simpver = 2`, the simplified version 2 is selected:

$$y = a \sin \zeta,$$
$$x = b \cos \zeta \left(1 + c_1 \sin \zeta\right),$$
$$r = \sqrt{x^2 + y^2},$$

where x and y represent the abscissa and ordinate of an arbitrary point on the Preston curve corresponding to an angle ζ; r represents the distance of the point from the origin; a, b, and c_1 are parameters to be estimated.

When `simpver = 3`, the simplified version 3 is selected:

$$y = a \sin \zeta,$$
$$x = b \cos \zeta \left(1 + c_2 \sin^2 \zeta\right),$$
$$r = \sqrt{x^2 + y^2},$$

where x and y represent the abscissa and ordinate of an arbitrary point on the Preston curve corresponding to an angle ζ; r represents the distance of the point from the origin; a, b, and c_2 are parameters to be estimated.

Value

`x`	the abscissa(s) of the Preston curve corresponding to the given angle(s).
`y`	the ordinate(s) of the Preston curve corresponding to the given angle(s).
`r`	the distance(s) of the Preston curve corresponding to the given angle(s) from the origin.

Note

ζ is NOT the polar angle corresponding to r, i.e.,

$$y \neq r \sin \zeta,$$
$$x \neq r \cos \zeta.$$

Let φ be the polar angle corresponding to r. We have:

$$\zeta = \arcsin \frac{r \sin \varphi}{a}.$$

Author(s)

Peijian Shi <`pjshi@njfu.edu.cn`>, Johan Gielis <`johan.gielis@uantwerpen.be`>, Brady K. Quinn <`Brady.Quinn@dfo-mpo.gc.ca`>.

References

Biggins, J.D., Montgomeries, R.M., Thompson, J.E., Birkhead, T.R. (2022) Preston's universal formula for avian egg shape. *Ornithology* In press. doi:10.1093/ornithology/ukac028

Biggins, J.D., Thompson, J.E., Birkhead, T.R. (2018) Accurately quantifying the shape of birds' eggs. *Ecology and Evolution* 8, 9728−9738. doi:10.1002/ece3.4412

Preston, F.W. (1953) The shapes of birds' eggs. *The Auk* 70, 160−182.

Shi, P., Gielis, J., Quinn, B.K., Niklas, K.J., Ratkowsky, D.A., Schrader, J., Ruan, H., Wang, L., Niinemets, Ü. (2022) 'biogeom': An R package for simulating and fitting natural shapes. *Annals of the New York Academy of Sciences* 1516, 123−134. doi:10.1111/nyas.14862

Todd, P.H., Smart, I.H.M. (1984) The shape of birds' eggs. *Journal of Theoretical Biology* 106, 239−243. doi:10.1016/00225193(84)900213

See Also

`EPE`, `lmPE`, `TSE`

Examples

```
zeta <- seq(0, 2*pi, len=2000)
Par1 <- c(10, 6, 0.325, -0.0415)
Res1 <- PE(P=Par1, zeta=zeta, simpver=1)
Par2 <- c(10, 6, -0.325, -0.0415)
Res2 <- PE(P=Par2, zeta=zeta, simpver=1)

dev.new()
plot(Res1$x, Res1$y, asp=1, type="l", col=4, cex.lab=1.5, cex.axis=1.5,
     xlab=expression(italic(x)), ylab=expression(italic(y)))
lines(Res2$x, Res2$y, col=2)

dev.new()
plot(Res1$r, Res2$r, asp=1, cex.lab=1.5, cex.axis=1.5,
     xlab=expression(paste(italic(r), ""[1], sep="")),
     ylab=expression(paste(italic(r), ""[2], sep="")))
abline(0, 1, col=4)

graphics.off()
```

`shoots` *Height Growth Data of Bamboo Shoots*

Description

The height data of four species of bamboo at Nanjing Forestry University campus in 2016.

Usage

```
data(shoots)
```

Details

In the data set, there are four columns of variables: `Code`, `LatinName`, `x`, and `y`. `Code` saves the number codes of different bamboo species; `LatinName` saves the Latin names of different bamboo species; `x` saves the investigation times (days from a specific starting time of growth, and where every bamboo has a different starting time of growth); and `y` saves the measured aboveground height values (cm).

`Code = 1` represents *Phyllostachys iridescens*, and the starting time (namely `time = 0`) was defined as 12:00, 3rd April, 2016;

`Code = 2` represents *Phyllostachys mannii*, and the starting time (namely `time = 0`) was defined as 12:00, 4th April, 2016;

`Code = 3` represents *Pleioblastus maculatus*, and the starting time (namely `time = 0`) was defined as 12:00, 29th April, 2016;

`Code = 4` represents *Sinobambusa tootsik*, and the starting time (namely `time = 0`) was defined as 12:00, 18th April, 2016.

References

Shi, P., Fan, M., Ratkowsky, D.A., Huang, J., Wu, H., Chen, L., Fang, S., Zhang, C. (2017) Comparison of two ontogenetic growth equations for animals and plants. *Ecological Modelling* 349, 1−10. doi:10.1016/j.ecolmodel.2017.01.012

Examples

```
data(shoots)
attach(shoots)
# Choose a species
# 1: Phyllostachys iridescens; 2: Phyllostachys mannii;
# 3: Pleioblastus maculatus; 4: Sinobambusa tootsik
ind <- 4
x1  <- x[Code == ind]
y1  <- y[Code == ind]

dev.new()
plot(x1, y1, cex=1.5, cex.lab=1.5, cex.axis=1.5, xlab="Time (days)", ylab="Height (cm)")
```

```
  delta0  <- c(0.5, 1, 2, 5, 10, 20)
  ini.val <- list(600, 25, 0, 40, delta0)
  resu1   <- fitsigmoid(MLRFE, x=x1, y=y1, ini.val=ini.val, simpver=NULL,
                fig.opt=TRUE, control=list(reltol=1e-20, maxit=20000),
                subdivisions = 100L, rel.tol=.Machine$double.eps^0.25,
                abs.tol=.Machine$double.eps^0.25, stop.on.error=TRUE,
                keep.xy=FALSE, aux=NULL)


graphics.off()
```

`sigmoid` *Sigmoid Growth Equation*

Description

`sigmoid` is used to calculate the y values (e.g., biomass, height, body length, and so on) at given investigation times.

Usage

```
sigmoid(expr, P, x, simpver = 1, subdivisions = 100L,
        rel.tol = .Machine$double.eps^0.25,
        abs.tol = rel.tol, stop.on.error = TRUE,
        keep.xy = FALSE, aux = NULL)
```

Arguments

`expr`	a performance equation or one of its simplified versions.
`P`	the parameters of the performance equation or one of its simplified versions.
`x`	the given investigation times.
`simpver`	an optional argument to use the simplfied version of the performance equation.
`subdivisions`	please see the arguments for the `integrate` function in package **stats**.
`rel.tol`	please see the arguments for the `integrate` function in package **stats**.
`abs.tol`	please see the arguments for the `integrate` function in package **stats**.
`stop.on.error`	please see the arguments for the `integrate` function in package **stats**.
`keep.xy`	please see the arguments for the `integrate` function in package **stats**.
`aux`	please see the arguments for the `integrate` function in package **stats**.

Details

The performance equations denote `MbetaE`, `MBriereE`, `MLRFE`, and their simplified versions. The arguments of `P` and `simpver` should correspond to `expr` (i.e., `MbetaE` or `MBriereE` or `MLRFE`). The sigmoid curve is the integral of the performance equation or one of its simplified versions.

Value

The y values (i.e., biomass, height, body length, and so on) at given investigation times. The growth euqation is actually an integral of the performance equation or one of its simplified versions.

Note

Here, the user can define other performance equations, but new equations or their simplified versions should include the lower and upper thresholds in the x-axis corresponding to $y = 0$, whose indices of the parameters in `P` should be the same as those in `Mbeta` or `MBriere` or `MLRF`.

Author(s)

Peijian Shi <`pjshi@njfu.edu.cn`>, Johan Gielis <`johan.gielis@uantwerpen.be`>, Brady K. Quinn <`Brady.Quinn@dfo-mpo.gc.ca`>.

References

Jin, J., Quinn, B.K., Shi, P. (2022) The modified Brière equation and its applications. *Plants* 11, 1769. doi:10.3390/plants11131769

Shi, P., Fan, M., Ratkowsky, D.A., Huang, J., Wu, H., Chen, L., Fang, S., Zhang, C. (2017) Comparison of two ontogenetic growth equations for animals and plants. *Ecological Modelling* 349, 1−10. doi:10.1016/j.ecolmodel.2017.01.012

Shi, P., Gielis, J., Quinn, B.K., Niklas, K.J., Ratkowsky, D.A., Schrader, J., Ruan, H., Wang, L., Niinemets, Ü. (2022) 'biogeom': An R package for simulating and fitting natural shapes. *Annals of the New York Academy of Sciences* 1516, 123−134. doi:10.1111/nyas.14862

See Also

`fitsigmoid`, `MbetaE`, `MBriereE`, `MLRFE`

Examples

```
Pa1 <- c(3, 3, 10, 1)
xv1 <- seq(-5, 15, len=2000)
yv1 <- sigmoid(MBriereE, P=Pa1, x=xv1, simpver=1)
Pa2 <- c(3, 3, 2, 12, 1)
yv2 <- sigmoid(MBriereE, P=Pa2, x=xv1, simpver=NULL)

dev.new()
plot( xv1, yv2, cex.lab=1.5, cex.axis=1.5, type="l", col=4,
      xlab=expression(italic(x)), ylab=expression(italic(y)) )
lines( xv1, yv1, col=2 )

graphics.off()
```

`starfish` *Boundary Data of Eight Sea Stars*

Description

The data consist of the boundary data of eight sea stars from five species.

Usage

```
data(starfish)
```

Details

In the data set, there are four columns of variables: `Code`, `LatinName`, `x`, and `y`. `Code` saves the codes of individual sea stars; `LatinName` saves the Latin names of the eight sea stars; `x` saves the x coordinates of the eight sea stars in the Cartesian coordinate system (cm); and `y` saves the y coordinates of the eight sea stars in the Cartesian coordinate system (cm). In `Code`, codes 1-9 represent *Anthenoides tenuis*, *Culcita schmideliana* sample 1, *Culcita schmideliana* sample 2, *Culcita schmideliana* sample 3, *Stellaster equestris*, *Tosia australis*, *Tosia magnifica* sample 1, and *Tosia magnifica* sample 2, respectively. See Table A1 published in Shi et al. (2020).

References

Shi, P., Ratkowsky, D.A., Gielis, J. (2020) The generalized Gielis geometric equation and its application. *Symmetry* 12, 645. doi:10.3390/sym12040645

Examples

```
data(starfish)

uni.C <- sort( unique(starfish$Code) )
ind   <- 2
Data  <- starfish[starfish$Code==uni.C[ind], ]
x0    <- Data$x
y0    <- Data$y

dev.new()
plot( x0, y0, asp=1, type="l", cex.lab=1.5, cex.axis=1.5,
      xlab=expression(italic("x")), ylab=expression(italic("y")) )
length(x0)

Res1 <- adjdata(x0, y0, ub.np=400, times=1.2, len.pro=1/20)
x1   <- Res1$x
y1   <- Res1$y

dev.new()
plot( x1, y1, asp=1, type="l", cex.lab=1.5, cex.axis=1.5,
      xlab=expression(italic("x")), ylab=expression(italic("y")) )
length(x1)
```

```
graphics.off()
```

`SurfaceAreaEPE` *Calculation of the Surface Area of An Egg Based on the Explicit Preston Equation*

Description

`SurfaceAreaEPE` is used to calculate the surface area of an egg that follows the explicit Preston equation.

Usage

```
SurfaceAreaEPE(P, simpver = NULL, subdivisions = 100L,
          rel.tol = .Machine$double.eps^0.25, abs.tol = rel.tol,
          stop.on.error = TRUE, keep.xy = FALSE, aux = NULL)
```

Arguments

`P`	the parameters of the explicit Preston equation or one of its simplified versions.
`simpver`	an optional argument to use the simplified version of the explicit Preston equation.
`subdivisions`	please see the arguments for the `integrate` function in package **stats**.
`rel.tol`	please see the arguments for the `integrate` function in package **stats**.
`abs.tol`	please see the arguments for the `integrate` function in package **stats**.
`stop.on.error`	please see the arguments for the `integrate` function in package **stats**.
`keep.xy`	please see the arguments for the `integrate` function in package **stats**.
`aux`	please see the arguments for the `integrate` function in package **stats**.

Details

The formula of the surface area (S) of an egg based on the explicit Preston equation or one of its simplified versions is:

$$S(x) = 2\,\pi \int_{-a}^{a} y\,\sqrt{1+\left(\frac{dy}{dx}\right)^{2}}\,dx,$$

where y denotes the explicit Preston equation (i.e., `EPE`), and a denotes half the egg's length. When `simpver = NULL`, `P` has five parameters: a, b, c_1, c_2, and c_3; when `simpver = 1`, `P` has four parameters: a, b, c_1, and c_2; when `simpver = 2`, `P` has three parameters: a, b, and c_1; when `simpver = 3`, `P` has three parameters: a, b, and c_2.

Note

The argument `P` in the `SurfaceAreaEPE` function has the same parameters, as those in the `EPE` function.

Author(s)

Peijian Shi <`pjshi@njfu.edu.cn`>, Johan Gielis <`johan.gielis@uantwerpen.be`>, Brady K. Quinn <`Brady.Quinn@dfo-mpo.gc.ca`>.

References

Narushin, V.G., Romanov, M.N., Mishra, B., Griffin, D.K. (2022) Mathematical progression of avian egg shape with associated area and volume determinations. *Annals of the New York Academy of Sciences* 1513, 65–78. doi:10.1111/nyas.14771

Shi, P., Chen, L., Quinn, B.K., Yu, K., Miao, Q., Guo, X., Lian, M., Gielis, J., Niklas, K.J. (2022) Can we simply calculate the volume and surface area of an egg? Evidence from six species of poultry. *Annals of the New York Academy of Sciences* Under preparation.

See Also

`DEPE`, `EPE`, `fitEPE`, `VolumeEPE`

Examples

```
Par4 <- c(4.27, 2.90, 0.0868, 0.0224, -0.0287)
SurfaceAreaEPE(P = Par4, simpver = NULL)
```

`SurfaceAreaETE` *Calculation of the Surface Area of An Egg Based on the Explicit Troscianko Equation*

Description

`SurfaceAreaETE` is used to calculate the surface area of an egg that follows the explicit Troscianko equation.

Usage

```
SurfaceAreaETE(P, subdivisions = 100L,
          rel.tol = .Machine$double.eps^0.25, abs.tol = rel.tol,
          stop.on.error = TRUE, keep.xy = FALSE, aux = NULL)
```

Arguments

`P`	the parameters of the explicit Troscianko equation.
`subdivisions`	please see the arguments for the `integrate` function in package **stats**.
`rel.tol`	please see the arguments for the `integrate` function in package **stats**.
`abs.tol`	please see the arguments for the `integrate` function in package **stats**.
`stop.on.error`	please see the arguments for the `integrate` function in package **stats**.
`keep.xy`	please see the arguments for the `integrate` function in package **stats**.
`aux`	please see the arguments for the `integrate` function in package **stats**.

Details

The formula of the surface area (S) of an egg based on the explicit Troscianko equation is:

$$S(x) = 2\,\pi \int_{-a}^{a} y \sqrt{1 + \left(\frac{dy}{dx}\right)^2}\, dx,$$

where y denotes the explicit Troscianko equation (i.e., `ETE`), and a denotes half the egg's length.

Note

The argument `P` in the `SurfaceAreaETE` function has the same parameters, as those in the `ETE` function.

Author(s)

Peijian Shi <`pjshi@njfu.edu.cn`>, Johan Gielis <`johan.gielis@uantwerpen.be`>, Brady K. Quinn <`Brady.Quinn@dfo-mpo.gc.ca`>.

References

Narushin, V.G., Romanov, M.N., Mishra, B., Griffin, D.K. (2022) Mathematical progression of avian egg shape with associated area and volume determinations. *Annals of the New York Academy of Sciences* 1513, 65−78. doi:10.1111/nyas.14771

Shi, P., Chen, L., Quinn, B.K., Yu, K., Miao, Q., Guo, X., Lian, M., Gielis, J., Niklas, K.J. (2022) Can we simply calculate the volume and surface area of an egg? Evidence from six species of poultry. *Annals of the New York Academy of Sciences* Under preparation.

See Also

`DETE`, `ETE`, `fitETE`, `VolumeETE`

Examples

```
Par5 <- c(2.25, -0.38, -0.29, -0.16)
SurfaceAreaETE(P = Par5)
```

`SurfaceAreaNRGE` *Calculation of the Surface Area of An Egg Based on the Narushin-Romanov-Griffin Equation*

Description

`SurfaceAreaNRGE` is used to calculate the surface area of an egg that follows the Narushin-Romanov-Griffin equation.

Usage

```
SurfaceAreaNRGE(P, subdivisions = 100L,
          rel.tol = .Machine$double.eps^0.25, abs.tol = rel.tol,
          stop.on.error = TRUE, keep.xy = FALSE, aux = NULL)
```

Arguments

`P`	the parameters of the Narushin-Romanov-Griffin equation.
`subdivisions`	please see the arguments for the `integrate` function in package **stats**.
`rel.tol`	please see the arguments for the `integrate` function in package **stats**.
`abs.tol`	please see the arguments for the `integrate` function in package **stats**.
`stop.on.error`	please see the arguments for the `integrate` function in package **stats**.
`keep.xy`	please see the arguments for the `integrate` function in package **stats**.
`aux`	please see the arguments for the `integrate` function in package **stats**.

Details

The formula of the surface area (S) of an egg based on the Narushin-Romanov-Griffin equation is:

$$S(x) = 2\,\pi \int_{-A/2}^{A/2} y\,\sqrt{1+\left(\frac{dy}{dx}\right)^2}\,dx,$$

where y denotes the Narushin-Romanov-Griffin equation (i.e., `NRGE`), and A denotes the egg's length, which is the first element in the parameter vector, `P`.

Note

The argument `P` in the `SurfaceAreaNRGE` function has the same parameters, as those in the `NRGE` function.

Author(s)

Peijian Shi <`pjshi@njfu.edu.cn`>, Johan Gielis <`johan.gielis@uantwerpen.be`>, Brady K. Quinn <`Brady.Quinn@dfo-mpo.gc.ca`>.

References

Narushin, V.G., Romanov, M.N., Griffin, D.K. (2021) Egg and math: introducing a universal formula for egg shape. *Annals of the New York Academy of Sciences* 1505, 169−177. doi:10.1111/nyas.14680

Narushin, V.G., Romanov, M.N., Mishra, B., Griffin, D.K. (2022) Mathematical progression of avian egg shape with associated area and volume determinations. *Annals of the New York Academy of Sciences* 1513, 65−78. doi:10.1111/nyas.14771

Shi, P., Chen, L., Quinn, B.K., Yu, K., Miao, Q., Guo, X., Lian, M., Gielis, J., Niklas, K.J. (2022) Can we simply calculate the volume and surface area of an egg? Evidence from six species of poultry. *Annals of the New York Academy of Sciences* Under preparation.

See Also

`curveNRGE`, `DNRGE`, `fitNRGE`, `NRGE`, `VolumeNRGE`

Examples

```
Par6 <- c(4.51, 3.18, 0.1227, 2.2284)
SurfaceAreaNRGE(P = Par6)
```

`SurfaceAreaSGE` *Calculation of the Surface Area of An Egg Based on the Simplified Gielis Equation*

Description

`SurfaceAreaSGE` is used to calculate the surface area of an egg that follows the simplified Gielis equation.

Usage

```
SurfaceAreaSGE(P, subdivisions = 100L,
          rel.tol = .Machine$double.eps^0.25, abs.tol = rel.tol,
          stop.on.error = TRUE, keep.xy = FALSE, aux = NULL)
```

Arguments

`P`	the parameters of the simplified Gielis equation, including a, n_1, and n_2.
`subdivisions`	please see the arguments for the `integrate` function in package **stats**.
`rel.tol`	please see the arguments for the `integrate` function in package **stats**.
`abs.tol`	please see the arguments for the `integrate` function in package **stats**.
`stop.on.error`	please see the arguments for the `integrate` function in package **stats**.
`keep.xy`	please see the arguments for the `integrate` function in package **stats**.
`aux`	please see the arguments for the `integrate` function in package **stats**.

Details

The formula of the surface area (S) of an egg based on the simplified Gielis equation is:

$$S(\varphi) = 2\,\pi \int_{0}^{\pi} \sin\left(\varphi\right) r \sqrt{r^2 + \left(\frac{dr}{d\varphi}\right)^2}\, d\varphi,$$

where the polar raidus (r) is the function of the polar angle (φ):

$$r\left(\varphi\right) = a\left(\left|\cos\left(\frac{m}{4}\varphi\right)\right|^{n_2} + \left|\sin\left(\frac{m}{4}\varphi\right)\right|^{n_2}\right)^{-\frac{1}{n_1}},$$

namely the simplified Gielis equation (i.e., `GE`) with arguments `simpver = 1` and `m = 1`.

Note

The argument `P` in the `SurfaceAreaSGE` function only has the three parameters: a, n_1, and n_2.

Author(s)

Peijian Shi <`pjshi@njfu.edu.cn`>, Johan Gielis <`johan.gielis@uantwerpen.be`>, Brady K. Quinn <`Brady.Quinn@dfo-mpo.gc.ca`>.

References

Chen, Z. (2012) Volume and area of revolution under polar coordinate system. *Studies in College Mathematics* 15(6), 9–11.

Shi, P., Chen, L., Quinn, B.K., Yu, K., Miao, Q., Guo, X., Lian, M., Gielis, J., Niklas, K.J. (2022) Can we simply calculate the volume and surface area of an egg? Evidence from six species of poultry. *Annals of the New York Academy of Sciences* Under preparation.

See Also

`DSGE`, `fitGE`, `GE`, `VolumeSGE`

Examples

```
Par7 <- c(1.124, 14.86, 49.43)
SurfaceAreaSGE(P = Par7)
```

`TE` *The Troscianko Equation (TE)*

Description

`TE` is used to calculate y values at given x values using the re-expression of Troscianko's egg-shape equation, which was proposed by Biggins et al. (2018, 2022).

Usage

```
TE(P, x)
```

Arguments

`P`	the parameters of the Troscianko equation, including α_0, α_1, and α_2.
`x`	the given x values ranging from -1 to 1.

Details

The Troscianko equation is recommended as (Biggins et al., 2022):

$$y = \exp\left(\alpha_0 + \alpha_1\, x + \alpha_2\, x^2\right)\sqrt{1-x^2},$$

where x and y represent the abscissa and ordinate of an arbitrary point on the Troscianko curve; α_0, α_1, and α_2 are parameters to be estimated.

Value

The y values predicted by the Troscianko equation.

Note

Here, x and y in the Troscianko equation are actually equal to y/a and x/a, respectively, in the explicit Troscianko equation, where a represents half the egg length (See `ETE` for details). This means that the egg length is scaled to be 2, and the maximum egg width is correspondingly adjusted to keep the same scale.

Author(s)

Peijian Shi <`pjshi@njfu.edu.cn`>, Johan Gielis <`johan.gielis@uantwerpen.be`>, Brady K. Quinn <`Brady.Quinn@dfo-mpo.gc.ca`>.

References

Biggins, J.D., Montgomeries, R.M., Thompson, J.E., Birkhead, T.R. (2022) Preston's universal formula for avian egg shape. *Ornithology* In press. doi:10.1093/ornithology/ukac028

Biggins, J.D., Thompson, J.E., Birkhead, T.R. (2018) Accurately quantifying the shape of birds' eggs. *Ecology and Evolution* 8, 9728−9738. doi:10.1002/ece3.4412

Nelder, J.A., Mead, R. (1965). A simplex method for function minimization. *Computer Journal* 7, 308−313. doi:10.1093/comjnl/7.4.308

Shi, P., Gielis, J., Quinn, B.K., Niklas, K.J., Ratkowsky, D.A., Schrader, J., Ruan, H., Wang, L., Niinemets, Ü. (2022) 'biogeom': An R package for simulating and fitting natural shapes. *Annals of the New York Academy of Sciences* 1516, 123−134. doi:10.1111/nyas.14862

Shi, P., Wang, L., Quinn, B.K., Gielis, J. (2022) A new program to estimate the parameters of Preston's equation, a general formula for describing the egg shape of birds. *Symmetry* Under review.

Troscianko, J. (2014). A simple tool for calculating egg shape, volume and surface area from digital images. *Ibis*, 156, 874−878. doi:10.1111/ibi.12177

See Also

`fitETE`, `lmTE`

Examples

```
Par <- c(-0.377, -0.29, -0.16)
xb1 <- seq(-1, 1, len=20000)
yb1 <- TE(P=Par, x=xb1)
xb2 <- seq(1, -1, len=20000)
yb2 <- -TE(P=Par, x=xb2)

dev.new()
plot(xb1, yb1, asp=1, type="l", col=2, ylim=c(-1, 1), cex.lab=1.5, cex.axis=1.5,
  xlab=expression(italic(x)), ylab=expression(italic(y)))
lines(xb2, yb2, col=4)

graphics.off()
```

`TGE` *Calculation of the Polar Radius of the Twin Gielis Curve*

Description

`TGE` is used to calculate the polar radii of the twin Gielis equation or one of its simplified versions at given polar angles.

Usage

```
TGE(P, phi, m = 1, simpver = NULL, nval = 1)
```

Arguments

`P`	the parameters of the twin Gielis equation or one of its simplified versions.
`phi`	the polar angle(s).
`m`	the given m value that determines the number of angles of the twin Gielis curve within $[0, 2\pi)$.
`simpver`	an optional argument to use the simplified version of the twin Gielis equation.
`nval`	the specified value for n_2 or n_3 in the simplified versions.

Details

The general form of the twin Gielis equation can be represented as follows:

$$r\left(\varphi\right) = \exp\left\{\frac{1}{\alpha + \beta\ln\left[r_e\left(\varphi\right)\right]} + \gamma\right\},$$

where r represents the polar radius of the twin Gielis curve at the polar angle φ, and r_e represents the elementary polar radius at the polar angle φ. There is a hyperbolic link function to link their log-transformations, i.e.,

$$\ln\left[r\left(\varphi\right)\right] = \frac{1}{\alpha + \beta\ln\left[r_e\left(\varphi\right)\right]} + \gamma.$$

The first three elements of `P` are α, β, and γ, and the remaining element(s) of `P` are the parameters of the elementary polar function, i.e., $r_e\left(\varphi\right)$. See Shi et al. (2020) for details.

When `simpver = NULL`, the original twin Gielis equation is selected:

$$r_e\left(\varphi\right) = \left|\cos\left(\frac{m}{4}\varphi\right)\right|^{n_2} + \left|\frac{1}{k}\sin\left(\frac{m}{4}\varphi\right)\right|^{n_3},$$

where r_e represents the elementary polar radius at the polar angle φ; m determines the number of angles of the twin Gielis curve within $[0, 2\pi)$; and k, n_2, and n_3 are the fourth to the sixth elements in `P`. In total, there are six elements in `P`.

When `simpver = 1`, the simplified version 1 is selected:

$$r_e\left(\varphi\right) = \left|\cos\left(\frac{m}{4}\varphi\right)\right|^{n_2} + \left|\sin\left(\frac{m}{4}\varphi\right)\right|^{n_2},$$

where n_2 is the fourth element in `P`. There are four elements in total in `P`.

When `simpver = 2`, the simplified version 2 is selected:

$$r_e\left(\varphi\right) = \left|\cos\left(\frac{m}{4}\varphi\right)\right|^{n_2} + \left|\sin\left(\frac{m}{4}\varphi\right)\right|^{n_2},$$

where n_2 should be specified in `nval`, and `P` only includes three elements, i.e., α, β, and γ.

When `simpver = 3`, the simplified version 3 is selected:

$$r_e\left(\varphi\right) = \left|\cos\left(\frac{m}{4}\varphi\right)\right|^{n_2} + \left|\sin\left(\frac{m}{4}\varphi\right)\right|^{n_3},$$

where n_2 and n_3 are the fourth and fifth elements in `P`. There are five elements in total in `P`.

When `simpver = 4`, the simplified version 4 is selected:

$$r_e\left(\varphi\right) = \left|\cos\left(\frac{m}{4}\varphi\right)\right|^{n_2} + \left|\frac{1}{k}\sin\left(\frac{m}{4}\varphi\right)\right|^{n_2},$$

where k and n_2 are the fourth and fifth elelments in `P`. There are five elements in total in `P`.

When `simpver = 5`, the simplified version 5 is selected:

$$r_e\left(\varphi\right) = \left|\cos\left(\frac{m}{4}\varphi\right)\right|^{n_2} + \left|\frac{1}{k}\sin\left(\frac{m}{4}\varphi\right)\right|^{n_2},$$

where k is the fourth elelment in `P`. There are four elements in total in `P`. n_2 should be specified in `nval`.

Value

The polar radii predicted by the twin Gielis equation or one of its simplified versions.

Note

`simpver` here is different from that in the `GE` function.

Author(s)

Peijian Shi <`pjshi@njfu.edu.cn`>, Johan Gielis <`johan.gielis@uantwerpen.be`>, Brady K. Quinn <`Brady.Quinn@dfo-mpo.gc.ca`>.

References

Li, Y., Quinn, B.K., Gielis, J., Li, Y., Shi, P. (2022) Evidence that supertriangles exist in nature from the vertical projections of *Koelreuteria paniculata* fruit. *Symmetry* 14, 23. doi:10.3390/sym14010023

Shi, P., Gielis, J., Quinn, B.K., Niklas, K.J., Ratkowsky, D.A., Schrader, J., Ruan, H., Wang, L., Niinemets, Ü. (2022) 'biogeom': An R package for simulating and fitting natural shapes. *Annals of the New York Academy of Sciences* 1516, 123−134. doi:10.1111/nyas.14862

Shi, P., Ratkowsky, D.A., Gielis, J. (2020) The generalized Gielis geometric equation and its application. *Symmetry* 12, 645. doi:10.3390/sym12040645

See Also

areaGE, curveGE, fitGE, GE

Examples

```
TGE.par    <- c(2.88, 0.65, 1.16, 139)
varphi.vec <- seq(0, 2*pi, len=2000)
r2.theor   <- TGE(P=TGE.par, phi=varphi.vec, simpver=1, m=5)

dev.new()
plot( varphi.vec, r2.theor, cex.lab=1.5, cex.axis=1.5,
      xlab=expression(italic(varphi)), ylab=expression(italic("r")),
      type="l", col=4 )

starfish4 <- curveGE(TGE, P=c(0, 0, 0, TGE.par), simpver=1, m=5, fig.opt=TRUE)

graphics.off()
```

TSE *The Todd-Smart Equation (TSE)*

Description

`TSE` is used to calculate y values at given x values using the Todd and Smart's re-expression of Preston's universal egg shape.

Usage

```
TSE(P, x, simpver = NULL)
```

Arguments

`P`	the parameters of the original Todd-Smart equation or one of its simplified versions.
`x`	the given x values ranging from -1 to 1.
`simpver`	an optional argument to use the simplified version of the original Todd-Smart equation.

Details

When `simpver = NULL`, the original Preston equation is selected:

$$y = d_0 z_0 + d_1 z_1 + d_2 z_2 + d_3 z_3,$$

where

$$z_0 = \sqrt{1 - x^2},$$

$$z_1 = x\sqrt{1 - x^2},$$

$$z_2 = x^2\sqrt{1-x^2},$$
$$z_3 = x^3\sqrt{1-x^2}.$$

Here, x and y represent the abscissa and ordinate of an arbitrary point on the Todd-Smart curve; d_0, d_1, d_2, and d_3 are parameters to be estimated.

When `simpver = 1`, the simplified version 1 is selected:

$$y = d_0 z_0 + d_1 z_1 + d_2 z_2,$$

where x and y represent the abscissa and ordinate of an arbitrary point on the Todd-Smart curve; d_0, d_1, and d_2 are parameters to be estimated.

When `simpver = 2`, the simplified version 2 is selected:

$$y = d_0 z_0 + d_1 z_1,$$

where x and y represent the abscissa and ordinate of an arbitrary point on the Todd-Smart curve; d_0, and d_1 are parameters to be estimated.

When `simpver = 3`, the simplified version 3 is selected:

$$y = d_0 z_0 + d_2 z_2,$$

where x and y represent the abscissa and ordinate of an arbitrary point on the Todd-Smart curve; d_0, and d_2 are parameters to be estimated.

Value

The y values predicted by the Todd-Smart equation.

Note

Here, x and y in the Todd-Smart equation are actually equal to y/a and x/a, respectively, in the Preston equation (See `PE` for details). Since a represents half the egg length, this means that the egg length is fixed to be 2, and the maximum egg width is correspondingly adjusted to keep the same scale.

Author(s)

Peijian Shi <`pjshi@njfu.edu.cn`>, Johan Gielis <`johan.gielis@uantwerpen.be`>, Brady K. Quinn <`Brady.Quinn@dfo-mpo.gc.ca`>.

References

Biggins, J.D., Montgomeries, R.M., Thompson, J.E., Birkhead, T.R. (2022) Preston's universal formula for avian egg shape. *Ornithology* In press. doi:10.1093/ornithology/ukac028

Biggins, J.D., Thompson, J.E., Birkhead, T.R. (2018) Accurately quantifying the shape of birds' eggs. *Ecology and Evolution* 8, 9728−9738. doi:10.1002/ece3.4412

Nelder, J.A., Mead, R. (1965). A simplex method for function minimization. *Computer Journal* 7, 308−313. doi:10.1093/comjnl/7.4.308

Preston, F.W. (1953) The shapes of birds' eggs. *The Auk* 70, 160−182.

Shi, P., Gielis, J., Quinn, B.K., Niklas, K.J., Ratkowsky, D.A., Schrader, J., Ruan, H., Wang, L., Niinemets, Ü. (2022) 'biogeom': An R package for simulating and fitting natural shapes. *Annals of the New York Academy of Sciences* 1516, 123−134. doi:10.1111/nyas.14862

Todd, P.H., Smart, I.H.M. (1984) The shape of birds' eggs. *Journal of Theoretical Biology* 106, 239−243. doi:10.1016/00225193(84)900213

See Also

`lmPE`, `PE`

Examples

```
Par <- c(0.695320398, -0.210538656, -0.070373518, 0.116839895)
xb1 <- seq(-1, 1, len=20000)
yb1 <- TSE(P=Par, x=xb1)
xb2 <- seq(1, -1, len=20000)
yb2 <- -TSE(P=Par, x=xb2)

dev.new()
plot(xb1, yb1, asp=1, type="l", col=2, ylim=c(-1, 1), cex.lab=1.5, cex.axis=1.5,
  xlab=expression(italic(x)), ylab=expression(italic(y)))
lines(xb2, yb2, col=4)

graphics.off()
```

`veins` *Leaf Vein Data of* Michelia compressa

Description

The data consist of the leaf vein data of a leaf of *M. compressa* sampled at Nanjing Forestry University campus in late July 2019.

Usage

```
data(veins)
```

Details

In the data set, there are two columns of variables: `x` and `y`. `x` saves the x coordinates of the leaf veins in the Cartesian coordinate system (cm); `y` saves the y coordinates of the leaf veins in the Cartesian coordinate system (cm).

Note

The data cannot be adjusted by the `adjdata` function.

References

Shi, P., Yu, K., Niinemets, Ü., Gielis, J. (2021) Can leaf shape be represented by the ratio of leaf width to length? Evidence from nine species of *Magnolia* and *Michelia* (Magnoliaceae). *Forests* 12, 41. doi:10.3390/f12010041

See Also

`fracdim`

Examples

```
data(veins)

dev.new()
plot(veins$x, veins$y, cex=0.01, asp=1, cex.lab=1.5, cex.axis=1.5,
     xlab=expression(italic("x")), ylab=expression(italic("y")))

graphics.off()
```

`VolumeEPE` *Calculation of the Volume of An Egg Based on the Explicit Preston Equation*

Description

`VolumeEPE` is used to calculate the volume of an egg that follows the explicit Preston equation.

Usage

```
VolumeEPE(P, simpver = NULL)
```

Arguments

`P`	the parameters of the explicit Preston equation or one of its simplified versions.
`simpver`	an optional argument to use the simplified version of the explicit Preston equation.

Details

When `simpver = NULL`, the volume formula (V) of the explicit Preston equation is selected:

$$V(x) = \frac{4\,\pi}{315} a\, b^2 \left(105 + 21\, c_1^2 + 42\, c_2 + 9\, c_2^2 + 18\, c_1\, c_3 + 5\, c_3^2\right),$$

where `P` has five parameters: a, b, c_1, c_2, and c_3.

When `simpver = 1`, the volume formula of the simplified version 1 is selected:

$$V(x) = \frac{4\,\pi}{315} a\, b^2 \left(105 + 21\, c_1^2 + 42\, c_2 + 9\, c_2^2\right),$$

where `P` has four parameters: a, b, c_1, and c_2.

When `simpver = 2`, the volume formula of the simplified version 2 is selected:

$$V(x) = \frac{4\,\pi}{315} a\, b^2 \left(105 + 21\, c_1^2\right),$$

where `P` has three parameters: a, b, and c_1.

When `simpver = 3`, the volume formula of the simplified version 3 is selected:

$$V(x) = \frac{4\,\pi}{315} a\, b^2 \left(105 + 42\, c_2 + 9\, c_2^2\right),$$

where `P` has three parameters: a, b, and c_2.

Note

The argument `P` in the `VolumeEPE` function has the same parameters, as those in the `EPE` function.

Author(s)

Peijian Shi <`pjshi@njfu.edu.cn`>, Johan Gielis <`johan.gielis@uantwerpen.be`>, Brady K. Quinn <`Brady.Quinn@dfo-mpo.gc.ca`>.

References

Narushin, V.G., Romanov, M.N., Mishra, B., Griffin, D.K. (2022) Mathematical progression of avian egg shape with associated area and volume determinations. *Annals of the New York Academy of Sciences* 1513, 65−78. doi:10.1111/nyas.14771

Shi, P., Chen, L., Quinn, B.K., Yu, K., Miao, Q., Guo, X., Lian, M., Gielis, J., Niklas, K.J. (2022) Can we simply calculate the volume and surface area of an egg? Evidence from six species of poultry. *Annals of the New York Academy of Sciences* Under preparation.

See Also

`EPE`, `fitEPE`, `SurfaceAreaEPE`

Examples

```
Par3 <- c(4.27, 2.90, 0.0868, 0.0224, -0.0287)
VolumeEPE(P=Par3, simpver=NULL)

# Test the case when simpver = NULL
a    <- Par3[1]
b    <- Par3[2]
c1   <- Par3[3]
c2   <- Par3[4]
c3   <- Par3[5]
pi*4/315*a*b^2*(105+21*c1^2+42*c2+9*c2^2+18*c1*c3+5*c3^2)

myfun <- function(x){
  pi*EPE(P=Par3, x=x, simpver=NULL)^2
}
integrate(myfun, -4.27, 4.27)$value
```

`VolumeETE` *Calculation of the Volume of An Egg Based on the Explicit Troscianko Equation*

Description

`VolumeETE` is used to calculate the volume of an egg that follows the explicit Troscianko equation.

Usage

```
VolumeETE(P, subdivisions = 100L,
         rel.tol = .Machine$double.eps^0.25, abs.tol = rel.tol,
         stop.on.error = TRUE, keep.xy = FALSE, aux = NULL)
```

Arguments

`P`	the parameters of the explicit Troscianko equation.
`subdivisions`	please see the arguments for the `integrate` function in package **stats**.
`rel.tol`	please see the arguments for the `integrate` function in package **stats**.
`abs.tol`	please see the arguments for the `integrate` function in package **stats**.
`stop.on.error`	please see the arguments for the `integrate` function in package **stats**.
`keep.xy`	please see the arguments for the `integrate` function in package **stats**.
`aux`	please see the arguments for the `integrate` function in package **stats**.

Details

The formula of the volume (V) of an egg based on the explicit Troscianko equation is:

$$V(x) = \pi \int_{-a}^{a} y^2 \, dx,$$

where y denotes the explicit Troscianko equation (i.e., `ETE`), and a denotes half the egg's length.

Note

The argument `P` in the `VolumeETE` function has the same parameters, as those in the `ETE` function.

Author(s)

Peijian Shi <`pjshi@njfu.edu.cn`>, Johan Gielis <`johan.gielis@uantwerpen.be`>, Brady K. Quinn <`Brady.Quinn@dfo-mpo.gc.ca`>.

References

Narushin, V.G., Romanov, M.N., Mishra, B., Griffin, D.K. (2022) Mathematical progression of avian egg shape with associated area and volume determinations. *Annals of the New York Academy of Sciences* 1513, 65−78. doi:10.1111/nyas.14771

Shi, P., Chen, L., Quinn, B.K., Yu, K., Miao, Q., Guo, X., Lian, M., Gielis, J., Niklas, K.J. (2022) Can we simply calculate the volume and surface area of an egg? Evidence from six species of poultry. *Annals of the New York Academy of Sciences* Under preparation.

See Also

`ETE`, `fitETE`, `SurfaceAreaETE`

Examples

```
Par5 <- c(2.25, -0.38, -0.29, -0.16)
VolumeETE(P=Par5)

myfun <- function(x){
  pi*ETE(P=Par5, x=x)^2
}
integrate(myfun, -2.25, 2.25)$value
```

`VolumeNRGE` *Calculation of the Volume of An Egg Based on the Narushin-Romanov-Griffin Equation*

Description

`VolumeNRGE` is used to calculate the volume of an egg that follows the Narushin-Romanov-Griffin equation.

Usage

```
VolumeNRGE(P, subdivisions = 100L,
          rel.tol = .Machine$double.eps^0.25, abs.tol = rel.tol,
          stop.on.error = TRUE, keep.xy = FALSE, aux = NULL)
```

Arguments

`P`	the parameters of the Narushin-Romanov-Griffin equation.
`subdivisions`	please see the arguments for the `integrate` function in package **stats**.
`rel.tol`	please see the arguments for the `integrate` function in package **stats**.
`abs.tol`	please see the arguments for the `integrate` function in package **stats**.
`stop.on.error`	please see the arguments for the `integrate` function in package **stats**.
`keep.xy`	please see the arguments for the `integrate` function in package **stats**.
`aux`	please see the arguments for the `integrate` function in package **stats**.

Details

The formula of the volume (V) of an egg based on the Narushin-Romanov-Griffin equation is:

$$V(x) = \pi \int_{-A/2}^{A/2} y^2 \, dx,$$

where y denotes the Narushin-Romanov-Griffin equation (i.e., `NRGE`), and A denotes the egg's length, which is the first element in the parameter vector, `P`.

Note

The argument `P` in the `VolumeNRGE` function has the same parameters, as those in the `NRGE` function.

Author(s)

Peijian Shi <`pjshi@njfu.edu.cn`>, Johan Gielis <`johan.gielis@uantwerpen.be`>, Brady K. Quinn <`Brady.Quinn@dfo-mpo.gc.ca`>.

References

Narushin, V.G., Romanov, M.N., Griffin, D.K. (2021) Egg and math: introducing a universal formula for egg shape. *Annals of the New York Academy of Sciences* 1505, 169−177. doi:10.1111/nyas.14680

Narushin, V.G., Romanov, M.N., Mishra, B., Griffin, D.K. (2022) Mathematical progression of avian egg shape with associated area and volume determinations. *Annals of the New York Academy of Sciences* 1513, 65−78. doi:10.1111/nyas.14771

Shi, P., Chen, L., Quinn, B.K., Yu, K., Miao, Q., Guo, X., Lian, M., Gielis, J., Niklas, K.J. (2022) Can we simply calculate the volume and surface area of an egg? Evidence from six species of poultry. *Annals of the New York Academy of Sciences* Under preparation.

See Also

`curveNRGE`, `fitNRGE`, `NRGE`, `SurfaceAreaNRGE`

Examples

```
Par6 <- c(4.51, 3.18, 0.1227, 2.2284)
VolumeNRGE(P=Par6)

myfun <- function(x){
  pi*NRGE(P=Par6, x=x)^2
}
integrate(myfun, -4.51/2, 4.51/2)$value
```

VolumeSGE *Calculation of the Volume of An Egg Based on the Simplified Gielis Equation*

Description

`VolumeSGE` is used to calculate the volume of an egg that follows the simplified Gielis equation.

Usage

```
VolumeSGE(P, subdivisions = 100L,
          rel.tol = .Machine$double.eps^0.25, abs.tol = rel.tol,
          stop.on.error = TRUE, keep.xy = FALSE, aux = NULL)
```

Arguments

`P`	the parameters of the simplified Gielis equation, including a, n_1, and n_2.
`subdivisions`	please see the arguments for the `integrate` function in package **stats**.
`rel.tol`	please see the arguments for the `integrate` function in package **stats**.
`abs.tol`	please see the arguments for the `integrate` function in package **stats**.
`stop.on.error`	please see the arguments for the `integrate` function in package **stats**.
`keep.xy`	please see the arguments for the `integrate` function in package **stats**.
`aux`	please see the arguments for the `integrate` function in package **stats**.

Details

The formula of the volume (V) of an egg based on the simplified Gielis equation is:

$$V\left(\varphi\right) = \frac{2}{3}\,\pi\int_{0}^{\pi}\sin\left(\varphi\right)\,r^{3}\left(\varphi\right)\,d\varphi,$$

where the polar raidus (r) is the function of the polar angle (φ):

$$r\left(\varphi\right) = a\left(\left|\cos\left(\frac{m}{4}\varphi\right)\right|^{n_2} + \left|\sin\left(\frac{m}{4}\varphi\right)\right|^{n_2}\right)^{-\frac{1}{n_1}},$$

namely the simplified Gielis equation (i.e., `GE`) with arguments `simpver = 1` and `m = 1`.

Note

The argument `P` in the `VolumeSGE` function only has the three parameters: a, n_1, and n_2.

Author(s)

Peijian Shi <`pjshi@njfu.edu.cn`>, Johan Gielis <`johan.gielis@uantwerpen.be`>, Brady K. Quinn <`Brady.Quinn@dfo-mpo.gc.ca`>.

References

Chen, Z. (2012) Volume and area of revolution under polar coordinate system. *Studies in College Mathematics* 15(6), 9−11.

Shi, P., Chen, L., Quinn, B.K., Yu, K., Miao, Q., Guo, X., Lian, M., Gielis, J., Niklas, K.J. (2022) Can we simply calculate the volume and surface area of an egg? Evidence from six species of poultry. *Annals of the New York Academy of Sciences* Under preparation.

See Also

`fitGE`, `GE`, `SurfaceAreaSGE`

Examples

```
Par7 <- c(1.124, 14.86, 49.43)
VolumeSGE(P = Par7)
```

`whitespruce` *Planar Coordinates of* Picea glauca *Tree Rings*

Description

The data consist of the planar coordinates of *Picea glauca* tree rings.

Usage

```
data(whitespruce)
```

Details

In the data set, there are three columns of variables: `Code`, `x`, and `y`. `Code` saves the age codes of tree rings from the 2nd year to the 44th year; `x` saves the x coordinates of the tree rings in the Cartesian coordinate system (cm); and `y` saves the y coordinates of the tree rings in the Cartesian coordinate system (cm).

References

Shi, P., Huang, J., Hui, C., Grissino-Mayer, H.D., Tardif, J., Zhai, L., Wang, F., Li, B. (2015) Capturing spiral radial growth of conifers using the superellipse to model tree-ring geometric shape. *Frontiers in Plant Science* 6, 856. doi:10.3389/fpls.2015.00856

Examples

```
data(whitespruce)

uni.C <- sort( unique(whitespruce$Code) )
Data  <- whitespruce[whitespruce$Code==uni.C[10], ]
x0    <- Data$x
y0    <- Data$y
Res1  <- adjdata(x0, y0, ub.np=2000, len.pro=1/20)

plot( Res1$x, Res1$y, asp=1, cex.lab=1.5, cex.axis=1.5, type="l",
      xlim=c(3, 13), ylim=c(3, 13), col="grey73", lwd=2,
      xlab=expression(italic("x")), ylab=expression(italic("y")) )



  uni.C <- sort( unique(whitespruce$Code) )
  for(i in 1:length(uni.C)){
    Data  <- whitespruce[whitespruce$Code==uni.C[i], ]
    x0    <- Data$x
    y0    <- Data$y

    Res1  <- adjdata(x0, y0, ub.np=200, len.pro=1/10)

    if(i == 1){
      plot( Res1$x, Res1$y, asp=1, cex.lab=1.5, cex.axis=1.5, type="l",
            xlim=c(3, 13), ylim=c(3, 13), col=1, lwd=1,
            xlab=expression(italic("x")), ylab=expression(italic("y")) )
    }
    if(i > 1) lines(Res1$x, Res1$y, col=1, lwd=1)

  }

  uni.C   <- sort( unique(whitespruce$Code) )
  uni.C   <- uni.C[1:12]
  Length  <- c()
  results <- data.frame(Code=c(), x0=c(), y0=c(), theta=c(),
                  a=c(), k=c(), n1=c(), r.sq=c(), RSS=c(), N=c())
  for(i in 1:length(uni.C)){
    Data      <- whitespruce[whitespruce$Code==uni.C[i], ]
    x0        <- Data$x
    y0        <- Data$y
    Res1      <- adjdata(x0, y0, ub.np=200, len.pro=1/10)
    x1        <- Res1$x
    y1        <- Res1$y
    x0.ini    <- mean( x1 )
    y0.ini    <- mean( y1 )
    theta.ini <- c(0, pi/4, pi/2)
    a.ini     <- 0.9
    k.ini     <- 1
    n1.ini    <- c(1.5, 2, 2.5)
    ini.val   <- list(x0.ini, y0.ini, theta.ini,
                      a.ini, k.ini, n1.ini)
```

```
  print(paste("Progress: ", i, "/", length(uni.C), sep=""))
  H <- NULL
  try( H <- fitGE(GE, x=x1, y=y1, ini.val=ini.val,
                m=4, simpver=9, unit="cm", par.list=FALSE,
                stand.fig=FALSE, angle=NULL, fig.opt=FALSE,
                control=list(reltol=1e-20, maxit=20000),
                np=2000), silent=TRUE )
  if(is.null(H)){
    RE <- data.frame(Code=uni.C[i], x0=NA, y0=NA, theta=NA,
           a=NA, k=NA, n1=NA, r.sq=NA, RSS=NA, N=NA)
  }
  if(!is.null(H)){
    RE     <- data.frame(Code=uni.C[i], x0=H$par[1], y0=H$par[2],
              theta=H$par[3], a=H$par[4], k=H$par[5], n1=H$par[6],
              r.sq=H$r.sq, RSS=H$RSS, N=H$sample.size)
    Length <- c(Length, max(max(H$y.stand.pred)[1]-min(H$y.stand.pred)[1],
              max(H$x.stand.pred)[1]-min(H$x.stand.pred)[1])[1])
    if(i == 1){
      plot(H$x.obs, H$y.obs, asp=1, xlim=c(7.4, 8.6), ylim=c(7.4, 8.6),
           cex.lab=1.5, cex.axis=1.5, type="l", lwd=2, col="grey70",
           xlab=expression(italic("x")), ylab=expression(italic("y")))
      lines(H$x.pred, H$y.pred, col=2)
    }
    if(i > 1){
      lines(H$x.obs, H$y.obs, lwd=2, col="grey70")
      lines(H$x.pred, H$y.pred, col=2)
    }

  }
  results <- rbind(results, RE)
}

# To adjust the estimates of partial parameters to ensure k <= 1
results2           <- results
Ind                <- results$k > 1
results2$theta[Ind] <- results$theta[Ind] + pi/2
results2$a[Ind]     <- results$a[Ind] * results$k[Ind]^(1/results$n1[Ind])
results2$k[Ind]     <- 1/results$k[Ind]
results2
Length/2
```

Index

∗ **package**
    biogeom, 15

adjdata, 3, *11*, *13*, *62*, *104*
areaGE, 5, *24*, *49*, *65*, *102*
areaovate, 7, *28*, *57*, *60*, *78*, *80*, *83*

bambooleaves, 9
bilat, 10
biogeom, 15

curveEPE, 18, *37*, *41*, *71*
curveETE, 20, *39*, *45*
curveGE, *7*, 22, *49*, *65*, *102*
curveNRGE, 25, *53*, *86*, *97*, *109*
curveovate, *8*, 26, *57*, *78*, *80*, *83*

DEPE, 28, *29*, *94*
DETE, *30*, 30, *95*
DNRGE, 31, *32*, *97*
DSGE, 33, *34*, *49*, *65*, *98*

eggs, 34
EPE, *19*, *29*, 36, *37*, *88*, *93*, *94*, *106*
ETE, *22*, *30*, *31*, *38*, 38, *95*, *99*, *107*, *108*

fitEPE, *19*, *29*, *37*, 39, *71*, *94*, *106*
fitETE, *22*, *31*, *38*, *39*, 43, *72*, *73*, *75*, *95*, *99*, *108*
fitGE, *7*, *24*, *34*, 46, *65*, *98*, *102*, *111*
fitNRGE, *26*, *33*, 51, *86*, *97*, *109*
fitovate, *8*, *28*, 55, *78*, *80*, *83*
fitsigmoid, 58, *91*
fracdim, *13*, 61, *105*

GE, *7*, *24*, *34*, *49*, 63, *97*, *98*, *101*, *102*, *110*, *111*
ginkgoseed, 66

integrate, *6*, *7*, *59*, *90*, *93*, *94*, *96*, *97*, *107*, *108*, *110*
intersect.owin, *13*

kp, 67

lmPE, *19*, *37*, *41*, 68, *88*, *104*
lmTE, *38*, *45*, 72, *73*, *99*
locator, *11*

MbetaE, *8*, *27*, *28*, *55*, *57*, *59*, *60*, 76, *80*, *83*, *91*
MBriereE, *8*, *27*, *28*, *55*, *57*, *59*, *60*, *78*, 79, *83*, *91*
MLRFE, *7*, *8*, *27*, *28*, *55*, *57*, *59*, *60*, *78*, *80*, 81, *91*

Neocinnamomum, 83
NRGE, *25*, *26*, *32*, *33*, *52*, *53*, 84, *96*, *97*, *109*

optim, *40*, *43*, *44*, *47*, *48*, *51*, *55*, *56*, *59*

PE, *19*, *37*, *41*, *71*, 86, *103*, *104*

quantile, *3*

set.seed, *11*
shoots, 89
sigmoid, *8*, *60*, *78*, *80*, *83*, 90
starfish, 92
SurfaceAreaEPE, *29*, *37*, *93*, 93, *106*
SurfaceAreaETE, *31*, *39*, 94, *95*, *108*
SurfaceAreaNRGE, *33*, *86*, 95, *96*, *109*
SurfaceAreaSGE, *34*, *49*, *65*, 97, *98*, *111*

TE, *38*, *45*, *75*, 98
TGE, *7*, *24*, *49*, *65*, 100
TSE, *19*, *37*, *41*, *71*, *88*, 102

veins, *63*, 104
VolumeEPE, *37*, *94*, 105, *106*
VolumeETE, *39*, *95*, *107*, 107
VolumeNRGE, *86*, *97*, 108, *109*
VolumeSGE, *49*, *65*, *98*, *110*, 110

whitespruce, 111
wilcox.test, *12*