Package 'logspiral'

August 8, 2020

Type Package

Title Generates and fits logarithmic spirals

Version 0.9.6

Date 2020-08-08

Author A.E. (Tony) Aldridge

Maintainer Tony Aldridge <tony@southnet.co.nz>

Description Fits linear and curved logarithmic spirals to outline coordinates.

Suggests MASS,
wmtsa

Depends R (>= 3.5.0)

License GPL (>= 2)

LazyData true

R topics documented:

`logspiral-package` *Fits a range of spirals to a sequence of outline coordinates in two dimensions(2D)*

Description

Fits **log-piecewise** and **log-polynomial** spirals to an outline. Deviations are extracted from each spiral fit.

Outline smoothing provides alternative non-spiral based deviations. Also available is a function that can generate up to four connected or log-piecewise spirals with user-specified spiral angles, change locations, noise and the number of cycles. Other functions can generate user-specified curved or nonlinear spirals. Generated spirals are useful for investigating spiral fits and resulting deviations against actual outline data.

A spiral for an unknown outline can be indicated from a catalogue of twelve types of spiral (log-piecewise and log-curved).

Details

Fitting uses the base R function **nls** with the spiral(s) as exponential equations for minimisation of residual sums of squares. Other optimisers have been investigated (eg nlxb, ms, optim), but **nls** remains reliable when initial parameter values are near to the optimum.

Alternative methods for optimisation are Monte Carlo and particle swarm. However, Monte Carlo (JAGS, STAN) is not yet possible because independent variables also need to be updated as the axis location is changed when searching for an optimum. Particle swarms similarly fail but mght be are better for model simulations in a designed search (e.g. see the Latin hypercube in `hydroPSO`).

Developers, please note that functions in logspiral have been mostly created and tested as standalone code. So the names of variables are not always consistent between functions. There has been the convention of prefixing temporary or working variables with the letters x, xx, yy, z or zz. The resulting code is sometimes not pretty and should be condensed in later versions of logspiral. These variable names are different from the usual 2D coordinate names of x,y.

Output from all fitting is in terms of intercept and slope on a logarithmic scale of radius versus angle about an estimated axis location. The outline must start near an umbo (zero angle) and turns anti-clockwise for the slope to be positive. Outlines turning clockwise are flipped to be anticlockwise along with a warning message. Once warned, you should flipping and translate your outline coordinates before any final spiral fit. A clockwise turning outline with all or some negative coordinates may not produce a sensible fit.

The slope (also known as the expansion rate of a spiral) is reported as spiral angle (k) in radian units, which by default is converted to degrees as (180/pi)*atan(1/k). Degree units are more easily interpreted. A spiral angle of 90 degrees is a circle, while an spiral angle of zero degrees is a straight line. The spiral expansion rate as a spiral angle parameter must not be confused with angle turned about the spiral axis.

Spiral angle has also been reported as a whorl expansion (W) in Raup (1966), and in McGhee (1980). Spiral angle (k) in radians converts to the whorl expansion in natural logarithm units with multiplication by `2*pi` (i.e. assigment as an object, lnW: `lnW <-2*pi*k`).

Package:	logspiral
Type:	Package
Version:	0.9.6
Date:	2020-08-08

License: GPL (>= 2)

`generateAnyLinear fitAnyLinear fitCheckL2Spiral` are key functions to demonstrate this package. `doSetupGraphics` opens four graphic windows, or alternatively, repeat `dev.new()` four times to display results. Examples throughout assume four graphic windows are open as devices 2, 3, 4, and 5.

Author(s)

A.E. Aldridge

Maintainer: A.E. Aldridge <tony@southnet.co.nz>

References

Aldridge (1998) Brachiopod outline and the importance of the logarithmic spiral. Paleobiology 24, 215-226

Aldridge (1999) Brachiopod outline and episodic growth. Paleobiology 25, 471-482

Aldridge (2009) Can beak shape help to research the life history of squid? New Zealand Journal of Marine and Freshwater Research 43 1061-1067

Aldridge & Gaspard D. (2011) Brachiopod life histories from spiral deviations in shell shape and microstructural signature - preliminary report. Memoirs of the Association of Australasian Paleontologists 41, 256-268

Clark J., et al (2015) Application of shell spiral deviations methodology to fossil brachiopods. Palaeontologica Electronica 18.3.54A:1-39

Gaspard D., Aldridge et al (2018) Analysis of growth and form in *Aerothyris kerguelenensis* (rhynchonelliform brachiopod) - Shell spiral deviations, microstructure, trace element contents and stable isotope ratios. Chemical Geology, 483: 474-490

McGhee, G.R.(1980) Shell form in the biconvex articulate Brachiopoda: a geometric analysis. Paleobiology 6, 57-76.

Perez-Huerta A., et al (2014) Brachiopod shell spiral deviations (SSD): implications for trace element proxies. Chemical Geology 374-375, 13-24.

Raup D.M. (1966) Geometric analysis of shell coiling: general problems. Journal of Paleontology, 40(5), 1178-1190.

`arcSmooth2D` *smooth (spline) an outline and output deviations*

Description

Smooth an outline using arc length and splines

Usage

```
arcSmooth2D(x = x, y = y, smooth.factor = NULL,summary=FALSE,
 plot.deviation=FALSE, plot.it = FALSE, print.summary= FALSE,
output = TRUE)
```

Arguments

x	x coord
y	y coord
smooth.factor	spar value between zero and 1
summary	print a summary of the smooth fit
plot.deviation	no plotting of deviation as a default
plot.it	plot
print.summary	print a summary of the smoothing
output	output yes or no

Details

See Base R `smooth.spline` documentation for the range of spar (smooth parameter) values. A NULL value for smooth.factor forces an estimate of a smoothing factor (see **bestSmooth**).

Value

arc	arc length (starting at zero)
deviation	signed perpendicular distance from smooth to the outline coordinates
x, y	coordinates of the smoothed outline

Note

Each coordinate (x, y) is separately smoothed against arc length for fitted x, y as a smooth. Too much smoothing loses detail in spiral deviations.

A smooth factor near to 1.0 has large outline deviations that merges useful detail (e.g. periodicities) with the background noise. On the other hand, a factor close to zero tracks the outline and has little variation in deviations. The best smooth (see the function **bestSmooth**) aims to remove background noise and leave only the useful deviation detail.

Author(s)

A.E. Aldridge

References

Miller J. (2009) Shape curve analysis using curvature. PhD Thesis Univerisity of Glasglow

http://theses.gla.ac.uk/854/1/2009millerphd.pdf (accessed 24 Sept. 2019)

Maxwell E.A. (1958) Elementary Coordinate Geometry page 81

Examples

```
# create four graphic windows
doSetupGraphics()
# generate a two spiral outline with noise
x.df <- generateAnyLinear( k1=30, k2=30, k3=40, k4=40, noise=0.1)
xx <- x.df$x
yy <- x.df$y
## outline is smoothed using spline and arc length using default smooth value
```

```
zsmooth <- arcSmooth2D(x=xx, y= yy, plot.it=TRUE)

## determine a reasonable spar value (see details)
bestspar <- bestSmooth( xx, yy, plot.it=TRUE)
# can also print bestspar to view the value
bestspar

## now smooth the outline using this value
zsmooth <- arcSmooth2D( xx,yy, smooth.factor=bestspar, plot.it=TRUE)

## automatically choose best spar and produce smooth and deviations
zsmooth <- arcSmooth2D( xx,yy, smooth.factor=NULL, plot.it=FALSE)
plot( zsmooth$arc, zsmooth$zdeviation, type="l")
```

`bestSmooth` *finds the best smoothing value to use with arcSmooth2D*

Description

Find the best spline by stepping through values of smoothing value, spar. Criterion is maximum curvature of increase in standard deviation about smooth Alternatively, the best spar is matched to a a user specified target noise value.

Usage

```
bestSmooth(x, y, start = 0.01, end = 0.98, delta = 0.005, target = NULL, plot.it = FALSE)
```

Arguments

`x`	x coordinate
`y`	y coordinate
`start`	beginning spar value (default 0.005)
`end`	end spar value (default 0.98)
`delta`	step increase in spar value (default 0.005)
`target`	the noise target to match with spar
`plot.it`	plots of standard deviation and curvature with increasing spar values (default no plot)

Details

spar is the value for smoothing parameter see the help for **smooth.spline** in Base R. The best smoothing value is defined when the rate of change in standard devation (curvature) is a maximum with increasing spar values. Useful to set `plot.it=TRUE` to view that curvature does indeed have a global maximum. Plots are placed on graphic windows 2 and 3.

Value

`best`	the spar value at max curvature of increasing std deviation

Note

See details on `smooth.spline` where spar set by cross validation. Too much smoothing and we lose important detail on the actual outline deviations. Too little overfits.

If a target noise is supplied the spar is found to match this noise in the 2D outline vertical (perpendicular) deviations from smooth (see **arcSmooth2D**).

Author(s)

A.E. Aldridge

References

see `help(smooth.spline)` in base R

See Also

`doSetupGraphics arcSmooth2D doCancelNoise`

Examples

```
## Not run:
# create four graphic windows
doSetupGraphics()
# generate a two spiral (L2) outline with noise
x.df <- generateAnyLinear( k1=30, k2=30, k3=40, k4=40,noise=0.1)
#
## outline smoothed using spline and arc length using default smooth value
zsmooth <- arcSmooth2D(x=x.df$x, y=x.df$y, plot.it=TRUE)
#
## determine a reasonable spar value (see details)
xspar <- bestSmooth( xx, yy, plot.it=TRUE)
# can print xspar to view the value
## outline smoothed using spline and arc length using best smooth value
zsmooth <- arcSmooth2D(xx, yy, smooth.factor=xspar, plot.it=TRUE)
zsmooth <- arcSmooth2D(xx, yy, smooth.factor=0.78, plot.it=TRUE)

## End(Not run)
```

`catalogueDeviationFeatures` *Catalogue of 30 features from 838 instances of 11 types of logarithmic spiral.*

Description

Summary of features from fitting a single spiral to 838 different spirals

Usage

```
data(catalogueDeviationFeatures)
```

Format

A data frame with 838 observations on the following 43 variables.

`seq` row number in Catalogue

`k1` spiral angle 1 (degree units)

`k2` spiral angle 2 (degree units)

`k3` spiral angle 3 (degree units)

`k4` spiral angle 4 (degree units)

`dist` the distance from axis to start of outline

`c1` location of 1st change as proportion of total outline points

`c2` location of 2nd change as proportion of total outline points

`c3` location of 3rd change as proportion of total outline points

`quadratic` the value of 2nd power of angle about axis

`cubic` the value of 3rd power of angle about axis

`quartic` the value of 4th power of angle about axis

`type` numeric code for spiral type - used in sorting and plotting

`W1` Wavelet variance from highest (W1) to lowest (W10)frequency..

`W2` Wavelet variance (W2)

`W3` Wavelet variance (W3)

`W4` Wavelet variance (W4)

`W5` Wavelet variance (W5)

`W6` Wavelet variance (W6)

`W7` Wavelet variance (W7)

`W8` Wavelet variance (W8)

`W9` Wavelet variance (W9)

`W10` highest Wavelet frequency variance (N=1024)

`F01` Fourier variance from lowest (F01) to highest (F20) frequency...

`F02` Fourier variance (F02)

`F03` Fourier variance (F03)

`F04` Fourier variance (F04)

`F05` Fourier variance (F05)

`F06` Fourier variance (F06)

`F07` Fourier variance (F07)

`F08` Fourier variance (F08)

`F09` Fourier variance (F09)

`F10` Fourier variance (F10)

`F11` Fourier variance (F11)

`F12` Fourier variance (F12)

`F13` Fourier variance (F13)

`F14` Fourier variance (F14)

`F15` Fourier variance (F15)

`F16` Fourier variance (F16)

`F17` Fourier variance (F17)

`F18` Fourier variance (F18)

`F19` Fourier variance (F19)

`F20` highest Fourier variance (N=1024)

Details

The first 13 columns describe the type of spiral shape, next ten columns are the wavelet variances (using **wmtsa** package and defaults for 1024 observations). The remaining 20 columns are Fourier spectrum variances using base R's **fft** function with the same N=1024.

Source

The **generate** functions of logspiral were used to output a shape (NOT equal spaced coordinates). Then the single spiral fitting was applied to extract the deviations, which were then interpolated to 1024 and scaled to 10 vertical units and 100 horizontal (arc length) units.

References

William Constantine and Donald Percival (2016). wmtsa: Wavelet Methods for Time Series Analysis. R package version 2.0-2. https://CRAN.R-project.org/package=wmtsa

Percival, D.B and Walden A.T. Donald Percival (2000) Wavelet Methods for Time Series Analysis. Cambridge.

Examples

```
data(catalogueDeviationFeatures)
#
# print (to screen) the first four rows and 13 columns in the data frame
catalogueDeviationFeatures[1:4, 1:13]
#
# table the types (numeric code)of spiral in the catalogue
table(catalogueDeviationFeatures$type)
#
# total number of instances of spirals in the catalogue
sum(table(catalogueDeviationFeatures$type))
```

---

compareFeaturesAB *Compares deviation features of TWO outlines, each fitted with a single (L1) spiral*

Description

Compares the single logarithmic spiral features of TWO outlines

Usage

```
compareFeaturesAB(spiralA = spiralA, spiralB = spiralB, print.it = FALSE)
```

Arguments

spiralA	Deviation features of first outline (A).
spiralB	Deviation features of second outline (B)
print.it	print the distance between the two outlines

Details

Local distance is based on Wavelet features (variance) and global distance is based on Fourier features (variance).

Value

distance between two spirals: local and global.

Warning

Never a perfect match to input outline. Check general pattern of the eleven spirals. Also consider shapes NOT in the Catalogue (e.g double quadratic, and bent cable to linear, LQL2) then compare deviations

Note

Closest spirals in the catalogue can be created and tweaked using the printed type of spiral and generating similar spirals (see functions beginning with 'generate' as in (for example) generate-QuadraticLinear and generateQuadratic.

Author(s)

A.E. Aldridge

References

Choi, H., Ombao, H., Ray, B. (2008). Sequential Change-Point Detection Methods for Nonstationary Time Series. Technometrics, 50(1):40-52.

William Constantine and Donald Percival (2016). wmtsa: Wavelet Methods for Time Series Analysis. R package version 2.0-2. https://CRAN.R-project.org/package=wmtsa

Percival, D.B and Walden A.T. Donald Percival (2000) Wavelet Methods for Time Series Analysis. Cambridge.

See Also

catalogueDeviationFeatures, computeDeviationsFeatures

Examples

```
## Not run:
za <- generateAnyLinear( k1=30,k2=30,k3=40,k4=40)
zb <- generateAnyLinear( k1=40,k2=45,k3=50,k4=50)
aspiral <- computeL1DeviationsFeatures(x=za$x, y=za$y)
bspiral <- computeL1DeviationsFeatures(x=zb$x, y=zb$y)
compareFeaturesAB(spiralA=aspiral, spiralB=bspiral)

## End(Not run)
```

compareFeaturesToCatalogue
Compares deviation features of an outline to a catalogue of spirals.

Description

Compares features of an outline to features to a catalogue then outputs the nearest NINE spirals (ranked) as a prior choice for fitting the outline.

Usage

```
compareFeaturesToCatalogue(spiral.features = spiral.features,  print.it = FALSE,
             plot.detail = FALSE,output.results=FALSE, graphic.windows = c(2, 3, 4, 5))
```

Arguments

spiral.features	the output from fitting a single log spiral to an outline
print.it	print summary of NINE nearest spirals
plot.detail	plot the Wavelet (local) and Fourier (global) detail in separate windows
output.results	list of the nine closest local (Wavelet) and nine closest global (Fourier) distances
graphic.windows	graphics windows to use in the plotting 4 = default output, detail in 3 = Wavelet (all), 2 = Fourier (all), 5 = local & global ratios for ranked spirals

Details

Default output is a graph of local (Wavelet) and global (Fourier) features in order of closest fit. Measure of closeness is the symmetric ratio of Choi et al (2008).

Value

wratio.diff.sum	sum of local, Wavelet differences
fdiff.sum	sum of global, Fourier differences

Warning

Never a perfect match to input outline because of periodicities in actual growth. Check the general pattern of the nine ranked spirals. Also consider shapes NOT in the Catalogue (e.g. double quadratic, bent cable to linear LQL2).

Note

Closest spirals in the catalogue can be created and tweaked using the printed type of spiral and generating similar spirals (see functions beginning with 'generate' as in (for example) generateQuadraticLinear and generateQuadratic.

Author(s)

A.E. Aldridge

References

Choi, H., Ombao, H., Ray, B. (2008). Sequential Change-Point Detection Methods for Nonstationary Time Series. Technometrics, 50(1):40-52.

William Constantine and Donald Percival (2016). wmtsa: Wavelet Methods for Time Series Analysis. R package version 2.0-2. https://CRAN.R-project.org/package=wmtsa

Percival, D.B and Walden A.T. Donald Percival (2000) Wavelet Methods for Time Series Analysis. Cambridge.

See Also

`catalogueDeviationFeatures` `computeDeviationFeatures` `compareFeaturesAB`

Examples

```
## Not run:
z <- generateAnyLinear( k1=40, k2=45, k3=50, k4=60) # four spirals
zfeatures <- computeL1DeviationFeatures( x = z$x, y=z$y)
compareFeaturesToCatalogue(zfeatures)
compareFeaturesToCatalogue(zfeatures, plot.detail=TRUE)
zout <- compareFeaturesToCatalogue(zfeatures, plot.detail=TRUE, output.results=TRUE)

## End(Not run)
```

---

compareSpiralFits *Summarises and compares spirals on partitioned variation*

Description

Compares up to five fitted spirals using three partitions of variation (see details)

Usage

```
compareSpiralFits(s1 = NULL, s2 = NULL, s3 = NULL, s4 = NULL, s5 = NULL, ratios = TRUE)
```

Arguments

`s1`	output from first spiral
`s2`	output from second spiral
`s3`	output from third spiral
`s4`	output from fourth spiral
`s5`	output from fifth spiral
`ratios`	should a table of ratios be produced (default TRUE)

Details

Ratios only computed and output if TWO or more spirals are included.

Sums of squares about zero spiral deviation are estimated at each Fourier frequency. Estimation is through the fast Fourier transform when calling the function `partitionSumsSquares`.

Variation is partitioned as low frequency, growth periodicity, and high frequency. Percentages of total variation is tablulated for all spirals. Low frequency combines the Fourier frequencies 1 to 4, periodicity combines 5 to 19 and the high frequency from 20 to the the maximum (n/2).

If the low frequency partition is more than half the total variation then the fitted spiral is probably the wrong type for the outline.

If ratios=TRUE, then spiral partitions are divided in the order input. So the first ratio will be spiral s1 compared to s2 as a ratio for each of the three partitions.

Value

two tables (if ratios=TRUE) with percentage variation

Note

when only one spiral is input then only percentages of total variation are output.

Author(s)

A.E. Aldridge

References

no references yet

See Also

partitionSumsSquares

Examples

```
## Not run:

# generate and fit a spiral using defaults
#
z <- generateAnyLinear()
xs <- fitAnyLinear(x=z$x, y=z$y)
#
# a single (L1) spiral with 35 degree spiral angle
# using function shows that 80
# deviations is in the high frequency components.
#
compareSpiralFits(xs)
#
# Now generate a L2 spiral with change at 100th coordinate
# then fit a L1 spiral and a L2 spiral to this outline.
#
z <- generateAnyLinear(k1=35, k2=35, k3=40, k4=40)
xs1 <- fitAnyLinear(x=z$x, y=z$y)
xs2 <- fitAnyLinear(x=z$x, y=z$y, m=100)
#
```

```
# Compare deviations from fitting L1, L2 spirals to the same
# outline data (z)
#
compareSpiralFits(s1 = xs1, s2 = xs2)
#
# Shows that xs1 (the L1 fit) has 73 percent of variance in the low
# frequencies so this is not a good fit, xs2 has only 1.7 percent.
# Periodicity (cyclic) frequencies exhaust ~ 15 to 18 percent of
# variation and deemed part of the random or noise component
# along with high frequency components.

## End(Not run)
```

`computeDeviationWavelets` *Wavelet decomposition of spiral deviations*

Description

Computes a wavelet decomposition and plots the one with maximum variance

Usage

```
computeDeviationWavelets(spiral.output = spiral.output, wave.levels = NULL,
     print.var = TRUE, plot.var = TRUE, plot.maxWave = TRUE, units = "mm")
```

Arguments

`spiral.output`	output from a spiral fit
`wave.levels`	number of levels (powers of 2) default is NULL see details
`print.var`	print variances at each level as a percentage of total variance
`plot.var`	plot the variance and confidence limits (see details)
`plot.maxWave`	wavelet decomposition column with maximum variance
`units`	units to include on plot

Details

Levels are the maximum power of 2 less than number of deviations. Actual levels are less than maximum because of the default width and type of mother wavelet used(MODWT with least asymmetric filter, Percival & Walden 2000).

Variances and confidence limits as in Percival & Walden (2000) pages 311-315. The default scales have units that are powers of 10. For example a 2^5 scale, would be log base 10 of 32 = 1.505. Same for variances.

Alternative is to plot variances from the wmtsa function wavVar and change the mother wavelet. See example below, where less mother wavelet resolution provides some estimate at all scales possible (7 in this case). wavVar stops with an error message if you ask too many levels for the wavelet specified.

Value

data frame with rows the number of deviations and columns as values of each wavelet in the decomposition

Note

see the *wmtsa* help for `wavMODWT`, `wavMRD`, and `wavVar` that are used in this function

Author(s)

A.E. Aldridge

References

William Constantine and Donald Percival (2016). wmtsa: Wavelet Methods for Time Series Analysis. R package version 2.0-2. https://CRAN.R-project.org/package=wmtsa

Percival, D.B and Walden, A.T. Donald Percival (2000) Wavelet Methods for Time Series Analysis. Cambridge.

See Also

`doCancelNoise PeakCycle`

Examples

```
## Not run:
# a data frame of outline (x,y) coordinates based on a L1 spiral with six cycles
xy.df <- generateAnyLinear( waves=6)
# default points generated is 200, so max wavelet scale is 2^7 = 128
#
spiral.fit <- fitAnyLinear(xy.df$x, xy.df$y)  # fits the L1 spiral
## multi-resolution of deviations into wavelets
# will load the package wmtsa if not already loaded.
#
 dev.waves <- computeDeviationWavelets( spiral.fit )
# default will plot deviations along with max variance wavelet scale
# in this case scale 5 or 2^5 = 32 units for the wavelength
# default generation of L2 is 200 coordinates so 200/32 = 6 cycles
#
## Check by plotting.  Wavelet scales 4 and 5 explains 90 percent
# of all spiral deviation variance. Verify by adding together wavelet
# columns 4 and 5 in the object dev.waves
#
plot ( spiral.fit$arc.length, spiral.fit$deviations, type = "s")
lines (spiral.fit$arc.length, dev.waves[ , 4] + dev.waves[ , 5], col= 4, lwd=1.5)
#
## alternative wavelet variance plotting using wmtsa function wavVar
# first the number scales used in computeWaveletDeviations,
# then the full levels using shorter, but less reliable mother wavelets
#
plot(wmtsa::wavVar(spiral.fit$deviations, wavelet = "s4", n.levels = 5))
plot(wmtsa::wavVar(spiral.fit$deviations, wavelet = "haar", n.levels = 7))
plot(wmtsa::wavVar(spiral.fit$deviations, wavelet = "s2", n.levels = 7))
#
```

```
## End(Not run)
```

`computeL1DeviationFeatures` *Computes local and global features of deviations from a single spiral*

Description

Computes 10 local and 20 global features of deviations from a single spiral

Usage

```
computeL1DeviationFeatures(x = x, y = y, plot.it = T)
```

Arguments

`x`	x coordinate of outline
`y`	y coordinate of an outline
`plot.it`	plot (default) the deviations

Details

10 local features are the wavelet variances, and 20 global features are variances from the first twenty Fourier frequencies. First frequency is the variance around mean, which should be zero.

The 1024 interpolated (equal arc length) deviations are reduced to ten wavelet scales. Fourier spectrum is computed using the base R fast Fourier transform (fft) while the wavelet spectrum (also called scalogram) uses the R package wmtsa (Constantine and Percival, 2011). Wavelet scalogram applies the default maximal overlap, discrete transform (MODWT) with the default least asymmetric filter (Percival and Walden, 2000).

Value

numeric vector of 30 values

Note

Wavelet features seem the most useful. Might only need ten (not 20) Fourier frequencies.

Author(s)

A.E. Aldridge

References

William Constantine and Donald Percival (2016). wmtsa: Wavelet Methods for Time Series Analysis. R package version 2.0-2. https://CRAN.R-project.org/package=wmtsa

Percival, D.B and Walden, A.T. Donald Percival (2000) Wavelet Methods for Time Series Analysis. Cambridge.

Fast Fourier Transform (`fft`) in base R. See `help(fft)`

See Also

wmtsa::wavMODWT doSpectrum

doVariogram

Examples

```
## Not run:
xy <- generateAnyLinear(k1=30, k2=30, k3=40, k4=40)
z <- computeL1DeviationFeatures(x=xy$x, y=xy$y)
compareFeaturesToCatalogue(z)

## End(Not run)
```

---

convertMetricToPixel *converts metric coordinates to raster pixel coordinates*

Description

metric coordinates are returned to the pixel or raster coordinates that were used to digitise the outline of an image

Usage

```
convertMetricToPixel(xeqn = c(1,0,0), yeqn = c(0,1,0), x = x, y = y,
                     check.input = FALSE, check.output = FALSE)
```

Arguments

xeqn	the pixel to metric equation for x outline coordinate (three numbers)
yeqn	the pixel to metric equation for x outline coordinate (three numbers)
x	the x coordinates to be converted
y	the y coordinates to be converted.
check.input	option to print out the input transformation values for the conversion
check.output	option to print the output (inverse matrix) of the transformation values

Details

Mostly for a Euclidean transformation : translation and rotation. Affine is also possible. Default setting is the Identity matrix for no transformation.

The equation numbers are in the order: x coefficient, y coefficient, and intercept. In 2D there is one equation for the x coordinate (xeqn) and one for the y coordinate (yeqn). Input coordinates were computed through the transformation 2 x 2 matrix (elements a, subscripts i,j) as:

x(metric) = a11*x(pixel) + a12*y(pixel) + xintercept

y(metric) = a21*x(pixel) + a22*y(pixel) + yintercept

The output coordinates use the inverse 2 x 2 matrix (elements b, subscripts i,j)

x(pixel) = b11*(x(metric) - xintercept) + b12*(y(metric) - yintercept)

y(pixel) = b21*(x(metric) - xintercept) + b22*(y(metric) - yintercept

Equation can be produced by the digitising software used or be calculated using linear regression. For example, in Vextractor the key F6 gives the reference points used in the digitising scale along with the equations for conversion of pixels to the user units (usually millimetres).

Useful to have the converted coordinates for plotting the fitted spiral, deviations, and spiral axis location on top of the original image.

See the vignette *fitGiantSquid*

Value

coordinates in the scale that was used in the digitised image.

Note

assumes matrix inversion is non singular.

Author(s)

A.E. Aldridge

References

Brannan, D.A., Esplen, M.F., Gray, J.J. 2012. Geometry (2nd ed). Cambridge University Press.

Simon Barthelme (2019). imager: Image Processing Library Based on 'CImg'. R package version 0.41.2. https://CRAN.R-project.org/package=imager

Vextractor x64 7.1 http://www.vextrasoft.com/

See Also

`eqscplot` in MASS

Examples

```
## Not run:
data(exGiantSquidBeak)
xy.df <- exGiantSquidBeak
x <- xy.df$x; y <- xy.df$y  # outline (x,y) coordinates
spiral.fit <- fitAnyLinear(x=x, y=y)  # fitted spiral (L1 in this example)
#
x.axis <- spiral.fit$parameters1[1] # estimated L1 spiral axis location
y.axis <- spiral.fit$parameters1[2]
x <- spiral.fit$xpred.orig # fitted spiral coordinates in original(metric) values.
y <- spiral.fit$ypred.orig
# not the correct transformation - used as example
xeqn <- c(0.0153728, 8.62282*10^-5, 70.0251)
yeqn <- c(-1.1789*10^-5,-0.0153494,45.4761)
#
# convert to pixels the spiral axis and fitted values.
#
xy.axis <- convertMetricToPixel(xeqn=xeqn, yeqn=yeqn, x=x.axis, y=y.axis)
xy.image.df <-  convertMetricToPixel(xeqn=xeqn, yeqn=yeqn, x=x, y=y)
#
# image input as an object and pixels values as an overlay
#
library(imager)  # must be installed first - not part of logspiral requirements (yet)
```

```
GiantSquidBeak <- load.image(file="####.JPG")
plot( GiantSquidBeak, axes = F)
points( xy.image.df$xx, xy.image$yy, cex=0.5, col=2) # overlay fitted points in red
points ( xy.axis[1], xy.axis[2], pch=10,cex=1.5, col=2) # spiral axis location in red
# end of plotting image with overlay of points (can save as pdf, jpg, etc)
rm( GiantSquidBeak ) # remove what can be a rather large object in the workspace.


## End(Not run)
```

`cumulativeAngle` *accumulates angle (radians)*

Description

accumulates angle (radians) along an outline about a fixed point

Usage

```
cumulativeAngle(x, y, u, v)
```

Arguments

x	x coordinate
y	y coordinate
u	x coordinate of fixed point such as a spiral axis location
v	y coordinate of fixed point such as a spiral axis location

Details

Checks whether coordinate sequence is regressing and changes sign accordingly. Function used as part of spiral fitting see fitAnyLinear.

Angle starts close to zero (at angle eps = 0.00001).

Value

theta	the accumulated angle (radians) about the fixed point (u,v)

Warning

zig-zags or growth regressing in an accretionary outline

Note

Beware zig-zags when points are dense: don't overdo the digitising. Useful to print and plot detail in the outline. Drawing segments from fixed point to outline can verify that the accumulated angle is correct.

Author(s)

A.E. Aldridge

References

https://en.wikipedia.org/wiki/Dot_product

See Also

`rotate2D fitAnyLinear equalSpace`

Examples

```
## Not run:
MASS:: eqscplot( c(4,0,-3,1), c(-5,5,10,15), pch=c("1","2", "3", "4"))
 points(6,4, pch=10, cex=2)

# clockwise produces negative angles
cumulativeAngle(  c(4,0,-3,1), c(-5,5,10,15), u=6,v=4)
# anti-clockwise has positive angles
cumulativeAngle(  rev(c(4,0,-3,1)), rev(c(-5,5,10,15)), u=6,v=4)

## End(Not run)
```

`curvatureNoPenalty` *compute outline curvature using spline smoothing*

Description

compute curvature along an outline and derivatives

Usage

```
curvatureNoPenalty(x, y, smooth.factor = 5)
```

Arguments

`x`	x coord of outline (x,y)
`y`	y coord of outline (x,y)
`smooth.factor`	degrees of freedom, up to the number of unique x values

Details

see `smooth.spline` in base R, and differential geometry for curvature formula

For a single linear spiral (L1) a plot of 1/curvature vs radius will produce a line of slope 1/sin(spiral angle). Or,plot 1/curvature vs arc length from umbo to see the slope. a change in slope is masked by noise unless there is a >= 5 degree difference in spiral angles and low level of outline noise.

Note that smoothing uses base R smooth.spline, which does not smooth the second derivative, otherwise use smooth.Pspline with order=4 and method =2, from the package spline.

The outline coordinates are assumed NOT to be a closed shape.

Value

arc	arc length from first coordinate pair
curvature	curvature value
zdx1	first derivative in x direction
zdx2	second derivative in x
zdy1	first derivative in y direction
zdy2	second derivature in y

Note

Curvature is often unstable at extremes of outline (eg near axis of a spiral). When plotting curvature may need to trim values or limit the curvature axis using ylim.

Alternative method is to use circle fitting to arcs along the outline. Curvature is the reciprocal of the circle radius. Stable circle fitting (Chernov, 2010) can be applied. The curvature at outline extremes is better but overall curvature is too noisy to be useful when comparing spirals. Need for a better algorithm that varies outline points according to curvature. Such an algorithm described in Lin et al (2010) and Allward, M. (2019), but is not implemented yet in logspiral.

Penalised splines are NOT used in this function as they risk over-smoothing the outline especially and miss major changes in spirals. However, smoothing 2nd derivative through spline::smooth.Pspline is the better method. The function is still in progresss.

Author(s)

A.E. Aldridge

References

Chernov N.(2010) Circular and Linear Regression. https://people.cas.uab.edu/~mosya/cl/ (accessed 24 Nov, 2019) Note links to code for fitting.

Gray et al (1997) Modern differential geometry of curves and surfaces with Mathematica (2nd Edition).

Lin et al (2010) Robust and Accurate Curvature Estimation using Adaptive Line Integrals. Journal on Advances in Signal Processing. https://doi.org/10.1155/2010/240309

Miller James (2009) Shape curve analysis using curvature. PhD Thesis Univeristy of Glasglow http://theses.gla.ac.uk/854/1/2009millerphd.pdf (accessed 23 Sept 2019)

See Also

arcSmooth2D

Examples

```
## Not run:
require(MASS)
doSetupGraphics() # creates four graphic windows
# generate L1 spiral outline with some noise
#
x.df <- generateAnyLinear( k1=40, k2=40, k3=40,k4=40, noise=0.1)
#
## outline smoothed using spline and arc length using a smooth
#   value greater than default
```

```
#
zc <- curvatureNoPenalty( x.df$x, x.df$y, smooth.factor=15)
#
## spiral parameter values of the generated L1 outline
a <- 2
k <- 1/tan(40*pi/180)
#
## curvature vs arc length and actual relationship
#
dev.nex()
plot( zc$arc, zc$curvature)
lines (zc$arc,1/( k*zc$arc + a*sqrt(1+k^2)), col=4, lwd=1.5)
#
# alternatively
dev.next()
plot( zc$arc, 1/zc$curvature, ylim=c(0,200))
lines (zc$arc,k*zc$arc + a*sqrt(1+k^2), col=4, lwd=1.5)

## End(Not run)
#
```

`doCancelNoise` *Cancels the noise in deviations from a fitted spiral*

Description

Cancels the noise in **vertical or radial deviations** from a fitted spiral after finding the best smooth value for the outline

Usage

```
doCancelNoise(x = x, y = y, spiral.output = spiral.output,
              vertical = TRUE, plot.it = TRUE, gtitle = "")
```

Arguments

`x`	x coordinate of outline
`y`	y coordinate of outline
`spiral.output`	spiral.output the output from a spiral fit
`vertical`	should vertical deviations (if available) be used (default TRUE)
`plot.it`	plot.it if TRUE then plot the noise cancelled deviations
`gtitle`	gtitle title for the graphics window (default blank)

Details

Applies a smooth to the (x,y) outline using a smooth factor (spar) so that the std. dev. of spiral deviations will be similar to those from the smooth. The std. dev. is based on the square of successive differences (mssd). The deviations from smooth are then subtracted from the spiral fit. A check is made if the outline was flipped (X axis) for spiral fitting with a positive spiral angle. If flipped then the vertical deviations from smooth are multiplied by -1.

Value

x	original x coordinate
y	original y coordiate of outline
arc	arc length
spiral.dev	vertical deviations from spiral fit
smooth.dev	vertical deviations from the smooth fit
no.noise	noise cancelled spiral deviations

Warning

If output not looking as being noise cancelled, then maybe sign error in deviations. Use vertical = FALSE if any fitting other than linear (L1 to L4)

Note

Assumes outline starts from the umbo. You need to manually check that this is so by plotting the outline and first point. To change order use the `rev()` function and check outline before any use of spiral fitting.

Only linear spiral fitting (L1 to L4) outputs vertical deviations. Otherwise use the function `doCompareOutlines` on actual and fitted outlines.

Author(s)

A.E. Aldridge

References

Bissell, A.F. and Williamson (1998) Successive difference tests - theory and interpretation. Journal of Applied Statistics 15(3) 305-323

See Also

`bestSmooth fitAnyLinear arcSmooth2D`

Examples

```
## Not run:
require(MASS)
doSetupGraphics()
  xy.df <- generateAnyLinear( k1=30, k2=30, k3=40,k4=40, noise=0.1)
xx <- xy.df$x
yy <- xy.df$y

## no mouse location and no plotting of results.
xspiral <- fitAnyLinear(xx, yy, axis.start=c(110, 100))
#
## if no axis.start YOU will be prompted to guess axis location on graphics device 2 (left mouse click)
 xspiral <- fitAnyLinear(xx, yy)
xspiral.smooth <- doCancelNoise( xx,yy, xspiral, plot.it=T,
gtitle="Deviations from single spiral fit to two spirals")

## End(Not run)
```

`doCompareOutlines` *sequental deviations between two outlines*

Description

Computes deviations as perpendicular distances between two outlines with equal or matched points.

Usage

```
doCompareOutlines(a.df, b.df, plot.deviations = TRUE, print.area = TRUE)
```

Arguments

`a.df`	reference data frame with columns named as x and y coordinates
`b.df`	data frame with columns named as x and y coordinates
`plot.deviations`	should deviations be plotted (default is TRUE)
`print.area`	should the total area under deviations be printed (default is TRUE)

Details

Both outlines should have the same number of coordinates (x,y) and preferably equally spaced.

Area between outlines is the area under deviations when plotted against arc length (uses Trapezoid Rule of integration).

Deviations are signed to be positive when beyond the reference outline (a.df). The sign also depends on location of the origin so do check that the sign makes sense, if not multiply deviations by -1.

Value

`x1`	x coordinate of data frame a.df
`y1`	y coordinate of data frame a.df
`x2`	x coordinate of data frame b.df
`y2`	y coordinate of data frame b.df
`arc.length`	arc length from start of outline in a.df
`deviation`	perpendicular distance from outline a to outline b

Note

Think of *a.df* has having the raw outline and *b.df* with the fitted or smooth curve this outline. In spiral fitting, deviations are the perpendicular or vertical nearest distance from the outline to the fit. Area is a measure of 'not fitting' or error between the fit (or smooth) and the raw outline.

This function is also useful when using ordinary Procrustes to align and match outlines. See function `procOPA` in the `shapes` package. Deviations between outlines can be plotted with increasing arc length.

Author(s)

A.E. Aldridge

References

Maxwell E.A. (1958) Elementary Coordinate Geometry (page 81)

Ian L. Dryden (2018). shapes: Statistical Shape Analysis. R package version 1.2.4. https://CRAN.R-project.org/package=shapes

See Also

equalSpace arcSmooth2D bestSmooth

Examples

```
## Not run:
require(MASS)
 library(logspiral)
require(MASS)
doSetupGraphics()
# generate a L2 spiral with change at 100th coordinate pair
# and a difference of 3 degrees in spiral angle. Fit a L1 spiral
# then extract fitted coordinates as a data frame.
#
xy.df <- generateAnyLinear( k1=40, k2=40, k3=43,k4=43, noise=0.1)
xys.fit <- fitAnyLinear( x=xy.df$x, y=xy.df$y)
xyfit.coords <- data.frame(x=xys.fit$xpred.orig, y=xys.fit$ypred.orig)
xcompare <-  doCompareOutlines( a.df=x.df, b.df=xyfit.coords)
##
# Fit an L2 spiral and extract fitted coordinates, and compare
#
 xys2.fit <- fitAnyLinear( x=xy.df$x, y=xy.df$y,m=100)
 xyfit2.coords <- data.frame(x=xys2.fit$xpred.orig, y=xys2.fit$ypred.orig)
 xy2compare <-  doCompareOutlines( a.df=xy.df, b.df=xyfit2.coords)
##
# Fit a smooth to the outline and extract fitted coordinates, and compare
#
 xysmooth <- arcSmooth2D( x=xy.df$x, y=xy.df$y)
 xysm.coords <- data.frame( x=xysmooth$xp, y=xysmooth$yp)
 xsm.compare <-  doCompareOutlines( a.df=xy.df, b.df=xysm.coords)
#
## can work well on any pairs of outline shapes

## End(Not run)
```

---

doLocateChange *Interactively locate a sequence of changes in spiral deviations*

Description

plots deviations and allows you to interactively locate changes such as maxima and minima

Usage

```
doLocateChange(spiral.output=spiral.output, nc=NULL, eps.factor=1)
```

Arguments

spiral.output	output from either spiral fitting or noise cancelling deviations
nc	number of changes to locate on the deviation plot
eps.factor	multiplier of the average distance between outline points

Details

User specified changes on a deviation plot. The plot is arc length and vertical deviation. The locator function is used to interact with the plot. If the number of changes is not given the function is terminated. By default deviations are plotted with vertical lines for specified change locations (can be extrema, or any feature of the deviations).

Value

locate.change	sequence of change locations as coordinates (x,y)

Author(s)

A.E. Aldridge

References

see minima and maxima in Clark et al (2015) pages 36-38

Clark J., et al (2015) Application of shell spiral deviations methodology to fossil brachiopods. Palaeontologica Electronica 18.3.54A:1-39

Examples

```
## Not run:
data(exGiantSquidBeak)
x.df <- exGiantSquidBeak
x <- x.df$x ;  y <- x.df$y
MASS::eqscplot(x,y)
zspiral <- fitAnyLinear( x,y, plot.deviations=TRUE)
zchanges <- doLocateChange( zspiral, nc=2)
zchanges
#
znoise.cancel <- doCancelNoise( x,y,spiral.output=zspiral)
zzchanges <- doLocateChange( znoise.cancel, nc=2)
zzchanges

## End(Not run)
```

---

doSetupGraphics *set up graphics windows*

Description

set up FOUR graphic windows numbered 2, 3, 4 and 5

Usage

```
doSetupGraphics( high=4, wide = 4, pt.size = 8)
```

Arguments

high	height of window in points
wide	width of window in points
pt.size	number of points per unit of screen display

Details

Graphic windows are located assuming a wide screen with only the R command window on the left. Height and width and point size can be altered. Subsequent manual resizing of windows can distort placement of graphics.

Value

Four graphic windows with the last one the active window.

Note

Need to manually copy and edit function if changing location or size of graphic windows. Create your own graphics function (e.g. `mySetupGraphics`) to set up four graphic windows for your own display device.

Author(s)

A.E. Aldridge

See Also

`dev.set dev.list dev.next dev.new` in base R.

Examples

```
##doSetupGraphics()
## return to the first graphics window
dev.set(2)

## The function is currently defined as
function (high = 4, wide = 4, pt.size = 8)
{
    windows(height = high, width = wide, pointsize = pt.size,
        xpos = -25, ypos = 0)
    windows(height = high, width = wide, pointsize = pt.size,
        xpos = -450, ypos = 0)
    windows(height = high, width = wide, pointsize = pt.size,
        xpos = -25, ypos = -30)
    windows(height = high, width = wide, pointsize = pt.size,
        xpos = -450, ypos = -30)
    invisible()
  }
```

doSpectrum *plot the Fourier sums of squares against wave number (frequency)*

Description

Uses Fast Fourier Transform (fft) to estimate the sums of squares of each Fourier frequency from zero to N/2 (N = series length).

Usage

```
doSpectrum(Series, W=1, wave1 = NULL, wave2 = NULL, Order = 2, Title = NA, output = FALSE)
```

Arguments

Series	series of equally spaced values in space (eg arc length).
W	smoothing value, or the number of periodogram ordinates being averaged (W=1 no smoothing).
wave1	minimum Fourier frequency (wave number) to plot
wave2	maximum Fourier frequency (wave number) to plot
Order	order of autoregressive (AR) model's spectrum to compare with actual spectrum
Title	title for plotting
output	output the spectrum (default is no output)

Details

Equal spacing of values is assumed. First frequency (wave1=0) is the variation about the mean of the series, which should be zero. Apparent periodicity in a series can be generated by an AR process so pays to check if this is likely. A very approximate confidence line is included to judge the significance (or not) of any peaks in a spectrum produced. Confidence limits are based on spectrum terms being exponentially distributed (dashed lines).

Variance predicted for an AR process (default order is 2) is the solid (blue) line.

Value

output	sums of squares (raw if W=1, smooth if W>1) at each wave number

Note

Adapted from RB Davies Splus code. Any mistakes are the fault of logspiral package author.

Author(s)

A.E. Aldridge

References

Thanks to Dr Robert Davies (Statistics Research Associates Ltd) kindly supplied most of the code as an S-plus function, which the author ported to R.

See Also

plotSpiralDeviations partitionSumSquares

Examples

```
## Not run:
z <- generateAnyLinear( waves=6 )
xs <- fitAnyLinear(x=z$x, y=z$y)
doSpectrum( Series=xs$deviations, W=1, wave1=0, wave2=15, Title="L1 spiral with six cycles")

## End(Not run)
```

---

doVariogram *produces a semi-variogram to highlight trends or cycles in a series*

Description

Estimates variation based on increasing differencs (lags) between values.

Usage

```
doVariogram(Series, nlags = NULL, plot.it = T, output = F, Title = NA)
```

Arguments

Series	N equally spaced values
nlags	number of lags to compute (maximum is N/2)
plot.it	plot the semi-variogram (default)
output	output the results (default is no output)
Title	title for the semi-variogram

Details

Variation as standard deviation and variance is estimated for each lag (from 1 to N/2). The standard deviation is plotted against increasing lag. If output is TRUE then lag number standard deviation, variance, and number of values for each lag are combined into a data frame. As lag number increases there are fewer values for which to estimate variation.

Warning: assumes no missing values

Value

output	data frame with: lag, standard deviation, variance, number of values

Note

Code for R adapted from Visual Basic code, see Table D.1 of Smith (2001)

Author(s)

A.E. Aldridge

References

Smith P.L. 2001. A primer for sampling solids, liquids and gases. SIAM, ASA.

See Also

plotSpiralDeviations doSpectrum doLocateChange

Examples

```
## Not run:
z <- generateAnyLinear()
xs <- fitAnyLinear(x=z$x, y=z$y, waves=6)
doVariogram( Series=xs$deviations, Title="L1 with six cycles" )
# should see three peaks and troughs up to N/2 lags

## End(Not run)
```

---

equalSpace *equal space outline coordinates using arc length*

Description

equal spacing along an outline coordinates using arc length

Usage

```
equalSpace(x, y, nequal = 50, light.smooth = FALSE)
```

Arguments

x	x coord
y	y coord
nequal	number of equally spaced points required
light.smooth	light smooth (default is none)

Details

Uses linear interpolation. Do check the effect on an outline by plotting a few pointsof the actual and different numbers of equal spaced coordinates. Double or quadruple the actual number of digitised points often works well when NO zig-zags in the growth outline.

Value

x,y	coordinates of equally spaced points using arc length
arc.length	arc length along the sequence of output x, y values

Note

If very few points then nequal needs to match the input with an even,or odd total points

Author(s)

A.E. Aldridge using code from Ramsay & Silverman (2002) online resources.

References

Ramsay and Silverman (2002) Applied Functional Data Analysis

Section 8.2. Analysing shapes without landmarks (notchproc2.fun)

http://www.stats.ox.ac.uk/~silverma/fdacasebook/ (accessed 24 Sept. 2019)

Examples

```
## Not run:
x <- c(1,4,6,7,8,9)
y <- c(2.3, 4.5, 2, 6.5, 8.5,5.5)
#
MASS::eqscplot(x,y,type="b")
same.pts <- equalSpace(x,y,nequal=6)
same.pts.smooth <- equalSpace(x,y,nequal=6,light.smooth=TRUE)
dense.pts <- equalSpace(x,y, nequal=14)
#
MASS::eqscplot(x,y,type="b")
points( same.pts$x, same.pts$y, type="b",pch=16)
eqscplot(x,y,type="b")
points( same.pts.smooth$x, same.pts.smooth$y, type="b",pch=17)
#
MASS::eqscplot(x,y,type="b")
points( dense.pts$x, dense.pts$y, type="b",pch=15)

## End(Not run)
```

---

exArctica Arctica islandica *shell outline*

Description

Two dimensional outline of the ocean quahog *Arctica islandica* valve

Usage

```
data(exArctica)
```

Format

A data frame with 1xx observations on the following 2 variables.

`x` a numeric vector

`y` a numeric vector

`z` a numeric vector

Details

More than one spiral is best fit. Jones (1983) finds age of 12 years for this specimen.

Source

Figure 3 in Jones (1983)

References

Jones, D. 1983. Sclerochronology: Reading the Record of the Molluscan Shell. American Scientist 71(4):384-391

Jones, D. 1981. Repeating layers in the molluscan shell are not always periodic. Journal of Paleontology 55(5) 1076-1082. See Figure 1.

Examples

```
## Not run:
# load the data file
data(exArctica)
x <- exArctica$x ;  y <- exArctica$y
#
MASS::eqscplot(x,y)
zzz <- fitAnyLinear(x,y, plot.deviations=TRUE)

## End(Not run)
```

---

exColossalSquidBeak *Upper beak of a Colossal Squid*

Description

Upper beak rostrum of a Colossal Squid (*Mesonychoteuthis hamiltoni*)

Usage

```
data("exColossalSquidBeak")
```

Format

A data frame with 392 observations on the following 2 variables.

`x` a numeric vector

`y` a numeric vector

Details

Outline is from beak (rostrum) to hood, digitised to 42 pixels per mm. A transition or bent cable spiral is best fit. Deviations indicate an age of at least 7 years.

Source

AE Aldridge, from Te Papa collection specimen M.316689, more information in Remeslo et al (2019), page 122.

References

Aldridge (2009) Can beak shape help to research the life history of squid? New Zealand Journal of Marine and Freshwater Research 43 1061-1067

Remeslo et al (2019). Distribution and biology of the colossal squid, *Mesonychoteuthis hamiltoni*: New data from depredation in toothfish fisheries and sperm whale stomach contents.

Deep-Sea Research Part I 147, 121-127.

Examples

```
 library(logspiral)
 library(MASS)  # install if not already available
 library(wmtsa) # install if not already available
doSetupGraphics() # applies only to the Windows platform
                  # otherwise type dev.new() four times
#
xy.df <- exColossalSquidBeak # copy of the 392 outline coordinates
#
csbeak.df <- equalSpace( x=xy.df$x, y=xy.df$y, nequal=600)
#
xx <- csbeak.df$x; yy <- csbeak.df$y # the beak outline coordinates
#
set.seed(6543) # to reproduce the results in Aldridge (20??)
#
L1features <- computeL1DeviationFeatures(x=xx, y=yy, plot.it=FALSE)
#
# you need to click on graph to locate an initial spiral axis.
# Note that the start of the outline has a red solid symbol.
# After computing the L1 features the outline will have a
# blue cross for the initial axis and a red, closed cross for the estimated L1 axis
# Note: loads wavelet package wmtsa (install if not already available)

csbeakSpirals <- compareFeaturesToCatalogue(spiral.features= L1features,
                  plot.detail = FALSE,output.results=TRUE)

# L1 spiral fitted as the object csbeak.L1
#
csbeak.L1 <- fitAnyLinear( x=xx,y=yy,
             axis.start=c(141,21), plot.deviations=TRUE )
#
## CAN SKIP THE CHECKING OF CHANGES and proceed
## direct to fitting the remaining spirals below
#####################################################
# First change(L2 spiral)
checkL2 <- fitCheckL2Spiral( x=xx, y=yy,
                            axis.start=c(142, 21))
#
# progresses through outline (may take some time)
# finds change point at coordinate 443 with min error 0.111084
#
# Second change (L3 spiral)
# see if there is a second change location along outline
# takes some time, especially slow 1/3 through outline
#
checkL3 <- fitCheckL3Spiral( x=xx, y=yy, m=443,
                            axis.start=c(142,21) )
```

```
#
# finds second change at coordinate 87 with min error 0.1003496
#
# Third change (L4 spiral)  is there a third possible change
#
checkL4 <- fitCheckL4Spiral( x=xx, y=yy, m=c(87,443),
                            axis.start=c(142,21) )
#
# finds a third minima at 371, but the fitting fails
# in part of the outline (approx 150 to 250 coordinates)
# minimum error now 0.09393627
#########################################################
## FITIING ALL SPIRALS suggested from catalogue ###
#
csbeak.L2 <- fitAnyLinear( x=xx,y=yy,m=c(443),
         axis.start=c(141,21),plot.deviations=TRUE )
csbeak.L3 <- fitAnyLinear( x=xx,y=yy,m=c(87, 443),
             axis.start=c(141,21),plot.deviations=TRUE )
csbeak.L4 <- fitAnyLinear( x=xx,y=yy,m=c(87,371, 443),
             axis.start=c(141,21),plot.deviations=TRUE )
#
# Transition spirals, LQL and L2QL and their nonquadratic
# options as objects gLQL and gL2QL
#
csbeak.LQL <- fitBentCable( x=xx, y=yy, m=443,
            axis.start=c(141,21),trace.check=F,
    start.values=c(a=10, k1=29, k2=25, g=0.05, kappa=2),
            quadratic=T, plot.deviations=T)
#
csbeak.gLQL <- fitBentCable( x=xx, y=yy, m=443,
             axis.start=c(144,11),trace.check=F,
    start.values=c(a=10, k1=29, k2=25, g=0.05, kappa=2),
            quadratic=F, plot.deviations=T,
             title.label="Generalised Bent Cable")
#
csbeak.L2QL <- fitLinear2BentCable( x=xx, y=yy, m=c(87,443),
               axis.start=c(144,11),trace.check=F,
     start.values=c(a0=10, k0=32, k1=29, k2=25, g=0.05, kappa=2),
               quadratic=T, plot.deviations=T)
#
csbeak.gL2QL <- fitLinear2BentCable( x=xx, y=yy, m=c(87,443),
                axis.start=c(144,11),trace.check=F,
     start.values=c(a0=10, k0=32, k1=29, k2=25, g=0.05, kappa=2),
                quadratic=F, plot.deviations=T,
               title.label="Generalised Linear to Bent Cable")
#
# Choice of best fitting spiral is the L2QL fit
#
# Now a Wavelet decomposition of L2QL spiral deviations
#
csbeak.wavelets <- computeDeviationWavelets( spiral.output=csbeak.L2QL,
                   wave.levels=9 )
#
# plots the wavelet with maximum variance (level 5 on power of 2)
# can plot each wavelet of the decomposition, e.g. level 6
#
plot( csbeak.L2QL$arc.length, csbeak.L2QL$deviations, type="s")
```

```
lines( csbeak.L2QL$arc.length, csbeak.wavelets$V6,col="blue",lwd=2)
```

---

exGiantSquidBeak *Upper beak of a Giant Squid*

Description

Upper beak of a Giant Squid (*Architeuthis dux*)

Usage

```
data(exGiantSquidBeak)
```

Format

A data frame with 154 observations on the following 2 variables.

`x` a numeric vector

`y` a numeric vector

Details

Outline is from beak tip (rostrum) to hood. More than one spiral is best fit. Deviations indicate an age of 6 years.

Source

Tony Aldridge, from NIWA (Darren Stevens) collection specimen 02/00 labelled as "v.immature". Aldridge label as NIWA04 (out of 11 photographed).

NIWA: National Institute of Water and Atmospheric Research (NZ)

References

Aldridge (2009) Can beak shape help to research the life history of squid? New Zealand Journal of Marine and Freshwater Research 43 1061-1067

Examples

```
## Not run:
data(exGiantSquidBeak)
x.df <- exGiantSquidBeak
x <- x.df$x ;  y <- x.df$y
require(MASS)
eqscplot(x,y)
## fit single spiral.  Note that both four spirals (L4), and
## the L2BentCable highlight the underlying periodicity.
zzz <- fitAnyLinear( x,y)

## End(Not run)
```

exGryphaea *Lower valve in a Gryphaea*

Description

Shell growth line in *Gryphaea arcuata*

Usage

```
data(exGryphaea)
```

Format

A data frame with 116 observations on the following 2 variables.

`x` a numeric vector

`y` a numeric vector

Details

More than one spiral is best fit. This is the raw digitised data, yet to be equally spaced. Common name for this species is Devil's Toenail and is a "mollusc emulating a brachiopod".

Source

Figure 6, page 171. Lower image. of Jones & Gould (1999)

References

Direct Measurement of Age in Fossil Gryphaea: The Solution to a Classic Problem in Heterochrony Douglas S. Jones and Stephen Jay Gould Paleobiology Vol. 25, No. 2 (Spring, 1999), pp. 158-187

Examples

```
## Not run:
data(exGryphaea)
x <- exGryphaea$x ;  y <- exGryphaea$y
require(MASS)
eqscplot(x,y)
## fit a single spiral and note the deviation shape is
## classic for a two spiral fit.
zzz <- fitAnyLinear( x,y, plot.deviations=T, plot.fit=T)

## End(Not run)
```

exLaqueus *Laqueus rubellus*

Description

Three brachiopod specimens each with dorsal and ventral valve

Usage

```
data(exLaqueus)
```

Format

A data frame with 1318 observations on the following 4 variables.

`specimen` a factor with levels `F1` `F3` `FR`

`valve` a factor with levels `dorsal` `ventral`

`x` a numeric vector

`y` a numeric vector

Details

See References for which are juvenile and adult brachiopods. Easier to apply spiral fitting if you first extract a specimen and a valve from the data frame.

Source

Professor Alberto Perez-Huerta (University of Alabama)

References

Perez-Huerta A., et al (2014) Brachiopod shell spiral deviations (SSD): implications for trace element proxies. Chemical Geology 374-375, 13-24.

Examples

```
## Not run:
data(exLaqueus)
x.df <- exLaqueus
 table(x.df$specimen, x.df$valve)
x1.df <- x.df[x.df$specimen=="F1", ]
x <- x1.df$x[x1.df$valve=="ventral"]
y <- x1.df$y[x1.df$valve=="ventral"]
require(MASS)
eqscplot(x,y)
eqscplot( x1.df$x, x1.df$y,pch=16, cex=0.7)
xs <- fitAnyLinear( x=x, y=y, plot.deviations=T)

## End(Not run)
```

exMagnirostris *Galapagos finch beak : Geospiza magnirostris*

Description

Bird upper and lower beak of one specimen.

Usage

```
data(exMagnirostris)
```

Format

A data frame with 75 observations on the following 3 variables.

`x` a numeric vector

`y` a numeric vector

`side` a character vector

Details

Only the upper, dorsal surface of the beak has enough curvature and points suitable for a spiral fit. Source image was from a paper reprint so there was not the resolution for detail digitising. All specimens in Figure 5 of Grant (2001) were amenable to spiral fitting. Grant (2001), Grant & Grant (2008) use beak length, width and thickness (not spiral angles) to distinguish finch species.

Source

Figure 5 of Grant(2001)

References

Grant, P. (2001) Reconstructing the evolution of birds on islands : 100 years of research. Oikos 92: 385-403.

Grant, P. & Grant, R. (2008) How and why species multiply. Princeton. 218p.

Examples

```
## Not run:
doSetupGraphics()
z.df <- exMagnirostris
## extract the dorsal or upper beak.
z.df <- z.df[ z.df$side=="dorsal", ]
x <- z.df$x
y <- z.df$y
MASS::eqscplot(x,y)
## fit one spiral.  Note that the outline is flipped for fitting.

gm.spiral <- fitAnyLinear( x, y, plot.deviations=TRUE, plot.fit=T)

## End(Not run)
```

exNautilisShell *Nautilis shell section*

Description

Shell section of a *Nautilis*

Usage

```
data(exNautilisShell)
```

Format

A data frame with 262 observations on the following 2 variables.

`x` a numeric vector

`y` a numeric vector

Details

An example of fitting **several whorls of accretionary growth**.

More than one spiral is best fit. Not equally spaced coordinates and you may need to double using the function `equalSpace`

Source

Image from Dr. Mark Norman (~2003) Museum Victoria, Melbourne, Australia

References

Mark Norman: Museum Victoria, Melbourne, Australia

Examples

```
## Not run:
data(exNautilisShell)
x <- exNautilisShell$x ;  y <- exNautilisShell$y
#
MASS::eqscplot(x,y)
zzz <- fitAnyLinear( x,y,plot.deviations=T, plot.fit=T)

## End(Not run)
```

exRafinesquina *Rafinesquina sp* outline of a shell outline

Description

Two dimensional, sagittal section of a strophomenid Ordovician brachiopod, *Rafinesquina*

Usage

```
data("exRafinesquina")
```

Format

A data frame with 150 observations on the following 2 variables.

`x` a numeric vector

`y` a numeric vector

Details

Example of applying two very different spirals to an outline. Clark et al (2015): "impossible to fit a spiral to this valve", but here we have two spirals that fit well with sprial deviations that can be extracted (see Example code below).

Source

Figure 8 (left), page 11 of Clark et al (2015).

References

Clark J., et al (2015) Application of shell spiral deviations methodology to fossil brachiopods. Palaeontologica Electronica 18.3.54A:1-39

Examples

```
## Not run:
data(exRafinesquina)
x.df <- exRafinesquina
# First part is a L1 spiral
xs1a <- fitAnyLinear( x = x.df$x[1:36], y = x.df$y[1:36], plot.fit=T, plot.deviations=T)
# extract coordinates for second part of the outline
 xx <- x.df$x[-c(1:36)]
 yy <- x.df$y[-c(1:36)]
# fit spiral to second part
xsq <- fitQuartic( xx,yy, plot.fit=T, quadratic=0.2, cubic=0.07,
                   axis.start=c(8, 11.4), plot.deviations=T)
# plot fitted spirals and the digitised outlines
 eqscplot( x.df$x, x.df$y, xlab="", ylab="");axis(4,labels=F);axis(3,labels=F)
 lines(xs1a$xpred.orig, xs1a$ypred.orig, col=2, lwd=2)
 lines(xsq$xpred.orig, xsq$ypred.orig, col=2, lwd=2)
 title( "Rafinesquina sp: outline and fitted spirals", adj=0)
#
# plot concatenated deviations from both spirals
```

```
 plot( c(xs1a$arc.length, xsq$arc.length+max(xs1a$arc.length)),
            c(xs1a$deviations, xsq$deviations), type="s",
       xlab="Arc length (mm)", ylab="Deviation from fitted spirals (mm)")
 abline(v=max(xs1a$arc.length), col=2)
 abline(h=0, col=4)
title( "Rafinesquina sp: spiral deviations", adj=0)

## End(Not run)
```

---

exSpirulaBeak *Rams horn shell: Spirula spirula*

Description

beak of squid *Spirula spirula*

Usage

```
data(exSpirulaBeak)
```

Format

A data frame with xx observations on the following 3 variables.

`x` a numeric vector

`y` a numeric vector

`rostrum` a character vector

`side` a character vector

Details

This beak from same specimen of *Spirula* in the file **exSpirulaShell**

Source

Image from Dr. Mark Norman (Dec. 2003) Museum Victoria, Melbourne, Australia

References

http://tolweb.org/Spirula_spirula/19989 (accessed 20 Nov. 2019)

Examples

```
## Not run:
## create some graphic windows ( 2 to 5)
aaSetupGraphics()
## load the data
data(exSpirulaBeak)
## extract the right outer (upper) beak and plot
z.df <- exSpirulaBeak
zright.df <- z.df[z.df$rostrum=="outer" & z.df$side=="right", ]
x <- zright.df$x
y <- zright.df$y
```

```
MASS::eqscplot(x,y)
## fit a single spiral
zspiral <- fitAnyLinear( x,y, plot.deviations=T, plot.fit=T)
## now see if there is a spiral change
zchk <- fitCheckL2Spiral( x,y, plot.detail=F)
## fit at the change point
zspiral <- fitAnyLinear( x,y, m=c(88), plot.deviations=T, plot.fit=T)
## noise cancel the deviations
znoise.cancel <- NoiseCancelDeviations( x,y, zspiral)

## End(Not run)
```

`exSpirulaShell` Rams horn coiled shell: *Spirula spirula*

Description

Shell of *Spirula spirula*

Usage

```
data(exSpirulaShell)
```

Format

A data frame with 962 observations on the following 2 variables.

`x` a numeric vector

`y` a numeric vector

Details

An example of fitting **several whorls** of accretionary growth.

Same specimen as for the *S. spirula* beak data. More than one spiral is best fit. Chambers show up in the deviation plot, especially if `doCancelNoise` is used on the spiral fit.

Source

Image from Dr. Mark Norman (Dec. 2003) Museum Victoria, Melbourne, Australia

References

http://tolweb.org/Spirula_spirula/19989 (accessed 20 Nov. 2019)

Examples

```
## Not run:
data(exSpirulaShell)
x <- exSpirulaShell$x ;  y <- exSpirulaShell$y
MASS::eqscplot(x,y)
## fit a single spiral WITHOUT the first ten points
zzz <- fitAnyLinear( x[-c(1:10)],y[-c(1:10)],
plot.deviations=T, plot.fit=T)
xnonoise <- NoiseCancelDeviations(spiral.output=zzz)

## End(Not run)
```

---

exTerebratula *Terebratula* fossil section

Description

dorsal and ventral valves as digitised, without equal spacing

Usage

```
data(exTerebratula)
```

Format

A data frame with 709 observations on the following 3 variables.

`valve` a character vector

`x` a numeric vector

`y` a numeric vector

Details

A particulary difficult outline because anterior is flattened. Best to remove part of the outline before fitting spirals.

Source

Professor Alberto Perez-Huerta (University of Alabama)

References

Clark J., et al (2015) Application of shell spiral deviations methodology to fossil brachiopods. Palaeontologica Electronica 18.3.54A:1-39

Examples

```
## Not run:
data(exTerebratula)
z.df <- exTerebratula
x <- z.df$x[z.df$valve=="ventral"]
y <- z.df$y[z.df$valve=="ventral"]
#
dev.new()
MASS::eqscplot(x,y)
dev.new()
MASS::eqscplot( z.df$x, z.df$y)

## End(Not run)
```

`fitAnyLinear` *fit up to four connected or piecewise logarithmic spirals.*

Description

fit up to four connected, piecewise linear spirals: L1, L2, L3 and L4.

Usage

```
fitAnyLinear(x, y, m = c(NULL, NULL, NULL), axis.start = c(NULL, NULL),
  specimen = "", res1 = NULL, res2 = NULL, tolerance.step = 0.01, flip = FALSE,
  plot.fit = FALSE, plot.deviations = FALSE, plot.diagnostics = FALSE,
  print.input = TRUE, orient.check=TRUE)
```

Arguments

`x`	x coord
`y`	y coord
`m`	change points for spirals (max three)
`axis.start`	axis start coordinates x, y
`specimen`	specimen name
`res1`	lower limit for deviation plot
`res2`	upper limit for deviation plot
`tolerance.step`	tolerance step to nls
`flip`	flip the outline if needed
`plot.fit`	plot fitted spiral and outline (default is none)
`plot.deviations`	plot spiral deviations (default is none)
`plot.diagnostics`	plot diagnostic plot (default is none)
`print.input`	plot the input to the function, especially change points
`orient.check`	check the orientation of the outline (default is to check)

Details

Assumes outline is in a anticlockwise direction. If orient.check is TRUE (default), then test for outline direction using first ten points. If clockwise then the outline is flipped about the Y (vertical) axis for computations. Values of fitted outline are flipped back to original orientation of outline (see Values).

The first outline point is assigned an angle of zero plus a tiny angle (eps =0.00001, see function `cumulativeAngle`).

Spiral parameter k is in radian units. Convert to degrees by `(180/pi)*atan(1/k)`

Parameters are estimated using `nls` with the default Gauss-Newton algorithm. The number of change points (m) determines which spiral is fitted. Intitial or starting values are embedded inside the nls call for linear spirals. Initial axis location is in the function argument axis.start, if not specified then you are asked to interactively locate the start of the spiral axis.

Convergence works well when starting spiral angles are large (approximately 80 degrees, or 0.2 as radian on the logarithmic scale). If not converging, check the Troubleshooting Guide, or copy & run a fitting function changing nls starting values (useful to have trace set to TRUE).

A spiral deviation is the radial difference between actual and fit from the estimated spiral axis. Vertical deviation is perpendicular distance from actual point to fitted spiral.

Value

`parameters1`	spiral axis location (u,v), and spiral parameters a, k
`parameters2`	standard deviations of fit (radius) and test of randomness
`parameters3`	N of outline points, degrees of freedom, and was outline flipped (1=yes)
`xpredict, ypredict`	are the coordinates for the fitted radius (translated or flipped outline)
`angle`	is the angle (radians) of the actual & fitted point about the spiral axis
`arc.length`	is the arc length from the first point (zero) to the last point on an outline
`deviations`	diference between in radii between actual and fit from the estimated spiral axis
`v.deviations`	the perpendicular distance from estimated spiral to outline coordinate
`fitted`	the fitted radius from estimated spiral axis
`xpred.orig, ypred.orig`	coordinates for the fitted radius in the original coordinate axes

Warning

may not always converge so read Troubleshooting Guide for workarounds

Note

Assumes outline starts from the umbo. You need to manually check that this is so by plotting the outline and first point. To change order use the `rev()` function then and check outline before any use of spiral fitting.

Test of randomness is the mean square successive difference (mssd) test. Unity is random, greater than 1 is evidence of sawtooth, and less than one for less frequent oscillations.

Author(s)

A.E. Aldridge

References

Aldridge (1999) Brachiopod outline and episodic growth. Paleobiology 25, 471-482

Bissell & Williamson (1998) Successive difference tests - theory and interpretation Journal of Applied Statistics 15(3) 305-323

See Also

fitL1Spiral fitL2Spiral fitL3Spiral fitL4Spiral

fitCheckL2Spiral fitCheckL3Spiral

fitCheckL3Restricted fitCheckL4Spiral

Examples

```
## Not run:

doSetupGraphics()
set.seed(1998) # set the random number generator so results can be reproduced.
#
# generate single linear logarithmic spiral (L1) using all defaults, including noise
#
 z <- generateLinearSpirals()
#
# fit this L1 spiral, specifying the initial guess of axis location
#
 xs <- fitAnyLinear(x = z$x, y = z$y, axis.start=c(160,112), plot.deviations=T, plot.fit=T)
#
# print to the console the spiral parameters and summary the fit from the fitted object
xs[1]  # for the parameters, you can also use xs$parameters1
xs[2]  # for the measures about the fit itself, you can also use xs$parameters2
#
# fit a single linear spiral, where initial axis location is interactive in a graphics window
## you will be prompted to guess axis location on graphics device 2 (right mouse click)
# a red symbol is the first point of the outline - this should be at the spiral start, NOT the end
#
 xs <- fitAnyLinear(x = z$x, y = z$y)
#
## See Vignette on Linear Spirals for two, three and four connected spirals


## End(Not run)
```

---

fitBentCable *Fits a generalised bent cable spiral to an outline of x,y coordinates*

Description

Fits generalised bent cable spiral to given outline coordinates. Three phases of growth: linear then a curved transition to the final linear phase.

Usage

```
fitBentCable(x, y, m = NULL, axis.start = c(NULL, NULL), quadratic = TRUE,
          start.values = c(a = 2, k1 = 36, k2 = 46, g = 0.3, kappa = 2.0),
         angle.deg = TRUE, plot.fit = FALSE, plot.deviations = FALSE,
         plot.diagnostics = FALSE, flip.x = FALSE, move.by = NULL,
         tolerance.step = 0.1, trace.check = FALSE,
         title.label = "Bent Cable outline")
```

Arguments

x	x coordinate
y	y coordinate
m	location coordinate for the transition (not necessarily central)
axis.start	start location of the spiral axis as a (x,y) coordinate
quadratic	should a quadratic bend be forced (default no quadratic)
start.values	initial values for other unknown values in the strict order: distance axis to to start of spiral (a1), two spiral angles (k1, k2), and shape of the transition (g, kappa)
angle.deg	units of spiral angle as start values and output (default FALSE)
plot.fit	plot the outline with fitted values (default FALSE)
plot.deviations	plot the radial deviations from spiral fit (default FALSE)
plot.diagnostics	plot diagnostics for deviations from fitted spiral (see Detail below)
flip.x	do we flip the outline to ensure positive spiral angle values
move.by	do we move (translate) the outline to ensure all coordinates are postive
tolerance.step	tolerance step for nls used in the fitting (minscale not specified)
trace.check	do we wish to trace the steps of the nls algorithm (default FALSE)
title.label	text for the title of the deviation plot (default none)

Details

Generalised bent cable of Khan & Kar (2018) is applied to the outline, which is converted to polar coordinates based on an initial axis location (two values input to the axis.start function argument).

If quadratic is TRUE then kappa is set at 2.0 for the bent cable of Chiu et al (2006). See Khan & Kar (2018) page 1803.

The nls function is applied to a residual sums of squares to estimate six unknown parameters (u, v, a1, k1, k2, g and kappa). The central location of transition must be supplied as the mth coordinate, which is the paramter tau in the generalised bent cable. Tau is not estimated, but is an output value as an angle (radian units) along the outline.

Value

parameters1	spiral axis location (u,v), and spiral parameters a1, k1,a2,k2,g,kappa,tau
parameters2	standard deviations of fit (radius) and test of randomness
parameters3	N of outline points, degrees of freedom, and was outline flipped
parameters4	start and end of transition: angle, coord no. and arc distance

`xpredict, ypredict`
: are the coordinates for the fitted radius (translated or flipped outline)

`angle`
: is the angle (radians) of the actual & fitted point about the spiral axis

`arc.length`
: is the arc length from the first point (zero) to the last point on an outline

`deviations`
: diference between in radii between actual and fit from the estimated spiral axis

`fitted`
: the fitted radius from estimated spiral axis

`xpred.orig, ypred.orig`
: coordinates for the fitted radius in the original coordinate axes

Warning

start or intial (esp. axis) values need to be close to actual so that nls converges

Note

g is the gamma in Chiu (2006) and Khan & Kar (2018). Parameters a2, a3 are estimated by direct calculation.

Bayesian methods using JAGS and STAN cannot be applied because the independent variables (axis locations) are updated at every iteration. Other spiral fitting functions can be applied to help find suitable starting values. For example, a **fitL1Spiral** to the beginning of the outline locates spiral axis, initial intercept, and slope. The functions **fitL2Check**, and a **fitL3Spiral** can help locate starting values for the transition location,and the slope or spiral angles.

Author(s)

A.E. Aldridge

References

Chiu, G., Lockhart, R. and Routledge, R. (2006), Bent-Cable Regression Theory and Applications, Journal of the American Statistical Association, 101, 542-553. DOI: 10.1198/016214505000001177 http://faculty.washington.edu/gchiu/Articles/bentcable-jasa.pdf

Khan, S.A. and Kar, S.C. (2018). Generalized bent-cable methodology for changepoint data: a Bayesian approach. Journal of Applied Statistics, 45(10), 1799-1822

See Also

`fitLinear2BentCable`. Have quadratic shape to the transition (i.e. kappa = 2) .

Examples

```
## Not run:
 z <- generateBentCableSpiral()
 xs <- fitBentCable(x = z$x, y=z$y, m=100, axis.start=c(1000,1000) ,
        plot.fit=TRUE, plot.deviations=TRUE)
# assumes four graphics windows are open (fit on device 3, and deviations on device 4)

## End(Not run)
```

`fitBentCable2Linear` *Fits a bent cable to linear spiral (LQL2) on an outline of x,y coordinates*

Description

Fits linear connected to a generalised bent cable spiral on the given outline coordinates. So there are FOUR phases of growth: linear to a quadratic transition then linear again before a final linear phase (ie three linear phases and one curved phase of growth).

Usage

```
fitBentCable2Linear(x, y, m = c(NULL, NULL), axis.start = c(NULL, NULL), quadratic = TRUE,
            start.values = c(a1 = 2, k1 = 36, k2 = 40, k3 = 55, g = 0.3, kappa = 2),
           angle.deg = TRUE, flip.x = FALSE, move.by = NULL, tolerance.step = 0.1,
                trace.check = FALSE, plot.fit = FALSE, plot.deviations = FALSE,
          plot.diagnostics = FALSE, title.label = "fit Bent Cable to Linear outline ")
```

Arguments

`x`	x coordinate
`y`	y coordinate
`m`	location coordinates for change between linear phases and centre of transition
`axis.start`	starting location of spiral axis in (x,y) coordinates
`quadratic`	should a quadratic bend be forced (default no quadratic)
`start.values`	initial values for other unknown values in order of distance axis to to start of first linear spiral (a0), three spiral angles (k0, k1, k2), and the shape of the transition (g, kappa)
`angle.deg`	units of spiral angle as start values and output (default FALSE)
`flip.x`	do we flip the outline to ensure positive spiral angle values
`move.by`	do we move (translate) the outline to ensure all coordinates are postive
`tolerance.step`	tolerance step for nls used in the fitting (minscale not specified)
`trace.check`	do we wish to trace the steps of the nls algorithm (default FALSE)
`plot.fit`	plot the outline with fitted values (default FALSE)
`plot.deviations`	plot the radial deviations from spiral fit (default FALSE)
`plot.diagnostics`	plot diagnostics for deviations from fitted spiral (see Detail below)
`title.label`	text for the title of the deviation plot (default none)

Details

The generalised bent cable of Khan & Kar (2018) abruptly changes to a final linear phase. The fit applies to the outline converted to polar coordinates based on an initial axis location (two values input to the axis.start function argument).

The `nls` function is applied to residual sums of squares to estimate seven unknown parameters (u, v, a1, k1, kdiff1,kdiff2, g and kappa).

kdiff1 is the difference in slopes k2-k1, and kdiff2 is the difference k3 - (k2-k1)

If quadratic is TRUE then kappa is set at 2.0 for the bent cable of Chiu et al (2006). See Khan & Kar (2018) page 1803. See Khan & Kar (2018) page 1803. The central location of transition must be supplied as the 1st coordinate, which is the paramter tau in the generalised bent cable. Tau is not estimated, but is an output value as an angle (radian units) along the outline.

The change of the final linear is SECOND of two locations in the input for m (like tau, this is given, not estimated).

Other spiral parameters (a1, a2, a3) are estimated by calculation (i.e. not through nls).

Value

`parameters1`	spiral axis location (u,v), and spiral parameters a1, k1, a2, k2,kdiff1,a3,k3,kdiff2, g, kappa, tau
`parameters2`	standard deviations of fit (radius) and test of randomness
`parameters3`	N of outline points, degrees of freedom, and was outline flipped
`parameters4`	start and end of transition: angle, coord no. and arc distance
`xpredict, ypredict`	are the coordinates for the fitted radius (translated or flipped outline)
`angle`	is the angle (radians) of the actual & fitted point about the spiral axis
`arc.length`	is the arc length from the first point (zero) to the last point on an outline
`deviations`	diference between in radii between actual and fit from the estimated spiral axis
`fitted`	the fitted radius from estimated spiral axis
`xpred.orig, ypred.orig`	coordinates for the fitted radius in the original coordinate axes

Warning

start values need to be close to final so that nls converges

Note

g is the gamma of Chiu (2006) and Khan and Kar (2018)

Author(s)

A.E. Aldridge

References

Chiu, G., Lockhart, R. and Routledge, R. (2006), Bent-Cable Regression Theory and Applications, Journal of the American Statistical Association, 101, 542-553. DOI: 10.1198/016214505000001177 http://faculty.washington.edu/gchiu/Articles/bentcable-jasa.pdf

Khan, S.A. and Kar, S.C. (2018). Generalized bent-cable methodology for changepoint data: a Bayesian approach. Journal of Applied Statistics, 45(10), 1799-1822.

See Also

`fitBentCable` and using kappa for more general shape to the transition.

Examples

```
## Not run:
 z <- generateBentCable2LinearSpiral()
 xs <- fitBentCable2Linear(x = z$x, y=z$y, m=c(50,150), axis.start=c(1000,1000),
  plot.fit=TRUE, plot.deviations=TRUE)
# assumes four graphics windows are open (fit on device 3, and deviations on device 4)
 xs$parameters1
# should be close to the estimates in the generation of the spiral.

## End(Not run)
```

`fitCheckL2Spiral` *locates where two linear spirals (L2) change on an outline*

Description

Locates best outline candidate for the change between TWO connected, linear spirals

Usage

```
fitCheckL2Spiral(x , y , axis.start = c(NULL, NULL),
         start.values = c(a = 2, k1 = 0.2, k2 = 0.2),
         specimen.name = " ", tolerance.step = 0.1,
         forward = 0, back = 0, plot.ss = TRUE,
         plot.detail = FALSE, keep.output = FALSE,
         no.error.messages = TRUE, orient.check = TRUE)
```

Arguments

`x`	x coord
`y`	y coord
`axis.start`	start in x, y
`specimen.name`	specimen name
`start.values`	initial values for other unknowns in strict order of : distance to start of spiral (a), and two spiral angles (k1, k2) in radian units.
`tolerance.step`	for nls convergence
`forward`	start this many points from umbo (default 8)
`back`	only go as far from end of outline (default 8)
`plot.ss`	plot the residual deviation (default is yes)
`plot.detail`	plot all relevant rss's (the default), otherwise only the usual rss
`keep.output`	default is no as usually only need minimum location
`no.error.messages`	default is yes to suppress what can be many non convergences
`orient.check`	check the orientation of the outline (default is to check)

Details

Iteration number and change point on outline can differ because some iterations do not converge.

This function has not been vectorised, or checked if vectorising could save a lot of execution time. Watching the progress steps gives a sense of how nls is performing with the starting values.

Value

yseq	iteration number
ystep	coordinate pair along outline where a change
ss.usual	residual deviation at the change point
axis.x	horizontal location for axis of best two spiral
axis.y	vertical location for axis of best two spiral

Warning

expect some gaps if convergence impossible along outline

Note

By default the iteration counts are flushed to the console (in tens). Warning messages from nls have been suppressed (silent = TRUE in the try function). Can reduce the nls tolerance and narrow the outline range to check stability of change location.

Using optim (instead of nls) increases success in the fitting sequence, but it is slower. Use of sapply barely reduces the time to check the coordinate sequence for change in spiral.

DO VERIFY THE CHANGE point by varying the number of outline points (see forward and back arguments)

Author(s)

A.E. Aldridge

References

Aldridge (1999) Brachiopod outline and episodic growth. Paleobiology 25, 471-482

See Also

fitAnyLinear

Examples

```
## Not run:

doSetupGraphics()
set.seed(1998)
# generate two linear spirals (35, and 40 degrees) with default change at 100 coordinate
 z <- generateAnyLinear( k1=35,k2=35,k3=40,k4=40, dist.zero=5)
#
# now locate where spirals change with interactive guess at axis location

 xchk <- fitCheckL2Spiral( x=z$x, y = z$y)

# can locate a starting axis location based on fitting a single linear spiral (not shown)
#
 xchk <- fitCheckL2Spiral( x=z$x, y = z$y, axis.start=c(130,115))

# fit a two linear spiral fit based on the change found at 100th coordinate

xs2 <- fitAnyLinear( x=z$x, y=z$y, m=100, plot.deviations=T, plot.fit=T)
```

```
xs2[2] # for statistics on the fit
#
# By default, the fit is plotted on device 3, and deviations on device 4
# You can also check any other coordinate in sequence from umbo by changing m
# and viewing the increase (decrease) in the residual standard deviation
#
xs2a <- fitAnyLinear( x=z$x, y=z$y, m=101, plot.deviations=T, plot.fit=T)
xs2a[2] # for statistics on the fit
xs2b <- fitAnyLinear( x=z$x, y=z$y, m= 99, plot.deviations=T, plot.fit=T)
xs2b[2] # for statistics on the fit
#

## End(Not run)
```

`fitCheckL3Restricted` *locates a SECOND change between THREE linear spirals, given a FIRST change location.*

Description

Locates change between THREE spirals when change testing is restricted between two specified locations, but still fitting the entire outline.

Usage

```
fitCheckL3Restricted(x, y, m = NULL, axis.start = c(NULL, NULL), specimen.name = " ",
                     start.values = c(a = 2, k1 = 0.2, k2 = 0.2, k3 = 0.2),
                     tolerance.step = 0.01, from = NULL, to = NULL,
                     plot.ss = TRUE, keep.output = FALSE,
                     no.error.messages = TRUE, orient.check = TRUE)
```

Arguments

`x`	x coord
`y`	y coord
`m`	the given first change location (as mth coordinate) from start of outline
`axis.start`	start location as (x,y) coordinates
`specimen.name`	specimen name used as text for graphic title
`start.values`	initial values for other unknowns in strict order of : distance to start of spiral (a), and two spiral angles (k1, k2) in radian units.
`tolerance.step`	step used in the nls algorithm (see nls)
`from`	first outline location to begin search for change location
`to`	outline location to end search
`plot.ss`	plot
`keep.output`	graphic summary has change points, otherwise can keep a summary of every step
`no.error.messages`	keep the silent = T in the try command for nls.
`orient.check`	check the orientation of the outline (default is to check)

Details

Given a possible change point (m) the function steps from **specified beginning** (forward argument) of the outline to the m th location. At each coordinate step three spirals are fitted.

Tolerance step is the tolerance input to the nonlinear fitting function nls (base)

This function has not been vectorised, or checked if vectorising could save a lot of execution time. Watching the progress steps gives a sense of how nls is performing with the starting values.

If dense coordinates consider decimating the outline before fitting spirals.

Value

change point

`yseq`	iteration number
`ystep`	coordinate pair along outline where a change
`ss.usual`	residual deviation at the change point
`axis.x`	horizontal location for axis of best two spiral
`axis.y`	vertical location for axis of best two spiral

Warning

from and `to` should both be on one side of the given change. The function plots these locations on the outline to assist with any errors generated.

There can be gaps in plot of std. dev. vs outline sequence `plot.ss=TRUE` where nls fails to converge.

Note

from is where the fitting starts on the outline, especially useful if there is a pathological shape near to the umbo (e.g. protoconch, sudden change to incurving) where only a few points (i.e. not enough to fit another curve).

to is the last outline coordinate to check for change in spiral.

The purpose of the from/to restriction is to speed up the sequential checking of each coordinate as a possible second spiral change.

Author(s)

A.E. Aldridge

References

Aldridge (1999) Brachiopod outline and episodic growth. Paleobiology 25, 471-482

See Also

`checkL3Spiral fitCheckL2Spiral VerifyL3Change`

Examples

```
## Not run:
# generate three linear spirals (L3) with angles 35,55,60 and changes at 100,150 (default)
set.seed(1998)
  z <- generateLinearSpirals( k1=35,k2=35,k3=55,k4=60, dist.zero=5)
#
# search first for the first (and largest) change in spiral angle.
#
xchk <- fitCheckL2Spiral( x=z$x, y = z$y) # should locate change at 103th coordinate
#
# input this change into searching for a second change in spiral angle
# going FORWARD from this change and note the residual standard deviation

#
xchk3 <- fitCheckL3Restricted( x=z$x,y=z$y, m=103, from=10, to=90)
# find minimal drop in error so no important change in this restricted range.
# next is to try a range AFTER the  given change location
#
xchk3 <- fitCheckL3Restricted( x=z$x,y=z$y, m=103, from=110, to=190)
#
# change estimated at coordinate 153 with smaller residual error
#
# Verify that this pair of changes (103,153) do give the best possible overall fit
#
xs3.verify <- verifyL3Change( x=z$x, y=z$y, m=c(103,153),  pp=seq(-4,2,1) )
#
# output the matrix of minima, find the least of these is at (100,150) coordinate locations
# coordinate locations.  Now can fit this final spiral as a L3 spiral
  xs3.verify  # prints out the matrix of coordinate pairs and residual standard deviation
#
xs3 <- fitAnyLinear( x =z$x, y=z$y, m=c(100,150) )
#
xs3[1]  # parameter values or use xs3$parameters1
xs3[2]  # statistics of fit or xs3$parameters2, mssd ratio now close to unity (1.0)
#

## End(Not run)
```

`fitCheckL3Spiral` *locate a SECOND change between THREE linear spirals, given FIRST change location.*

Description

Change in location between THREE spirals working either side of the given change for two spirals

Usage

```
fitCheckL3Spiral(x = x, y = y, m = NULL, axis.start = c(NULL, NULL),
                start.values = c(a = 2, k1 = 0.2, k2 = 0.2, k3 = 0.2),
               specimen.name = " ", tolerance.step = 0.01, forward = 0, back = 0,
                 plot.ss = TRUE, plot.detail = TRUE, keep.output = FALSE,
                 no.error.messages = TRUE, orient.check = TRUE)
```

Arguments

x	x coord
y	y coord
m	given change location
axis.start	as
start.values	initial values for other unknowns in strict order of : distance to start of spiral (a), and two spiral angles (k1, k2) in radian units.
specimen.name	name of outline being checked for a graph title
tolerance.step	step used in the nls algorithm (see nls)
forward	forward, start to the fitting
back	back from end of selected outline
plot.ss	plot the residual sums of squares along the outline
plot.detail	plot the sequential deviation rss (don't include other rss)
keep.output	graphic summary has change points, otherwise can keep a summary of every step
no.error.messages	keep the silent = T in the try command for nls.
orient.check	check the orientation of the outline (default is to check)

Details

Given a change point (m) the function steps along the outline to find if there is another change location. At each coordinate step three spirals are fitted.

Steps exclude six points at outline ends and at either side of the given change location.

If plot detail is FALSE then comparisons are made with minimum found for TWO spirals and the base (or overall minimum) expected for the outline. The defaut (TRUE) shows the unscaled RSS values at each step along the outline. Gaps indicate nls failure.

Tolerance step is the tolerance input to the nonlinear fitting function nls (base)

This function has not been vectorised, or checked if vectorising could save a lot of execution time. Watching the progress steps gives a sense of how nls is performing with the starting values.

If dense coordinates consider decimating the outline before fitting spirals.

Value

location of second change point (so now three spirals)

yseq	iteration number
ystep	coordinate pair along outline where a change
ss.usual	residual deviation at the change point
axis.x	horizontal location for axis of best two spiral
axis.y	vertical location for axis of best two spiral

Warning

likely to see gaps in plot where nls has failed to converge

Note

It can take time to step through all possibilities with a lot of warnings. Hint: use a fixed start value for the spiral axis (e.g. the estimate from single spiral fit) to speed up the testing the coordinate sequence.

Author(s)

A.E. Aldridge

References

Aldridge (1999) Brachiopod outline and episodic growth. Paleobiology 25, 471-482

See Also

`fitCheckL2Spiral` provides the given change point for two spirals that is input as `m` in this function.

Examples

```
## Not run:
# generate three linear spirals (L3) with angles 35,40,55 and changes at 100,150 (default)
set.seed(1998)
 z <- generateAnyLinear( k1=35,k2=35,k3=40,k4=55, dist.zero=5)
#
# search first for the first (and largest) change in spiral angle
#
xchk <- fitCheckL2Spiral( x=z$x, y = z$y) # should locate change at 149th coordinate
#
# input this change into searching for a second change in spiral angle
#
 xchk3 <- fitCheckL3Spiral( x=z$x, y=z$y, m=149) # finds second change at 99th coordinate
#
# Verify that this pair of changes (99,149) do give the best possible overall fit
#
 xs3.verify <- verifyL3Change( x=z$x, y=z$y, m=c(99,149) )
#
# output the matrix of minima, find the least of these is at (100,150) coordinate locations
# coordinate locations.  Now can fit this final spiral as a L3
  xs3.verify
#
xs3 <- fitAnyLinear( x =z$x, y=z$y, m=c(100,150) )
#
xs3[1]  # parameter values or use xs3$parameters1
xs3[2]  # statistics of fit or xs3$parameters2, mssd ratio now close to unity (1.0)
#

## End(Not run)
```

`fitCheckL4Spiral` *Locate THIRD change between FOUR linear spirals, given TWO change locations.*

Description

Checks the possibility of a fourth spiral given two change points

Usage

```
fitCheckL4Spiral(x, y, m = c(NULL, NULL), axis.start = c(NULL, NULL),
      start.values = c(a = 2, k1 = 0.2, k2 = 0.2, k3 = 0.2, k4 = 0.2),
      specimen.name = " ", tolerance.step = 0.01, forward = 0, back = 0,
      plot.detail = TRUE, plot.ss = TRUE, keep.output = FALSE,
      no.error.messages = TRUE, orient.check = TRUE)
```

Arguments

`x`	x coord
`y`	y coord
`m`	the given two change locations (as mth coordinates) from start of outline
`axis.start`	start location as (x,y) coordinate
`specimen.name`	specimen name used as text for graphic title
`start.values`	initial values for other unknowns in strict order of : distance to start of spiral (a), and three spiral angles (k1, k2, k3) in radian units.
`tolerance.step`	step used in the nls algorithm (see nls)
`forward`	forward, start to the fitting
`back`	back from end of selected outline
`plot.ss`	plot the residual sums of squares along the outline
`plot.detail`	plot the sequential deviation rss (don't include other rss)
`keep.output`	graphic summary has change points, otherwise can keep a summary of every step
`no.error.messages`	keep the silent = T in the try command for nls.
`orient.check`	check the orientation of the outline (default is to check)

Details

Given two change points (m) the function steps along the outline to find if there is another change location. At each coordinate step four spirals are fitted.

If plot detail is FALSE then comparisons are made with minimum found for TWO spirals and the base (or overall minimum) expected for the outline.

Fitting steps around the given change points, 6 locations either side of each.

If forward and back are used to eliminate part of the outline, the fitting does NOT include these eliminated parts of the outline.

Tolerance step is the tolerance input to the nonlinear fitting function nls (base)

This function has not been vectorised, or checked if vectorising could save a lot of execution time. Watching the progress steps gives a sense of how nls is performing with the starting values.

If dense coordinates consider decimating the outline before fitting spirals.

Error is output when there are no values for m, or the m values are NOT in a strict increasing sequence.

Value

location of third change point (so now four linear spirals)

`yseq`	iteration number
`ystep`	coordinate pair along outline where a change
`ss.usual`	residual deviation at the change point
`axis.x`	horizontal location for axis of best two spiral
`axis.y`	vertical location for axis of best two spiral

Warning

likely to see gaps in plot where nls has failed to converge

Note

It can take time to step through all possibilities with a lot of warnings. Consider using a fixed start value for the spiral axis (e.g. the estimate from single spiral fit) to speed up the testing the coordinate sequence.

Author(s)

A.E. Aldridge

References

Aldridge (1999) Brachiopod outline and episodic growth. Paleobiology 25, 471-482

See Also

`fitCheckL2Spiral`

`checkL3Spiral fitCheckL3SpiralRestricted verifyL3Change`

Examples

```
## Not run:
set.seed(1998)
# four spirals with angles 35, 40, 45, 55 with changes at 50, 100, 150 (default)
#
 z <- generateLinearSpirals( k1=35,k2=40,k3=45,k4=55, dist.zero=5) #
#
# checking for two and three spiral changes end up with changes at 101, 150 (actual 100,150)
#
# verify now if there is a third change (ie four linear spirals) before, inbetween,
# or after these two changes.  Not use of axis starting location to reduce iterations (and time)
#
xchk4 <- fitCheckL4Spiral( x=z$x, y = z$y, m=c(101,150) ,axis.start=c(135,112))
```

```
#
# fit the final four spiral and note that the mssd ratio is still way small
# inspection of deviations shows a step change at the second change.
xs4 <- fitAnyLinear(x=z$x, y=z$y, m=c(50,101,150), axis.start=c(135, 112) )
#
xs4[2] # statistics of fit
#
# manually checking changes either side shows that changes at 50,100,150
# not only have a little less residual sd, but more importantly the mssd ratio is better
#
xs4 <- fitAnyLinear(x=z$x, y=z$y, m=c(50,100,150), axis.start=c(135, 112) )
#
xs4[2] # statistics of fit
#

## End(Not run)
```

`fitCubic` *Fits a cubic spiral to an outline of x,y coordinates*

Description

Fits a cubic spiral to given outline coordinates. Cubic includes constant (intercept), linear and quadratic terms.

Usage

```
fitCubic(x, y, axis.start = c(NULL, NULL), flip.x = FALSE,
               dist.zero = 2, angle.start = 40,
               quadratic.start = 0.02, cubic.start = 0.001,
               move.by = NULL, tolerance.step = 0.01, angle.deg = TRUE,
               plot.fit =FALSE, plot.deviations = FALSE,
               plot.diagnostics = FALSE, title.label="")
```

Arguments

`x`	x coordinate
`y`	y coordinate
`axis.start`	location (u,v) coordinates as a starting axis for spiral being fitted.
`flip.x`	flip the x axis (default, no flipping) so outline is anticlockwise.
`dist.zero`	initial distance to outline start when angle is zero
`angle.start`	initial slope of spiral (in degrees)
`quadratic.start`	initial quadratic term
`cubic.start`	initial cubic term for fitting
`move.by`	moves (translates) the entire outline so all coordinates are positive
`tolerance.step`	nls tolerance
`angle.deg`	units for spiral angle (default is degrees)
`plot.fit`	plot the outline with fitted values (default FALSE)

`plot.deviations`
: plot the radial deviations from spiral fit (default FALSE)

`plot.diagnostics`
: plot diagnostics for deviations from fitted spiral (see Detail below)

`title.label`
: text for title (default none)

Details

If spiral angle is in degrees it is converted to radians for computations then converted back to degrees in the output. Outline coordinate pairs (x,y) are to be in sequential order from the beginning of growth (umbo in shells).

Parameters are estimated using nls. Intitial or starting values are function arguements, including the axis location. If initial axis is not not specified then you are asked to interactively locate the start of the spiral axis.

Value

Outline coordinates (Cartesian and polar) summary statistics, deviations from spiral, and fitted values (original coordinate axis and translated axis). see fitAnyLinear for examples of output and meaning.

Warning

read Troubleshooting Guide if not converging or an error

Note

Useful for outlines that appear mostly 'flat' with most curvature very near the umbo.

Author(s)

A.E. Aldridge

References

Aldridge, A.E. (1999) Brachiopod outline and episodic growth. Paleobiology 25, 471-482.

McGhee, G.R.(1980) Shell form in the biconvex articulate Brachiopoda: a geometric analysis. Paleobiology 6, 57-76.

See Also

`generateCurvedSpiral fitQuadratic fitQuartic`

Examples

```
## Not run:
# generate a cubic spiral that also includes a quadratic term
#
zc <- generateCurvedSpiral( quadratic= 0.06, cubic= -0.03, quartic=0)
#
# fit this spiral using interactive locating of axis on the graphics window
# use a reasonable location based on the axis from the generated spiral
#
 xc <- fitCubic( x= zc$x, y= zc$y)
```

```
#
# view the summary statistics
#
xc[1]  # can also use xc$parameters1
xc[2]  # can also use xc$parameters2
#

## End(Not run)
```

`fitL1Spiral` *fit a single linear (L1) logarithmic spiral*

Description

fit a single linear logarithmic spiral

Usage

```
fitL1Spiral(x, y, start.u = NULL, start.v = NULL,
                start.values = c(a = 2, k = 0.2),
                flip.x = FALSE, move.by = NULL,
                tolerance.step = 0.01, angle.deg = TRUE)
```

Arguments

`x`	x coord
`y`	y coord
`start.u`	start in x coord
`start.v`	start in y coord
`start.values`	initial values for other unknowns in strict order of : distance to start of spiral (a), and spiral angle (k) in radian units.
`flip.x`	flip
`move.by`	move
`tolerance.step`	nls tolerance
`angle.deg`	units for the spiral angle (default is degrees)

Details

Called by `fitAnyLinear`, not meant as a standalone function.

However start values can be input to help convergence, but need to use radian units.

Uses exponential form of logarithmic spiral as input to `nls`

Value

see fitAnyLinear

Warning

See troubleshooting guide if not converging or an error.

Axis start for the spiral requires input of location as two separate values.

Note

called by `fitAnyLinear`, not meant to be a stand alone function.

Author(s)

A.E. Aldridge

References

Aldridge, A. (1999) Brachiopod outline and episodic growth. Paleobiology 25, 471-482

See Also

`fitAnyLinear`

`fitL2Spiral` *fits TWO linear spirals (L2) at a specified change location*

Description

fits TWO linear spirals (L32) changing at ONE specified change point location

Usage

```
fitL2Spiral(x, y, m = NULL, start.u = NULL, start.v = NULL,
                start.values = c(a = 2, k1 = 0.2, k2 = 0.2),
                flip.x = FALSE, move.by = NULL,
                tolerance.step = 0.01, angle.deg = TRUE)
```

Arguments

`x`	x coord
`y`	y coord
`m`	change location
`start.u`	start in x coord
`start.v`	start in y coord
`start.values`	initial values for other unknowns in strict order of : distance to start of spiral (a), and two spiral angles (k1, k2) in radian units.
`flip.x`	flip
`move.by`	move
`tolerance.step`	nls tolerance
`angle.deg`	units for spiral angles (default degrees)

Details

Called by `fitAnyLinear`, not meant as a standalone function.

Start values can be used to assist convergence when function used alone.

Uses exponential form of logarithmic spirals as input to `nls`

Value

see `fitAnyLinear`

Warning

See Troubleshooting Guide if not converging or an error

Axis start for the spiral requires input of location as two separate values.

Note

called by `fitAnyLinear`

Author(s)

A.E. Aldridge

References

Aldridge, A. (1999) Brachiopod outline and episodic growth. Paleobiology 25, 471-482

`fitL3Spiral` *fits THREE linear spirals (L3) at specified change locations*

Description

fits THREE linear spirals (L3) changing at TWO specified change point locations

Usage

```
fitL3Spiral(x, y, m = c(NULL, NULL), start.u = NULL, start.v = NULL,
           start.values = c(a = 2, k1 = 0.2, k2 = 0.2, k3 = 0.2),
           flip.x = FALSE, move.by = NULL,
           tolerance.step = 0.01, angle.deg = TRUE)
```

Arguments

`x`	x coord
`y`	y coord
`m`	change location
`start.u`	start in x coord
`start.v`	start in y coord
`start.values`	initial values for other unknowns in strict order of : distance to start of spiral (a), and three spiral angles (k1, k2, k3) in radian units.
`flip.x`	flip
`move.by`	move
`tolerance.step`	nls tolerance
`angle.deg`	units for spiral angles (default is degrees)

Details

Called by `fitAnyLinear`, not meant as a standalone function.

Start values for assisting convergence if function used alone.

Uses exponential form of logarithmic spirals as input to `nls`

Value

see `fitAnyLinear`

Warning

See Troubleshooting Guide if not converging or an error.

Axis start for the spiral requires input of location as two separate values.

Note

called by `fitAnyLinear`

Author(s)

A.E. Aldridge

References

Aldridge, A. (1999) Brachiopod outline and episodic growth. Paleobiology 25, 471-482

`fitL4Spiral` *fits FOUR linear spirals (L4) at specified change locations*

Description

fits FOUR linear spirals (L4) changing at THREE specified change point locations

Usage

```
fitL4Spiral(x, y, m = c(NULL, NULL, NULL), start.u = NULL, start.v = NULL,
           start.values = c(a = 2, k1 = 0.2, k2 = 0.2, k3 = 0.2, k4 = 0.2),
           flip.x = FALSE, move.by = NULL,
           tolerance.step = 0.01, angle.deg = TRUE)
```

Arguments

`x`	x coord
`y`	y coord
`m`	given change points
`start.u`	start location x
`start.v`	start location y
`start.values`	initial values for other unknowns in strict order of : distance to start of spiral (a), and four spiral angles (k1, k2, k3, k4) in radian units.

flip.x	flip
move.by	move
tolerance.step	tolerance for nls
angle.deg	units for spiral angle output (default is TRUE)

Details

Called by fitAnyLinear, not meant as a standalone function.

Start values may assist convergence when function is used alone.

Axis start needs input as two separate values.

Uses exponential form of logarithmic spirals as input to nls

Value

see fitAnyLinear

Author(s)

A.E. Aldridge

References

Aldridge,A. (1999) Brachiopod outline and episodic growth. Paleobiology 25, 471-482

---

fitLinear2BentCable *Fits a linear-to-bent cable spiral on an outline of x,y coordinates*

Description

Fits linear connected to a generalised bent cable spiral on the given outline coordinates. So there are FOUR phases of growth: two linear then a quadratic transition to a final linear phase (ie three linear phases and one curved phase of growth).

Usage

```
fitLinear2BentCable(x, y, m = c(NULL, NULL), axis.start = c(NULL, NULL), quadratic = TRUE,
          start.values = c(a0 = 2, k0 = 36, k1 = 40, k2 = 55, g = 0.3, kappa = 2.0),
          angle.deg = TRUE, flip.x = FALSE, move.by = NULL, tolerance.step = 0.1,
              trace.check = FALSE, plot.fit = FALSE, plot.deviations = FALSE,
         plot.diagnostics = FALSE, title.label = "fit to Linear to Bent Cable outline ")
```

Arguments

x	x coordinate
y	y coordinate
m	location coordinates for change between linear phases and centre of transition
axis.start	starting location of spiral axis in (x,y) coordinates
quadratic	should a quadratic bend be forced (default no quadratic)

start.values	initial values for other unknown values in order of distance axis to to start of first linear spiral (a0), three spiral angles (k0, k1, k2), and the shape of the transition (g, kappa)
angle.deg	units of spiral angle as start values and output (default FALSE)
flip.x	do we flip the outline to ensure positive spiral angle values
move.by	do we move (translate) the outline to ensure all coordinates are postive
tolerance.step	tolerance step for nls used in the fitting (minscale not specified)
trace.check	do we wish to trace the steps of the nls algorithm (default FALSE)
plot.fit	plot the outline with fitted values (default FALSE)
plot.deviations	plot the radial deviations from spiral fit (default FALSE)
plot.diagnostics	plot diagnostics for deviations from fitted spiral (see Detail below)
title.label	text for the title of the deviation plot (default none)

Details

Initial linear growth abruptly changes to generalised bent cable of Khan & Kar (2018). The fit applies to the outline converted to polar coordinates based on an initial axis location (two values input to the axis.start function argument).

The nls function is applied to residual sums of squares to estimate seven unknown parameters (u, v, a0, k0, k1, k2, g and kappa). If quadratic is TRUE then kappa is set at 2.0 for the bent cable of Chiu et al (2006). See Khan & Kar (2018) page 1803. See Khan & Kar (2018) page 1803.

The change of the first linear is first of two locations in the input for m. The central location of transition must be supplied as the 2nd coordinate, which is the paramter tau in the generalised bent cable. Tau is not estimated, but is an output value as an angle (radian units) along the outline. Other spiral parameters (a1, a2) are estimated by calculation (i.e. not through nls).

Value

parameters1	spiral axis location (u,v), and spiral parameters a0, k0, a1, k1, a2, k2, g, kappa, tau
parameters2	standard deviations of fit (radius) and test of randomness
parameters3	N of outline points, degrees of freedom, and was outline flipped
parameters4	start and end of transition: angle, coord no. and arc distance
xpredict, ypredict	are the coordinates for the fitted radius (translated or flipped outline)
angle	is the angle (radians) of the actual & fitted point about the spiral axis
arc.length	is the arc length from the first point (zero) to the last point on an outline
deviations	diference between in radii between actual and fit from the estimated spiral axis
fitted	the fitted radius from estimated spiral axis
xpred.orig, ypred.orig	coordinates for the fitted radius in the original coordinate axes

Warning

start values need to be close to final so that nls converges

Note

g is gamma of Chiu (2006)

Author(s)

A.E. Aldridge

References

Chiu, G., Lockhart, R. and Routledge, R. (2006), Bent-Cable Regression Theory and Applications, Journal of the American Statistical Association, 101, 542-553. DOI: 10.1198/016214505000001177 http://faculty.washington.edu/gchiu/Articles/bentcable-jasa.pdf

Khan, S.A. and Kar, S.C. (2018). Generalized bent-cable methodology for changepoint data: a Bayesian approach. Journal of Applied Statistics, 45(10), 1799-1822.

See Also

`fitBentCable` and using kappa for more general shape to the transition.

Examples

```
## Not run:
 z <- generateLinear2BentCableSpiral()
 xs <- fitLinear2BentCable(x = z$x, y=z$y, m=c(30,150), axis.start=c(1000,1000) ,
  plot.fit=TRUE, plot.deviations=TRUE)
# assumes four graphics windows are open (fit on device 3, and deviations on device 4)


## End(Not run)
```

`fitLinearQuadratic` *fits a Linear Quadratic (LQ) spiral*

Description

Fits linear connected to a quadratic spiral

Usage

```
fitLinearQuadratic(x, y, m = NULL, axis.start = c(NULL, NULL),
                   start.values = c(a1 = 2, k1 = 40, k2 = 55),
                   angle.deg = TRUE, plot.fit = FALSE,
                   plot.deviations = FALSE, plot.diagnostics = FALSE,
                   flip.x = FALSE, move.by = NULL,tolerance.step = 0.1,
                   trace.check = FALSE, title.label = "")
```

Arguments

x	x coord
y	y coord
m	location (mth coordinate) where there is a change in spirals
axis.start	start location of spiral axis as a (x,y) coordinate
start.values	initial values for other unknown values in order of distance axis to to start of linear spiral (a1), two spiral angles (k1, k2). The quadratic component is calculated (see details)
angle.deg	units for spiral angle (default is degrees)
plot.fit	plot the outline with fitted values (default FALSE)
plot.deviations	plot the radial deviations from spiral fit (default FALSE)
plot.diagnostics	plot diagnostics for deviations from fitted spiral (see Detail below)
flip.x	do we flip the outline to ensure positive spiral angle values
move.by	do we move (translate) the outline to ensure all coordinates are postive
tolerance.step	tolerance step for nls used in the fitting (minscale not specified)
trace.check	do we wish to trace the steps of the nls algorithm (default FALSE)
title.label	text for the title of the deviation plot (default none)

Details

Allows for continual decrease/increase in spiral angle after the change from linear. The quadratic estimate is calculated from k1, k2, and the change location.

Value

parameters1	spiral axis location (u,v), and spiral parameters a1, k1, k2, q
parameters2	standard deviations of fit (radius) and test of randomness
xpredict, ypredict	are the coordinates for the fitted radius (translated or flipped outline)
angle	is the angle (radians) of the actual & fitted point about the spiral axis
arc.length	is the arc length from the first point (zero) to the last point on an outline
deviations	diference between in radii between actual and fit from the estimated spiral axis
fitted	the fitted radius from estimated spiral axis
xpred.orig, ypred.orig	coordinates for the fitted radius in the original coordinate axes

Note

Sequentially trial other change locations and not whether the output parameters2 improves, or not.

Author(s)

A.E. Aldridge

References

no references yet

See Also

`fitQuadraticLinear fitBentCable`

Examples

```
## Not run:
z <- generateLinearQuadraticSpiral(k2=0,q= -0.5, equal=F )
xsq <- fitLinearQuadratic( x=z$x, y=z$y, m=100, axis.start=c(990,990))
#
z <- generateLinearQuadraticSpiral( dist.zero=5,k1=40,k2=55, equal=F  )
xsq <- fitLinearQuadratic( x=z$x, y=z$y, m=50, axis.start=c(990,990))

## End(Not run)
```

`fitQuadratic` *Fits a quadratic spiral (Q) to a given outline*

Description

Fits a quadratic spiral to an outline. The quadratic includes terms for intercept (distance at zero angle), and slope (spiral angle)

Usage

```
fitQuadratic(x, y, axis.start = c(NULL, NULL), flip.x = FALSE,
              dist.zero = 2, spiral.angle = 40, quadratic = 0.02,
              move.by = NULL, tolerance.step = 0.01, angle.deg = TRUE,
            plot.fit = FALSE, plot.deviations = FALSE, plot.diagnostics = FALSE,
              title.label = "")
```

Arguments

`x`	x coordinate of outline
`y`	y coordinate of outline
`axis.start`	location (u,v) coordinates as a starting axis for spiral being fitted.
`flip.x`	flip the x axis (default, no flipping) so outline is anticlockwise.
`dist.zero`	initial distance to outline start when angle is zero
`spiral.angle`	initial slope of spiral (in degrees)
`quadratic`	initial value for quadratic term.
`move.by`	moves (translates) the entire outline so all coordinates are positive
`tolerance.step`	nls tolerance
`angle.deg`	units for spiral angle (default is degrees)
`plot.fit`	plot the outline with fitted values (default FALSE)
`plot.deviations`	plot the radial deviations from spiral fit (default FALSE)

`plot.diagnostics`
: plot diagnostics for deviations from fitted spiral (see Detail below)

`title.label`
: text for the title of the deviation plot (default none)

Details

If spiral angle is in degrees it is converted to radians for computations then converted back to degrees in the output. Outline coordinate pairs (x,y) must be in sequential order from the beginning of growth (umbo in shells).

Value

Outline coordinates (Cartesian and polar) summary statistics, deviations from spiral, and fitted values (original coordinate axis and translated axis). see fitAnyLinear for examples.

Warning

read Troubleshooting Guide if not converging or an error

Note

Useful for almost flat outlines with curvature very near the umbo.

Author(s)

A.E. Aldridge

References

Aldridge, A.E. (1999) Brachiopod outline and episodic growth. Paleobiology 25, 471-482.

McGhee, G.R.(1980) Shell form in the biconvex articulate Brachiopoda: a geometric analysis. Paleobiology 6, 57-76.

See Also

`generateCurvedSpiral fitCubic fitQuartic fitL4Spiral`

Examples

```
## Not run:
# generate a quadratic spiral (Q)
#
zq <- generateCurvedSpiral( quadratic=0.03, cubic=0, quartic=0)
#
# fit this spiral using interactive locating of axis on the graphics window
# use a reasonable location based on the axis from the generated spiral
#
xq <- fitQuadratic( x= zq$x, y= zq$y)
#
# view the summary statistics
#
xq[1]  # can also use xq$parameters1
xq[2]  # can also use xq$parameters2
#

## End(Not run)
```

`fitQuadraticLinear` *Fits a Quadratic Linear (QL) spiral*

Description

Fits quadratic connected to a linear spiral

Usage

```
fitQuadraticLinear(x, y, m = NULL, axis.start = c(NULL, NULL),
                   start.values = c(a1 = 2, k1 = 40, k2 = 55),
                   angle.deg = TRUE, plot.fit = FALSE,
                   plot.deviations = FALSE, plot.diagnostics = FALSE,
                   flip.x = FALSE, move.by = NULL,
                   tolerance.step = 0.1, trace.check = FALSE,
                   title.label = "")
```

Arguments

`x`	x coord
`y`	y coord
`m`	location (mth coordinate) where there is a change in spirals
`axis.start`	start location of spiral axis as a (x,y) coordinate
`start.values`	initial values for other unknown values in order of distance axis to to start of linear spiral (a1), two spiral angles (k1, k2). The quadratic component is calculated (see details)
`angle.deg`	units for spiral angle (default is degrees)
`plot.fit`	plot the outline with fitted values (default FALSE)
`plot.deviations`	plot the radial deviations from spiral fit (default FALSE)
`plot.diagnostics`	plot diagnostics for deviations from fitted spiral (see Detail below)
`flip.x`	do we flip the outline to ensure positive spiral angle values
`move.by`	do we move (translate) the outline to ensure all coordinates are postive
`tolerance.step`	tolerance step for nls used in the fitting (minscale not specified)
`trace.check`	do we wish to trace the steps of the nls algorithm (default FALSE)
`title.label`	text for the title of the deviation plot (default none)

Details

Allows for continual decrease/increase in spiral angle after the change from linear. The quadratic estimate is calculated from k1, k2, and the change location.

Value

parameters1	spiral axis location (u,v), and spiral parameters a1, k1, k2, q
parameters2	standard deviations of fit (radius) and test of randomness
xpredict, ypredict	are the coordinates for the fitted radius (translated or flipped outline)
angle	is the angle (radians) of the actual & fitted point about the spiral axis
arc.length	is the arc length from the first point (zero) to the last point on an outline
deviations	diference between in radii between actual and fit from the estimated spiral axis
fitted	the fitted radius from estimated spiral axis
xpred.orig, ypred.orig	coordinates for the fitted radius in the original coordinate axes

Note

Sequentially trial other change locations and check whether the output parameters2 improves, or not.

Author(s)

A.E. Aldridge

References

no references yet

See Also

generateQuadraticLinearSpiral fitQuadraticLinear fitBentCable

Examples

```
## Not run:
 z <- generateQuadraticLinearSpiral(k2=0,q= -0.5, equal=F )
xs <- fitQuadraticLinear( x=z$x, y=z$y, m=100, axis.start=c(990,990))
 xs$parameters1

## End(Not run)
```

---

fitQuartic *Fits quartic spiral to an outline of x,y coordinates*

Description

Fits a QUARTIC (QQ) spiral to given outline coordinates. Quartic includes intercept, linear, quadratic, and cubic terms.

Usage

```
fitQuartic(x, y, axis.start = c(NULL, NULL), flip.x = FALSE,
                 dist.zero = 2, spiral.angle = 40,
                 quadratic = 0.02, cubic = 1e-04, quartic = 1e-04,
                 move.by = NULL, tolerance.step = 0.01, angle.deg = TRUE,
                 plot.fit = FALSE, plot.deviations = FALSE,
                 plot.diagnostics = FALSE, title.label="")
```

Arguments

x	x coordinate
y	y coordinate
axis.start	location (u,v) coordinates as a starting axis for spiral being fitted.
flip.x	flip the x axis (default, no flipping) so outline is anticlockwise.
dist.zero	initial distance to outline start when angle is zero
spiral.angle	initial slope of spiral (in degrees)
quadratic	initial quadratic term
cubic	initial cubic term
quartic	initial quartic term
move.by	moves (translates) the entire outline so all coordinates are positive
tolerance.step	nls tolerance step
angle.deg	units for spiral angle (default is degrees)
plot.fit	plot the outline with fitted values (default FALSE)
plot.deviations	plot the radial deviations from spiral fit (default FALSE)
plot.diagnostics	plot diagnostics for deviations from fitted spiral (see Detail below)
title.label	text for the title of the deviation plot (default none)

Details

If spiral angle is in degrees it is converted to radians for computations then converted back to degrees in the output. Outline coordinate pairs (x,y) are to be in sequential order from the beginning of growth (umbo in shells).

Value

Outline coordinates (Cartesian and polar) summary statistics, deviations from spiral, and fitted values (original coordinate axis and translated axis). see fitAnyLinear for values output.

Note

Useful for flat outlines with most curvature at or near the umbo. Example is in the data set exRafinesquina where the adult part of growth matches a quartic spiral.

Otherwise, used to exclude this type of spiral to a given outline.

Author(s)

A.E. Aldridge

References

Aldridge, A.E. (1999) Brachiopod outline and episodic growth. Paleobiology 25, 471-482.

McGhee, G.R.(1980) Shell form in the biconvex articulate Brachiopoda: a geometric analysis. Paleobiology 6, 57-76.

See Also

generateCurvedSpiral exRafinesquina

Examples

```
## Not run:
# generate a quartic spiral (QQ) that also includes  quadratic and cubic terms
#
 zqq <- generateCurvedSpiral( quadratic= 0.01, cubic= -0.06, quartic=0.01)
#
# fit this spiral using interactive locating of axis on the graphics window
# use a reasonable location based on the axis from the generated spiral
# or use close to known value for axis and other parameters
#
xqq <- fitQuartic( x= zqq$x, y= zqq$y  )
#
 xqq <- fitQuartic( x= zqq$x, y= zqq$y, axis.start=c(999,999)  )
#
# view the summary statistics
#
xqq[1]  # can also use xc$parameters1
xqq[2]  # can also use xc$parameters2

## End(Not run)
```

---

generateAnyLinear *generate up to FOUR connected spirals as a single outline*

Description

generate up to four spirals using three specified spiral angles

Usage

```
generateAnyLinear(nstart = 200, change = c(50, 100, 150), rate = 0.4,
                   k1 = 35, k2 = 35, k3 = 35, k4 = 35, dist.zero = 2,
                  noise = 0.01,noise.increase = FALSE, orient = "NO",
                  distance = "far", angle = 0, plane = "X", plot = T,
                  waves = 1, show.changes = TRUE, theta.start = 0.001)
```

Arguments

nstart	number of outline points to generate (default 200)
change	where spirals change along the outline (default at the 50th, 100th and 150th
rate	rate at which the spiralhe rate spiral is grown (see details)

k1	spiral angle 1 (note if all equal then only one spiral episode is generated)
k2	spiral angle 2 (note if all equal then only one spiral episode is generated)
k3	spiral angle 3 (note if all equal then only one spiral episode is generated)
k4	spiral angle 4 (note if all equal then only one spiral episode is generated)
dist.zero	initial distance between axis and spiral when angle (theta) is zero
noise	standard deviation of error added to the distance from spiral axis
noise.increase	does error increase LINEARLY from umbo, default is no increase
orient	orientation
distance	distance to translate the outline
angle	angle to orient the spiral OUT OF PLANE
plane	to orient the spiral outline ("X" or "Y")
plot	plot
waves	waves or cycles in the deviations
show.changes	mark the outline locations where spirals change
theta.start	the initial value for starting angle (to avoid zero)

Details

Generates a spiral based on constant arc length increment (rate). Other function arguments are used to test fitting on a range of variations to an underlying spiral or spirals. For example, adding cycles to the growth, or twisting the outline out of the X-Y plane (outline coordinates are projection on X-Y plane).

Cycles of fixed amplitude and phase are fixed by the **waves** argument.

Value

xy	data frame with spiral coordinates as x, y and logr, angle

Warning

may need to trial and error to find best orientation plotting (TRUE) is required for obtaining output data frame

Note

Small spiral angles (eg k= 30) generate a 'flat' outline vs whorls when large spiral angle (eg k = 80). Spiral angles in degrees are converted to radians. A summary is printed on the console.

Author(s)

A.E. Aldridge

References

Aldridge (1999) Brachiopod outline and episodic growth. Paleobiology 25, 471-482

See Also

Giant squid data for example of small spiral angle and vignettes on generating and fitting linear spirals.

Examples

```
## Not run:
doSetupGraphics()
#
# One linear (L1) logarithmic spiral with default of 200 near equally spaced coordinates
#
 z1 <- generateAnyLinear()
#
# Two linear (L2) connected spirals with default 200 points, spiral change 100 coord from umbo
# first spiral angle is 30 degrees, second is 40 degrees
#
  z2 <- generateAnyLinear( k1=30, k2=30, k3=40, k4=40,  noise=0.001)
#
dev.set(3)  # plot on device 3 the angle about axis and log of radius
plot( z2$angle, z2$logr) # a broken stick line
dev.set(4)
plot (z2$x, z2$y,pch=16, cex=0.3) # the spiral outline
#
# Three linear spirals (L3)  default 200 points,changes at 100,150 coordinates from umbo
#
z3 <- generateAnyLinear( k1=35,k2=35,k3=40,k4=55, dist.zero=5)
#
# Four linear spirals (L3)  default 200 points, changes at 50, 100,150 coorinate from umbo
#
z4 <- generateAnyLinear(  k1=35,k2=40,k3=45,k4=55)
#

## End(Not run)
```

`generateBentCable2LinearSpiral`

generates a linear to generalised bent cable spiral

Description

generates a L2QL (LLQL)curved spiral with the LQL a generalised bent cable

Usage

```
generateBentCable2LinearSpiral(nstart = 200, change = c(50, 150),
       k1 = 35, k2 = 40, k3 = 55, g = 0.3, kappa = 2, dist.zero = 2,
       noise = 1e-04, theta.start = 1e-04, theta.end = 1.25,
       equal = FALSE, equal.n = 200, move = TRUE, move.factor = 1000,
     plot.it = TRUE, include.title = TRUE, show.change = TRUE, print.summary = TRUE)
```

Arguments

nstart	number of outline points to generate (default 200)
change	where the first is the centre of the quadratic transition, and the second locates the change from bent cable to the final linear part with slope k3.
k1	spiral angle, or slope of first linear part, BEFORE the quadratic transition
k2	spiral angle or slope of second linear part, AFTER the quadratic transition

k3	spiral angle, or slope of last linear part AFTER bent cable (in degrees)
g	width of the transition about change location
kappa	curved shape of transition (kappa = 2 is quadratic)
dist.zero	distance to outline when angle is zero.
noise	standard deviation of error added to the distance from spiral axis
theta.start	angle about axis to first point on outline (a small number)
theta.end	angle at end of outline IN RADIANS (approximate half a circle)
equal	should outline coordinates be equally spaced (default is no)
equal.n	if equal, then how many coordinates (default same as nstart, 200)
move	translate outline so all coordinates positive
move.factor	how much to move the outline
plot.it	plot the outline (default is yes)
include.title	text for title for plot(default is blank)
show.change	show the change location on the plot
print.summary	print summary (default is yes)

Details

Generates a spiral that is curved and straight when using logarithmic radius, and angle coordinates.

Input are three slopes: first to second then the third (all in degrees that are converted to radians). The transition between first and second slopes is a curve, leading to the generalised bent cable terminolgy. Useful to plot logr and angle (see Value below) to judge the change in slopes as being reasonable for the spirals being studied.

Value

xy	data frame with spiral coordinates as x, y, logr, angle, and arc length

Warning

may need to trial and error to match a specific outline

Note

A shape that is often detected through the shape of deviations in a two spiral (L2) fit of an outline.

Author(s)

A.E. Aldridge

References

Aldridge (1999) Brachiopod outline and episodic growth. Paleobiology 25, 471-482

Chiu, G., Lockhart, R. and Routledge, R. (2006), Bent-Cable Regression Theory and Applications, Journal of the American Statistical Association, 101, 542-553. DOI: 10.1198/016214505000001177 http://faculty.washington.edu/gchiu/Articles/bentcable-jasa.pdf

Khan, S.A. and Kar, S.C. (2018). Generalized bent-cable methodology for changepoint data: a Bayesian approach. Journal of Applied Statistics, 45(10), 1799-1822

See Also

fitAnyLinear using up to four spiral angles to match the quadratic transition.

generateBentCable generateAnyLinear fitLinear2BentCable fitBentCable2Linear

Examples

```
## Not run:
require(MASS)
doSetupGraphics()
z <- generateLinear2BentCable() # all the default settings are applied
xs <- fitLinear2BentCable( x=z$x, y=z$y, m=c(30,150), start.u=1000, start.v=1000)

## End(Not run)
```

---

generateBentCableSpiral

generates a generalised bent cable (LQL) logarithmic spiral

Description

generates a generalised bent cable logarithmic spiral

Usage

```
generateBentCableSpiral(nstart = 200, change = 100, k1 = 35, k2 = 45, g = 0.3, kappa = 2,
                dist.zero = 2, noise = 0.01, theta.start = 1e-04, theta.end = 1.25,
                   equal = FALSE, equal.n = 200, move = TRUE, move.factor = 1000,
                      plot.it = TRUE, include.title = TRUE, show.change = TRUE)
```

Arguments

nstart	number of outline points to generate
change	location of change (tau)
k1	first expansion angle
k2	second expansion angle
g	related to width of curved transition
kappa	power of curve (kappa = 2 is quadratic)
dist.zero	distance between axis and first outline coordinate (angle = 0)
noise	noise level (standard deviation in coordinates)
theta.start	initial angle at start of outline (a small number)
theta.end	final angle at end of outline
equal	create equally spaced outline coordinates
equal.n	number of equally spaced points
move	move the outline
move.factor	how much to move or translate the outline
plot.it	plot the outline or not
include.title	text for title (default is blank)
show.change	location the change

Details

Generates an outline (x,y) coordinates based on the generalised bent cable of Khan & Kar (2018).

Specifying kappa = 2 as input forces the bent cable to have the quadratic transition of Chiu et al (2006).

The transition location (angle in radian units) is specified as the parameter tau.

Value

xy	data frame with spiral coordinates as x, y, logr, angle, and arc length

Warning

plotting (TRUE) is required for obtaining output data frame

Note

g is gamma in the Chiu (2006) and Khan & Kar (2018). Parameter a2 is estimated by direct calculation.

Author(s)

A.E. Aldridge

References

Chiu, G., Lockhart, R. and Routledge, R. (2006), Bent-Cable Regression Theory and Applications, Journal of the American Statistical Association, 101, 542-553. DOI: 10.1198/016214505000001177 http://faculty.washington.edu/gchiu/Articles/bentcable-jasa.pdf

Khan, S.A. and Kar, S.C. (2018). Generalized bent-cable methodology for changepoint data: a Bayesian approach. Journal of Applied Statistics, 45(10), 1799-1822

See Also

generateL2BentCableSpiral

Examples

```
## Not run:
 z <- generateBentCableExtra()


## End(Not run)
```

`generateCurvedSpiral` *generates a curved spiral*

Description

generates a continuous curved (nonlinear) spiral up to the fourth power(quartic) of angle about spiral axis.

Usage

```
generateCurvedSpiral(nstart = 200, dist.zero = 2, k = 50,
           quadratic = 0.03, cubic = 0.03, quartic = 0.01,
          noise = 1e-05, theta.start = 1e-04, theta.end = 1.25,
          equal = FALSE, equal.n = 200, move = TRUE, move.factor = 1000,
          plot.it = TRUE, print.summary = TRUE, show.change = TRUE)
```

Arguments

`nstart`	number of outline points to generate (default 200)
`dist.zero`	initial distance between spiral axis and spiral when angle (theta) is zero
`k`	spiral angle or first power of angle about axis, units are degrees.
`quadratic`	second power of angle about axis
`cubic`	third power of angle about axis
`quartic`	fourth power of angle about axis
`noise`	standard deviation of error added to the distance from spiral axis
`theta.start`	angle about axis to first point on outline (a small number)
`theta.end`	angle at end of outline IN RADIANS (approximate half a circle)
`equal`	should outline coordinates be equally spaced (default is no)
`equal.n`	if equal, then how many coordinates (default same as nstart, 200)
`move`	translate outline so all coordinates positive
`move.factor`	how much to move the outline
`plot.it`	plot the outline (default is yes)
`print.summary`	print summary (default is yes)
`show.change`	show the change on the plot

Details

Generates a spiral that is curved when using logarithmic radius, and angle coordinates. The input quadratic, cubic and quartic terms can be negative, positive or zero. Spiral angle (first power, slope term) is input as degrees, then converted to radians for calculations.

For a quadratic spiral set to zero the cubic and quartic terms.

Value

`xy`	data frame with spiral coordinates as x, y, logr, angle, and arc length

Warning

may need to trial and error to find reasonable shape

Note

Small changes in curve's input values can create large changes in spiral, and length of the outline. Creates a 'flat' looking spiral with axis very near the umbo.

Default is NOT to equally space the generated coordinates. Equal spacing introduces small errors that can affect confirmation fitting.

Author(s)

A.E. Aldridge

References

no references yet

See Also

`fitQuadratic fitCubic fitQuartic`

Examples

```
## Not run:

doSetupGraphics()
# Quadratic spiral
 zq <- generateCurvedSpiral( quadratic=0.03, cubic=0, quartic=0)
dev.set(4)
plot( zq$angle, zq$logr) # upwards quadratic curve
#
# Cubic Spiral
 zc <- generateCurvedSpiral( quadratic=0.03, cubic=0.03, quartic=0)
dev.set(4)
plot( zc$angle, zc$logr) # upwards quadratic curve
#
# only the cubic term is specified.
 zc <- generateCurvedSpiral( quadratic=0.0, cubic=0.03, quartic=0)
#
# Quartic spiral
#
 zqq <- generateCurvedSpiral( quadratic= 0.01, cubic= -0.06, quartic=0.01)
dev.set(4)
plot( zqq$angle, zqq$logr) # sinuous log. radius
# flatter spiral
zqqq <- generateCurvedSpiral( quadratic= 0.01, cubic= -0.02, quartic=0.02)
dev.set(5)
plot( zqqq$angle, zqqq$logr) # sinuous log. radius
#

## End(Not run)
```

`generateLinear2BentCableSpiral` *generates a linear to generalised bent cable spiral*

Description

generates L2QL (LLQL)curved spiral with the LQL a generalised bent cable

Usage

```
generateLinear2BentCableSpiral(nstart = 200, change = c(30, 150),
      k0 = 35, k1 = 40, k2 = 55, g = 0.3, kappa = 2, dist.zero = 2,
      noise = 1e-04, theta.start = 1e-04, theta.end = 1.25,
      equal = FALSE, equal.n = 200, move = TRUE, move.factor = 1000,
     plot.it = TRUE, include.title = TRUE, show.change = TRUE, print.summary = TRUE)
```

Arguments

`nstart`	number of outline points to generate (default 200)
`change`	where the first linear part changes to bent cable and the centre of the quadratic transition.
`k0`	spiral angle, or slope of first linear part (in degress)
`k1`	spiral angle, or slope of second linear part, BEFORE the quadratic transition
`k2`	spiral angle or slope of last linear part, AFTER the quadratic transition
`g`	width of the transition about change location
`kappa`	curved shape of transition (kappa = 2 is quadratic)
`dist.zero`	distance to outline when angle is zero.
`noise`	standard deviation of error added to the distance from spiral axis
`theta.start`	angle about axis to first point on outline (a small number)
`theta.end`	angle at end of outline IN RADIANS (approximate half a circle)
`equal`	should outline coordinates be equally spaced (default is no)
`equal.n`	if equal, then how many coordinates (default same as nstart, 200)
`move`	translate outline so all coordinates positive
`move.factor`	how much to move the outline
`plot.it`	plot the outline (default is yes)
`include.title`	text for title for plot(default is blank)
`show.change`	show the change location on the plot
`print.summary`	print summary (default is yes)

Details

Generates a spiral that is both straight and curved when using logarithmic radius, and angle coordinates.

Input are three slopes: first to second then the third (all in degrees that are converted to radians). The transition between second and third slopes is a curve, leading to the generalised bent cable terminolgy. Useful to plot logr and angle (see Value below) to judge the change in slopes as being reasonable for the spirals being studied.

Value

xy data frame with spiral coordinates as x, y, logr, angle, and arc length

Warning

may need to trial and error to match a specific outline

Note

A shape that is often detected through the shape of deviations in a two spiral (L2) fit of an outline.

Author(s)

A.E. Aldridge

References

Aldridge (1999) Brachiopod outline and episodic growth. Paleobiology 25, 471-482

Chiu, G., Lockhart, R. and Routledge, R. (2006), Bent-Cable Regression Theory and Applications, Journal of the American Statistical Association, 101, 542-553. DOI: 10.1198/016214505000001177 http://faculty.washington.edu/gchiu/Articles/bentcable-jasa.pdf

Khan, S.A. and Kar, S.C. (2018). Generalized bent-cable methodology for changepoint data: a Bayesian approach. Journal of Applied Statistics, 45(10), 1799-1822

See Also

`fitAnyLinear` using up to four spiral angles to match the quadratic transition.

`generateBentCable` `generateAnyLinear` `fitLinear2BentCable`

Examples

```
## Not run:
require(MASS)
doSetupGraphics()
z <- generateLinear2BentCable() # all the default settings are applied
xs <- fitLinear2BentCable( x=z$x, y=z$y, m=c(30,150), axis.start=c(1000,1000))

## End(Not run)
```

---

generateLinearQuadraticSpiral

generates a linear connected to quadratic spiral (LQ)

Description

generates a connected linear then quadratic (LQ) spiral with a given change location

Usage

```
generateLinearQuadraticSpiral(nstart = 200, change = 100, k1 = 40, k2 = 50, q = 0,
               dist.zero = 2, noise = 0.01, theta.start = 0.001, theta.end = 1.25,
                  equal = FALSE, equal.n = 200,
              move = TRUE, move.factor = 1000, plot.it = TRUE, show.change = TRUE)
```

Arguments

nstart	number of outline points to generate
change	location of change (tau)
k1	first expansion angle
k2	second expansion angle
q	quadratic expansion value
dist.zero	distance between axis and first outline coordinate (angle = 0)
noise	noise level (standard deviation in coordinates)
theta.start	initial angle at start of outline (a small number)
theta.end	final angle at end of outline
equal	create equally spaced outline coordinates
equal.n	number o equally spaced points
move	move the outline
move.factor	how much to move or translate the outline
plot.it	plot the outline or not
show.change	location the change

Details

Method is not arc length based so NOT equally spaced by default. If checking a fit using `fitLinearQuadratic` do not use equally spaced values from this function.

Value

data frame with spiral coordinates as x, y, logr, angle, and arc length

Note

There is redundancy in specified parameter values

Author(s)

A.E. Aldridge

References

no reference yet

See Also

quadratic then linear spirals : `generateQuadraticLinearSpiral`

Examples

```
## Not run:
z <- generateLinearQuadraticSpiral(k2=0,q= -0.5, equal=F )
xs <- fitLinearQuadratic( x=z$x, y=z$y, m=100,axis.start=c(990,990))
#
z <- generateLinearQuadraticSpiral( dist.zero=5,k1=40,k2=55, equal=F  )
xsq <- fitLinearQuadratic( x=z$x, y=z$y, m=50, axis.start=c(990,990))
xsq$parameters1


## End(Not run)
```

`generateQuadraticLinearSpiral` *generate quadratic connected to a linear spiral (QL)*

Description

generate Quadratic connected to Linear spiral (QL) with given change location.

Usage

```
generateQuadraticLinearSpiral(nstart = 200, change = 100, k1 = 40, k2 = 50, q = 0,
dist.zero = 2, noise = 0.001, theta.start = 1e-04, theta.end = 1.25,
        equal = FALSE, equal.n = 200, move = TRUE, move.factor = 1000,
         plot.it = TRUE, show.change = TRUE)
```

Arguments

`nstart`	number of outline points to generate
`change`	location of change (tau)
`k1`	first expansion angle
`k2`	second expansion angle
`q`	quadratic factor
`dist.zero`	distance between axis and first outline coordinate (angle = 0)
`noise`	noise level (standard deviation in coordinates)
`theta.start`	initial angle at start of outline (a small number)
`theta.end`	final angle at end of outline
`equal`	create equally spaced outline coordinates
`equal.n`	number o equally spaced points
`move`	move the outline
`move.factor`	how much to move or translate the outline
`plot.it`	plot the outline or not
`show.change`	location the change

Details

no details yet

Value

data frame with spiral coordinates as x, y, logr, angle, and arc length

Note

not generated by equal arc lengths

Author(s)

A.E. Aldridge

References

no reference yet

See Also

generateAnyLinearSpirals in logspiral package

Examples

```
## Not run:
 z <- generateQuadraticLinearSpiral(k2=0,q= -0.5, equal=F )
xs <- fitQuadraticLinear( x=z$x, y=z$y, m=100, axis.start=c(990,990))
xs$parameters1
xs$parameters2

## End(Not run)
```

---

generateQuadraticSpirals

generates TWO quadratic spirals (Q2) with given change location

---

Description

generate TWO connected quadratic spirals with the given change location

Usage

```
generateQuadraticSpirals(nstart = 200, change = 100, k1 = 40, k2 = 50, q1 = 0.03,
            dist.zero = 2, noise = 0.001, theta.start = 1e-04, theta.end = 1.25,
              equal = TRUE, equal.n = 200, move = TRUE, move.factor = 1000,
              plot.it = TRUE, show.change = TRUE)
```

Arguments

nstart	number of outline points to generate
change	location of change (tau)
k1	first expansion angle
k2	second expansion angle
q1	first quadratic term (second is calculated using continuity)

dist.zero	distance between axis and first outline coordinate (angle = 0)
noise	noise level (standard deviation in coordinates)
theta.start	initial angle at start of outline (a small number)
theta.end	final angle at end of outline
equal	create equally spaced outline coordinates
equal.n	number o equally spaced points
move	move the outline
move.factor	how much to move or translate the outline
plot.it	plot the outline or not
show.change	location the change is included

Details

To generate one quadratic allow both linear spiral expansions to be equal (see example) The spiral axis is very close to first outline point (umbo).

Continuity conditions put severe limitations on the shape of the second quadratic.

This double quadratic is a NOT one of the standard spirals and has dubious value. McGhee (1980) has a quadratic formulation that may be better suited (see curved logarithmic spiral and fitQuadratic).

Value

data frame with spiral coordinates as x, y, logr, angle, and arc length

Note

not generated by equal arc lengths.

Author(s)

A.E. Aldridge

References

Spiral notebook (page 10B) for proof and the exponential formulation.

McGhee, G.R., 1980. Shell form in the biconvex articulate Brachiopoda: A geometric analysis. Paleobiology 6: 57-76.

See Also

generateAnyLinearSpirals generateCurvedSpiral

Examples

```
## Not run:
# ONE QUADRATIC SPIRAL using this function instead of generateCurvedSpiral
z <- generateQuadraticSpirals(k1=45,k2=45,q=0.03, dist.zero=2,theta.end=0.8,equal=F)
xsq <- fitOneQuadratic( x.df$x, x.df$y, axis.start=c(980,1000))
xsq$parameters1
xsq$parameters2
#
```

```
# TWO CONNECTED QUADRATIC SPIRALS ( if it makes sense or is even possible)
 z <- generateQuadraticSpirals(k1=45,k2=50, dist.zero=2,theta.end=0.8,equal=F)
xs <- fitTwoQuadratic( x=z$x, y=z$y, m=100, axis.start=c(980,1000))
## WARNING - THIS NOT WORKING WELL in the fitting (it is ok with generating)
## problem being that connection requires careful choice of input & start values


## End(Not run)
```

`partitionSumSquares` *Partitions the variation in a sequence of N equally spaced spiral deviations*

Description

Partitions the variation of spiral deviations for each of N/2 Fourier frequencies

Usage

```
partitionSumSquares(spiral.input = spiral.input, plot.it = T, print.output = T)
```

Arguments

`spiral.input`	results from a spiral fit are used as input
`plot.it`	should variation from Fourier frequencies be plottted (default is yes).
`print.output`	should variation from Fourier frequencies be printed (default is yes).

Details

Uses fast Fourier transform (fft in base R) to partition variance (power) at N/2 Fourier frequencies.

Deviations are equally spaced against arc length before applying fft.

The output value is sums of squares with a total that should closely match (when multiplied by N-1 and square root of the result)the standard deviation (sigma) of the spiral fit

Value

a vector of N/2 sums of squares of deviations about zero.

Note

Used in the function `compareSpiralFits`

Author(s)

A.E. Aldridge

References

no reference yet

See Also

`compareSpiralFits`

Examples

```
## Not run:
z <- generateAnyLinear()
xs <- fitAnyLinear(x=z$x, y=z$y)
xsum.squares <- partitionSumSquares(xs)

## End(Not run)
```

PeakCycle *finds maxima (minima) in a sequence of points*

Description

finds maxima in a timeseries

Usage

```
PeakCycle(Data = Data, SearchFrac = 0.02)
```

Arguments

Data	single vector of data values
SearchFrac	how much to searh around the suspected maximum

Details

see examples

Value

list of maxima

Note

depends on shape of peak whether it can be identified. Uses a fixed mother wavelet = Mexican hat from the double derivative of Gaussian shape.

Author(s)

William Constantine and Donald Percival for R package wmtsa

References

Detecting cycle maxima (peaks) in noisy time series (In R) - Stack Overflow 27/06/2013 09:24 Link to Scholkmann paper on "automatic multiscale-based peak detection" http://www.mdpi.com/1999-4893/5/4/588 "I've tried the "turningpoints" function in the "pastecs" package but it seemed to be too sensitive (i.e.,detected too many peaks)". by R. Batt

See Also

doLocateChange function that allows you to manually locate change on a graphics device.

Other ways of finding max and min using complete wavelet, empirical mode, or synchro-squeezed decompositions.

Examples

```
## Not run:
doSetupGraphics()
# create an outline with two spirals, change at the 100th coordinate
xy.df <- generateAnyLinear( k1=35, k2=35, k3=45, k4=45)
  # fit this outline with one spiral, manually guessing axis location
  # at the prompt on graphics device 2.
xys <- fitAnyLinear( xy.df$x, xy.df$y, plot.deviations=T)
  # find the peaks or maxima in following command
  # can reverse sign of the deviations to find minima
#
 max <- PeakCycle( Data=xys$deviations, SearchFrac=0.1)
 min <- PeakCycle( Data=-1*xys$deviations, SearchFrac=0.1)
## check by plotting
#
plot(xys$arc.length, xys$deviations,type="s")
 points(xys$arc.length[max[,1]], xys$deviations[max[,1]], pch=16,cex=1,col=4)
 points(xys$arc.length[min[,1]], xys$deviations[min[,1]], pch=15,cex=1,col=2)



## End(Not run)
```

`plotDiagnostics` *Four diagnostic plots from a spiral fit*

Description

Produces four plots in one graphic window:

Spiral deviations against distance from umbo

Log distance from spiral axis against angle about axis

Normal probability plot of deviations

Cumulative periodogram of deviations in sequence from umbo

Usage

```
plotDiagnostics(spiral.output = spiral.output, change = c(NULL, NULL, NULL),
                output.device = 5)
```

Arguments

`spiral.output`	output from the spiral fitting (a list)
`change`	locations (coordinate sequence from umbo) of change points
`output.device`	graphic window for diagnostic output (default is device 5)

Details

May need to create graphic windows by using `dev.new()`

`cpgram` is from MASS package, using Fourier frequencies with a default light taper.

A warning is given if there are extreme deviations (probability quantile = 0.95) at or near (within 4 coordinate pairs) of a change location.

Value

2 x 2 graphics layout on the one window.

Note

see documentation in MASS for cpgram: checks for randomness of deviations, percentage of total deviation variance can be allocated to various frequencies

Author(s)

A.E. Aldridge

References

Venables, W. N. & Ripley, B. D. (2002) Modern Applied Statistics with S. Fourth Edition. Springer, New York. ISBN 0-387-95457-0

See Also

`fitAnyLinear plotSpiralFit plotSpiralDeviations`

Examples

```
## Not run:
z <- generateAnyLinear()
xs <- fitAnyLinear( x = z$x, y=z$y, plot.diagnostics=T)

## End(Not run)
```

---

plotSpiralDeviations *plot deviations from a spiral fit*

Description

plots deviations from spiral fit output

Usage

```
plotSpiralDeviations(spiral.output = spiral.output,
change = c(NULL, NULL, NULL),
title.label = "", output.device = 4, units="mm")
```

Arguments

`spiral.output`	the fitted spiral
`change`	location of changes in spiral (if any)
`title.label`	title label, usually a specimen name or code
`output.device`	which device to plot
`units`	units for plot label (default is mm)

Details

note the graphic set up of windows

Value

plots without any values returned. Called by `fitAnyLinear` and the other fitting functions

Author(s)

A.E. Aldridge

See Also

`fitAnyLinear`

Examples

```
##none as internal to fitAnyLinear and other fitting functions
```

---

plotSpiralFit *plot results of fitting spiral(s)*

Description

plots fitted spiral or spirals with axis location

Usage

```
plotSpiralFit(spiral.output = spiral.output,
change = c(NULL, NULL, NULL), xdata = xdata, ydata = ydata,
title.label = " ", output.device = 5)
```

Arguments

`spiral.output`	the fitted spiral
`change`	location of changes in spiral (if any)
`xdata`	x coords used in fit
`ydata`	y coords used in fit
`title.label`	title label
`output.device`	which device to plot

Details

called by fitAnyLinear and some other fits. Note the graphic set up with four windows as device numbers 2, 3, 4, and 5.

Value

plots without any values returned

Author(s)

A.E. Aldridge

See Also

`fitAnyLinear`

Examples

```
##see fitAnyLinear

#
```

`rotate2D` *rotate 2D coordinates CLOCKWISE about a given location.*

Description

rotate 2D coordinates clockwise

Usage

```
rotate2D(x, y, theta, u = 0, v = 0)
```

Arguments

`x`	x coord
`y`	y coord
`theta`	angle to rotate (radians)
`u`	x coord about which to rotate
`v`	y coord about which to rotate

Details

theta is in radian units, input theta as `(pi/180)*angle` if angle is in degree units

Value

rotated coordinates returned as a data frame

Author(s)

A.E. Aldridge

References

Maxwell EA (1958) Elementary Coordinate Geometry

Examples

```
##data(Laqueus)

## The function is currently defined as
function (x, y, theta, u = 0, v = 0)
{
# theta in radians
    xx <- x - u
    yy <- y - v
    xp <- xx * cos(theta) + yy * sin(theta)
    yp <- -xx * sin(theta) + yy * cos(theta)
    output <- data.frame(x = xp + u, y = yp + v)
    return(output)
}
```

StartAngle *find the start angle of a sequence of outline points*

Description

find start angle of outline relative to given point (axis) u, v

Usage

```
StartAngle(xc, yc, u1, v1)
```

Arguments

xc	x coordinate sequence
yc	y coordinate sequence
u1	x coordinate of fixed point
v1	y coordinate of fixed point

Details

This function can be streamlined by using `atan2` and one `ifelse`

```
angle <-atan2( yc-v1,xc-u1)
angle <-ifelse( angle < 0,angle + 2*pi,angle)
```

Value

start.theta	starting angle in radians

Author(s)

A.E. Aldridge

Examples

```
##data( GiantSquidBeak)

## The function is currently defined as
function (xc, yc, u1, v1)
{
    start.theta <- atan(abs((yc - v1))/abs(xc - u1))
    if (xc - u1 < 0 && yc - v1 > 0)
        start.theta <- pi - start.theta
    else if (xc - u1 < 0 && yc - v1 < 0)
        start.theta <- pi + start.theta
    else if (xc - u1 > 0 && yc - v1 < 0)
        start.theta <- 2 * pi - start.theta
    return(start.theta)
  }
```

`verifyL3Change` *verify the TWO change locations in a three spiral fit*

Description

verify the TWO change locations in a three spiral fit

Usage

```
verifyL3Change(x = x, y = y, m = c(NULL, NULL), pp = c(-2, -1, 0, 1, 2),
                axis.start = c(NULL, NULL))
```

Arguments

`x`	x coord
`y`	y coord
`m`	specified location of TWO change points on an outline
`pp`	sequence of coordinate points either side of specified change points
`axis.start`	start in x, y

Details

Points either side are as above or a vector such as seq(-2,2,1). Location of minimum for the deviation rss is output. If not the same as for the three spiral fit, then use best change points for the final three spiral fit. Output is rss = residual sums of squares from the spiral fit.

Automatic use of the spiral checking functions may not provide the global minima for change locations.

BEWARE when really need two criteria - not only minimal RSS, but also maximum biological signal (e.g. periodicities).

Value

`rss.mat`	matrix of deviation rss's, with rows the first change location

Note

The matrix can be input to code for generating a contour plot (eg using lattice package)

Author(s)

A.E. Aldridge

References

Aldridge (1999) Brachiopod outline and episodic growth. Paleobiology 25, 471-482

See Also

`fitAnyLinear fitL3Spiral`

Examples

```
## Not run:
require(MASS)
doSetupGraphics()
x.df <- generateAnyLinear( k1=30, k2=30, k3=50, k4=40, noise=0.1)
xx <- x.df$x
yy <- x.df$y
## created three spirals with two change points at 100, 150 points from
## outline start or umbo
#
xspiral <- fitAnyLinear(xx, yy,m=c(99,149), axis.start=c(110, 100))
## fits three spirals varying change locations about 99 and 149 (true values
 # are 100 and 150. If axis.start is not specified, you will be
 # asked to input a location off graphics device 2.
#
xyz <- verifyL3Change(x = xx, y = yy, m = c(99,149), pp = c(-2, -1, 0, 1, 2))
 # xyz is a matrix with rows for first change, and columns for second change
 # the location of minimum is also output, which looking at the matrix should
 # be a 100 and 150.
 #
## also create a contour plot using lattice, but first need vectors
x <- rep( as.numeric( rownames(xyz) ),length(rownames(xyz))
y <- rep( as.numeric( colnames(xyz) ),each= length(colnames(xyz) ) )
z <- as.vector(xyz)
#
lattice::contourplot( z ~ x*y, cuts=10,
xlab="First change location",ylab="Second change location")

## End(Not run)
```

Index