Package: paleoAM (via r-universe)

November 9, 2024

Type Package

Title Simulating Assemblage Models of Abundance for the Fossil Record

Version 1.0.1

Date 2024-09-16

Author David W Bapst [aut, cre], Christina L Belanger [aut]

Maintainer David W Bapst <dwbapst@gmail.com>

BugReports https://github.com/dwbapst/paleoAM/issues

Depends R (>= 4.1.0)

Imports graphics, stats, vegan

Suggests spelling, testthat

License CC0

URL https://github.com/dwbapst/paleoAM

Language en-US

Encoding UTF-8

ByteCompile TRUE

Description Provides functions for fitting abundance distributions over environmental gradients to the species in ecological communities, and tools for simulating the fossil assemblages from those abundance models for such communities, as well as simulating assemblages across various patterns of sedimentary history and sampling. These tools are for particular use with fossil records with detailed age models and abundance distributions used for calculating environmental gradients from ordinations or other indices based on fossil assemblages.

RoxygenNote 7.3.2

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

RemoteUrl https://github.com/dwbapst/paleoam

RemoteRef HEAD

RemoteSha c65529ee5df19565bf9f8a1dbc697046db54270e

Contents

calculateImplicitParameters . . . 2
getProbOccViaPresAbs . . . 4
getRecoveredTransitionDuration . . . 6
getSampleDCA . . . 7
getSpeciesSpecificRescaledKDE . . . 8
getTimestepAbundances . . . 10
gulfOfAlaska . . . 11
hirnantian . . . 12
plotFossilAssemblageSeriesDCA . . . 13
plotGradientKDE . . . 14
plotHeatmapComparison . . . 16
sampleFossilSeries . . . 18
setupSimulatedGradientChange . . . 19
simMixedAssemblageSample . . . 21
simulateFossilAssemblageSeries . . . 22
simulateGradientQuantile . . . 26

Index **28**

`calculateImplicitParameters` *Calculate Implicit Parameters for Modeling Time-Series of Fossil Assemblages*

Description

Given a sufficient set of parameters for simulating fossil assemblages as a time-series, this function calculates the full set of parameters necessary for running each component model.

Usage

```
calculateImplicitParameters(
  eventChangeScale,
  bgGradientValue,
  fullGradientRange,
  eventSampleWidthRatio = NULL,
  sampleWidth = NULL,
  eventDuration = NULL,
  sedRatePerTimestep = NULL,
  maxSampleTimeStep = 500,
  minSampleTimeStep = 3,
  samplingCompleteness,
  transitionDurationRatio,
  bioturbDepthRatio
)
```

Arguments

`eventChangeScale`
: A value indicating the amount relative to the background value (`bgGradientValue`) and the maximum possible change as indicated by `fullGradientRange` (in other words, simulated change must be within observed gradient, so `eventChangeScale` is a proportional multiplier of the total possible change).

`bgGradientValue`
: The gradient value expected during background intervals during which no notable excursion is occurring on that environmental gradient.

`fullGradientRange`
: A vector of two values giving the minimum and maximum gradient values observed in the empirical data.

`eventSampleWidthRatio`
: How long should an event be relative to the amount of time (or sediment) captured within a sedimentary sample? This parameter is used for simulating event duration, sample width and sedimentation rate where any two of these three are defined and the third is not defined. This value is referred to as *Resolution Potential* in Belanger & Bapst (2023).

`sampleWidth`
: The 'width' of a sample relative to core depth or outcrop height, usually given in linear units (usually centimeters). For taking sediment samples from a core, this is straightforward (how thick is each sediment sample taken?) but for outcrops this may be more difficult to determine (what is the thickness of a horizon in a shale unit?).

`eventDuration`
: The duration (in time-units) of a simulated event during which the environmental gradient is at an excursion 'peak' level.

`sedRatePerTimestep`
: The rate of sedimentation, given as a ratio of sediment thickness (given in linear dimensions, in the same units as `sampleWidth`), over time (given in the same time units as `eventDuration`.

`maxSampleTimeStep`
: The maximum number of individual time-steps used for simulating a sample.

`minSampleTimeStep`
: The minimum number of individual time-steps used for simulating a sample.

`samplingCompleteness`
: The relative completeness of stratigraphic sampling. For example, if two-centimeter wide samples of sediment are taken from a sediment core, every ten centimeters, then the `samplingCompleteness` is two over 10, or that `samplingCompleteness = 1/5`. A simulation with a sampling completeness of 1 would be comparable to exhaustively sampling a core that recorded no gaps in sedimentation over its history. Rocky outcrops are more complicated, as fossil-bearing horizons may be relatively thin compared to the thickness of the section, such that outcrop-based fossil records should be simulated as having *very low* `samplingCompleteness`.

`transitionDurationRatio`
: The ratio of how long the transition between peak and background intervals should be, relative to the length of the peak 'event' duration (`eventDuration`). The longer this transition interval, the more chances of an assemblage being sampled that represents transitional gradient values.

`bioturbDepthRatio`
: The ratio of the sediment depth to which bioturbation occurs, made relative to the width of a sediment sample (`sampleWidth`). A `sampleWidth` of 3 cm and a `biotubDepthRatio` of 5 implies a bioturbation depth of 15 cm (`3 * 5`). Bioturbation depth varies considerably in the modern ocean, but is often the depth of active bioturbation is about 10 cm, such that the top ten centimeters of sediment (and the organic remains in those ten centimeters of sediment) are being regularly moved up and down by organism activity. For the purposes of this model, a bioturbation zone depth of 10 centimeters means that sampling a centimeter of sediment at location X, the apparent fossil assemblage that would be recovered is just as likely to include specimens that were deposited five centimeters away as those deposited at location X.

Details

Under the models considered in `paleoAM`, some parameterizations may be equivalent, even though the a particular analysis might be better simulated using a particular set of parameters. Allowing various different parameterizations is a useful generalization, but requires translating those equivalent parameters from one set to another (e.g. specifying parameters A, C & D, but running a simulation that requires parameters A, B & C).

This function mainly exists to calculate unspecified parameters for `simulateFossilAssemblageSeries` and also to identify conflicting parameter specifications.

Value

Returns a list giving the full set of parameters necessary for running `simulateFossilAssemblageSeries`.

References

Belanger, Christina L., and David W. Bapst. 2023. "Simulating our ability to accurately detect abrupt changes in assemblage-based paleoenvironmental proxies." Palaeontologia Electronica 26 (2), 1-32

See Also

`simulateFossilAssemblageSeries`

`getProbOccViaPresAbs` *Estimate the Per-Species Probability of Occurrence as a Function of an Environmental Gradient from Presence-Absence Data*

Description

This function calculates how species presence (not abundance) varies as a function of an underlying environmental gradient, using only the pattern of presence-absence observed in a given data set (usually data on species abundance).

Usage

```
getProbOccViaPresAbs(
  origAbundData,
  gradientOrigDCA,
  occurrenceFloor = 0,
  nBreaksGradientHist = 20
)
```

Arguments

origAbundData	The abundance data of the data you wish to model the abundance of.
gradientOrigDCA	The environmental gradient along which abundance varies, which you are fitting a KDE to.
occurrenceFloor	The minimum occurrence for every species, in every bin. The default is zero – increasing this value means every species has a non-zero chance of occurring in every bin, which tends to result in wildly more diverse assemblages than what is observed in the fossil record, so use with caution.
nBreaksGradientHist	The default is 20. Twenty what they asked? Twenty something.

Details

The rationale for this function was that simulating assemblages using the KDEs of species abundance alone (calculated with `getSpeciesSpecificRescaledKDE` tended to create very diversity-high assemblages that did not reflect reality. Accounting for species presence at all using separate models brings simulated assemblages much closer to real assemblages.

Value

An approximate function, created with `approx` that describes the relationship between gradient and probability of occurrence for all species.

See Also

`getSpeciesSpecificRescaledKDE`

Examples

```
# load data
data(gulfOfAlaska)

alaskaProbOccur <- getProbOccViaPresAbs(
   gradientOrigDCA = DCA1_GOA,
   origAbundData = abundData_GOA
   )
```

`getRecoveredTransitionDuration` *Measure the Duration of a Transition Period from Recovered Sequence of Fossil Assemblages with DCA-1 Scores*

Description

How long did a transition proceed from background to peak 'event' v

Usage

```
getRecoveredTransitionDuration(
  simRecord,
  bgUpperEnvelope,
  eventLowerEnvelope = NULL,
  returnAsAge = FALSE,
  trueEventDuration = NA,
  plot = FALSE
)
```

Arguments

`simRecord`	A simulated fossil record with assemblage change across multiple time-steps, with sedimentary thickness modeled.
`bgUpperEnvelope`	The upper envelope on what is considered a background value for a gradient value derived from the assemblage.
`eventLowerEnvelope`	The lower envelope on what is considered an event value for a gradient value derived from the assemblage.
`returnAsAge`	Should the estimated duration of the transition be returned as a duration in time-units? If `FALSE` (the default), the value is instead returned as a ratio relative to the true event duration.
`trueEventDuration`	The true duration of the event. This must be provided by the user if `returnAsAge = TRUE` to calculate the duration of the transition interval in simulation time-units.
`plot`	Should the data be plotted with the estimated transition interval on it, for visual checking?

Details

The envelope values can be calculated different ways, or even picked arbitrarily by the user. For example, `bgUpperEnvelope` is the upper envelope on what is considered a background value for a gradient value derived from the assemblage (for example, an ordination score). One way a user could calculate `bgUpperEnvelope` would be to repeatedly simulate assemblages at the background

value, calculate their apparent gradient value and estimate a 0.95 or 0.975 quantile. This can be done easily with function `simulateGradientQuantile`.

Value

A single value, reflecting (by default) a ratio of transition duration over the event duration. Can be modified with argument `returnAsAge`.

See Also

`simulateGradientQuantile`

`getSampleDCA` *Get the Detrended correspondence Analysis (DCA) Score Value for A Single Sample*

Description

For a single simulated assemblage sample, projects its location in DCA space defined by the original abundance data.

Usage

```
getSampleDCA(
  simSample,
  origAbundData,
  useTransformedRelAbundance = TRUE,
  projectIntoOrigDCA = TRUE,
  returnDCAforOrigAndSim = FALSE,
  whichAxes = 1,
  powerRootTransform = 1
)
```

Arguments

`simSample`	The assemblage data for a single sample (presumably from a simulation).
`origAbundData`	The original matrix of abundance data, to be used to project the simulated data into the same detrended correspondence analysis (DCA) space.
`useTransformedRelAbundance`	Should the DCA be analyzed
`projectIntoOrigDCA`	Should the new simulated data be projected in the DCA generated by analyzing the original data? This is `TRUE` by default, which is what most users will likely use, as it is the preferable way to consider how the new simulated data relates to the original data.

`returnDCAforOrigAndSim`
: Should the DCA score values for both the new simulated data and the original abundance data be returned? Default is `FALSE` as projecting the new data into the original DCA space means the original data should never be meaningfully different the original score values.

`whichAxes`
: Which dimensional score from the DCA should be used? By default this is 1. Unclear under what circumstances one would ever use a value other than 1, though, as detrending the correspondence analysis causes distortion along all scores other than the first axis of the ordination. Only the first score can be returned when `projectIntoOrigDCA = TRUE`.

`powerRootTransform`
: The power-root transform to be used on the abundance data before applying the DCA. By default this is 1, which means the data is not transformed at all. Note that the power-root transform is only performed if `useTransformedRelAbundance = TRUE`.

Details

Detrended correspondence analysis (DCA) is a common method for producing an ordination of ecological data. It isn't the only such method.

Value

A vector, containing either a single value, the DCA score value of the simulated sample, when `returnDCAforOrigAndSim = FALSE`, or a vector of the DCA scores for the original data and

See Also

This function is ultimately just a wrapper for using `decorana` in package `vegan`.

`getSpeciesSpecificRescaledKDE`

This is a function for Fitting a KDE to a specific species in Community Ecology Data

---

Description

This function fits a KDE to the abundance data of a particular species from community data given some ecological gradient variable.

Usage

```
getSpeciesSpecificRescaledKDE(
  gradientOrigDCA,
  origAbundData,
  abundanceFloorRatio = 0.5,
  nBreaksGradientHist = 20,
  modeledSiteAbundance = 10000
)
```

Arguments

`gradientOrigDCA`
: The environmental gradient along which abundance varies, which you are fitting a KDE to.

`origAbundData`
: The abundance data of the data you wish to model the abundance of.

`abundanceFloorRatio`
: The minimum value for the abundance in a given interval along the gradient – a probably arbitration value that is set to 0.5 by default.

`nBreaksGradientHist`
: The default is 20. Twenty what they asked? Twenty something.

`modeledSiteAbundance`
: The number of abundances the relative abundances will by multiplied by to formulate the KDE. The default is 10000.

Details

In many ways, this is an attempt to measure empirical representations of the abundance response curves relative to environmental gradients, as portrayed in figure within Patzkowsky & Holland (2012).

The ecological gradient variable is often an environmental gradient, such as depth, oxygenation, altitude, precipitation, but this is not necessarily so.

Value

A list containing the KDEs describing change in abundance for each species across the specified gradient.

References

Patzkowsky, M.E. and Holland, S.M., 2012. *Stratigraphic Paleobiology: Understanding the Distribution of Fossil Taxa in Time and Space.* University of Chicago Press. 259 pages.

See Also

`getProbOccViaPresAbs`, `plotGradientKDE`

Examples

```
# load data
data(gulfOfAlaska)

alaskaKDEs <- getSpeciesSpecificRescaledKDE(
    gradientOrigDCA = DCA1_GOA,
    origAbundData = abundData_GOA,
    abundanceFloorRatio = 0.5,
    nBreaksGradientHist = 20,
    modeledSiteAbundance = 10000
    )
```

```
plotGradientKDE(
    speciesKDEs = alaskaKDEs,
    fullGradientRange = c(min(DCA1_GOA), max(DCA1_GOA))
    )
```

`getTimestepAbundances` *Simulate Fossil Assemblages with Abundances at each Time-Step*

Description

Given a set of KDEs fit to species abundance and models of species occurrence relative to an environmental gradient, and given a sequence of gradient values, and a number of specimens to sample at each time-step, obtains a matrix containing abundances for species as a series of simulated assemblages.

Usage

```
getTimestepAbundances(
  kdeRescaled,
  probSpeciesOccur,
  gradientValues,
  specimensPerTimestep
)
```

Arguments

`kdeRescaled`	The list of modeled KDEs for species abundance, output from `getSpeciesSpecificRescaledKDE`.
`probSpeciesOccur`	The output from `getProbOccViaPresAbs`
`gradientValues`	A vector of gradient values to simulate over. A separate 'true' assemblage / community will be simulated for each value in the respective vector.
`specimensPerTimestep`	The number of specimens returned in a given time-step by `getTimestepAbundances`, usually set to an unrealistically high number to represent the true 'unsampled' fossil assemblage.

Details

`getTimestepAbundances` represents simulating the original biotic community that was present at some given point in time, which is not the same thing as a fossil assemblage that might be collected from sediments today as finite samples. That is covered by feeding the output from this function to `sampleFossilSeries`.

Thus, this function is generally run before running `sampleFossilSeries`, however most users will likely never run either function, instead running `simulateFossilAssemblageSeries`.

Value

A matrix containing abundances for species as a series of simulated assemblages.

See Also

This function is generally run before running `sampleFossilSeries`. Most users will likely never run either function, instead running `simulateFossilAssemblageSeries`.

`gulfOfAlaska` *Benthic Foram Abundances from the Gulf of Alaska Reported by Sharon et al 2021*

Description

Loads the foraminifera abundances collected and published by Sharon et al. (2021), as used in the simulation analyses published in Belanger and Bapst (2023).

Format

This data set is composed of three objects:

fullDataTable_GOA The full data table containing sample IDs, DCA-1 scores and species abundances.

DCA1_GOA A vector of just the DCA-1 scores for each sample.

abundData_GOA The number of specimens identified as a particular species for each sample.

Details

This data set contains the absolute abundances (number of specimens identified) for 48 species of benthic foraminifera, across 355 samples, taken from Sharon et al (2021). These samples were collected from the less than 63 micrometer size-fractions taken from Integrated Ocean Drilling Program Expedition 341 Site U141 and the co-located jumbo piston core, respectively located at 697 meters and 682 meters water depth in the Gulf of Alaska (Jaeger et al., 2014).

Source

Belanger, Christina L., and David W. Bapst. 2023. "Simulating our ability to accurately detect abrupt changes in assemblage-based paleoenvironmental proxies." Palaeontologia Electronica 26 (2), 1-32

Jaeger, J.M., Gulick, S.P.S., LeVay, L.J., Asahi, H., Bahlburg, H., Belanger, C.L., Berbel, G.B.B., Childress, L.B., Cowan, E.A., Drab, L., Forwick, M., Fukumura, A., Ge, S., Gupta, S.M., Kioka, A., Konno, S., März, C.E., Matsuzaki, K.M., McClymont, E.L., Mix, A.C., Moy, C.M., Müller, J., Nakamura, A., Ojima, T., Ridgway, K.D., Rodrigues Ribeiro, F., Romero, O.E., Slagle, A.L., Stoner, J.S., St-Onge, G., Suto, I., Walczak, M.H., and Worthington, L.L., 2014. Expedition 341 summary. In Jaeger, J.M., Gulick, S.P.S., LeVay, L.J., and the Expedition 341 Scientists, Proceedings of IODP, 341: College Station, TX (Integrated Ocean Drilling Program).

Sharon, Christina Belanger, Jianghui Du, and Alan Mix. "Reconstructing paleo-oxygenation for the last 54,000 years in the Gulf of Alaska using cross-validated benthic foraminiferal and geochemical records." Paleoceanography and Paleoclimatology 36, no. 2 (2021): e2020PA003986.

See Also

hirnantian

Examples

```
data(gulfOfAlaska)

########################
# (This is not to be run, just showing how data was loaded)
#
# # Loading the data files used by Belanger & Bapst 2023
#     # taken from Sharon et al. supplemental
# fullDataTable_GOA <- read.table(
#    "foram_abundances_forSimulations.txt",
#    header = TRUE)
#
# DCA1_GOA <- fullDataTable_GOA$DCA1
# abundData_GOA <- fullDataTable_GOA[,-(1:5)]
#
# save(fullDataTable_GOA, DCA1_GOA, abundData_GOA,
#    file = "data/gulfOfAlaska.Rdata")
```

---

hirnantian *Katian-Hirnantian Graptolite Assemblages Reported by Sheets et al. (2016)*

Description

This is a data set of relative abundances for planktonic graptolite species from two sections that cross the Katian-Hirnantian boundary in the late Paleozoic.

Format

This dataset is composed of two objects:

graptCommData A data table of sample IDs and relative species abundances.

graptSampleInfo A data table of additional locality and geologic age information related to each sample.

Details

This data set contains the relative abundances (proportion of specimens identified) for 43 species of planktonic graptolites, across 34 samples taken from two outcrops: (a) a section at Vininni Creek in Nevada, USA, and (b) a second section at Blackstone River in the Yukon, Canada. Both of these sections cross the Katian-Hirnantian boundary and the relative abundances reported here were provided as supplementary material with Sheets et al. (2016).

Source

Sheets, H. David, Charles E. Mitchell, Michael J. Melchin, Jason Loxton, Petr Štorch, Kristi L. Carlucci, and Andrew D. Hawkins. "Graptolite community responses to global climate change and the Late Ordovician mass extinction." Proceedings of the National Academy of Sciences 113, no. 30 (2016): 8380-8385.

See Also

gulfOfAlaska

Examples

```
data(hirnantian)

########################
# # (This is not to be run, just showing how data was loaded)
#
# # Sheets et al. community abundance data
# graptCommData <- read.csv(
#    "grapt_abundances_Sheets_et_al_Vinini&Blackstone_01-09-22.csv"
#    , row.names = 1, header = TRUE
#    )
#
# # sample specific info
# graptSampleInfo <- read.csv(
#    "grapt_siteData_SheetsEtAl.csv",
#    row.names = 1, header = TRUE,
#    stringsAsFactors = TRUE
#    )
#
# save(graptCommData, graptSampleInfo,
#    file = "data/hirnantian.Rdata")
```

---

plotFossilAssemblageSeriesDCA

Plot the Recovered DCA Values of a Fossil Assemblage Series

Description

Makes a plot of the simulated generating gradient over time along with the recovered gradient values in the same plot for a given simulated time series of fossil assemblages, particularly those from `simulateFossilAssemblageSeries`.

Usage

```
plotFossilAssemblageSeriesDCA(
  ...,
  colSimGenerating = "black",
  colSimRecovered = "navy"
)
```

Arguments

`...`	This function takes either the output from `simulateFossilAssemblageSeries` or requires specifying the three arguments `simTimeVar`, a data-frame of time-steps, sedimentary width and gradient values for a time-series simulation); `gradientRecovered`, the recovered gradient values; and `sampleAge`, the age of individual samples.
`colSimGenerating`	What color should be used for the generating ("true") gradient values?
`colSimRecovered`	What color should be used for the recovered gradient values calculated from the simulated data?

Details

The function will generally only be run on the output from `simulateFossilAssemblageSeries`, although the function is written so that the necessary elements can be provided separately.

Value

Returns nothing at all. Just a plot. That's all!

---

plotGradientKDE *Plot Kernel Density Estimates of Species Abundance Across a Focal Gradient*

Description

This function plots each rescaled KDE fit to each specific-specific rise-and-fall in abundance across some ecological gradient variable.

Usage

```
plotGradientKDE(
  speciesKDEs,
  fullGradientRange,
  xlim = NULL,
  ylim = c(0, 1),
  logY = FALSE
)
```

Arguments

`speciesKDEs`	A list of rescaled-KDE data, where each element is a different species, such as that output by `getSpeciesSpecificRescaledKDE`.
`fullGradientRange`	The minimum and maximum value of the ecological gradient variable at which ecological assemblage data was observed. If `xlim` isn't given, this defines the horizontal axis limits for resulting plot.
`xlim`, `ylim`	Vectors of two elements, defining the minimum and maximum values for the horizontal (x) axis and vertical (y) axis, respectively. The default for `xlim` is `NULL` and only needs to be defined if different axis limits than `fullGradientRange` is desired. The default for `ylim` is `c(0,1)` which likely leaves considerable empty white space above the KDEs, which can be reduced by adjusting this argument.
`logY`	Should the vertical axis (the relative height of rescaled KDEs) be portrayed with logarithmic scaling?

Details

In many ways, this is an attempt to create empirical versions of the the hypothetical figures portraying abundance response curves relative to an environmental gradient in Patzkowsky & Holland (2012).

The ecological gradient variable is often an environmental characteristic, such as depth, oxygenation, altitude, precipitation, but this is not necessarily so.

Value

Nothing is returned, just a plot is made.

References

Patzkowsky, M.E. and Holland, S.M., 2012. *Stratigraphic Paleobiology: Understanding the Distribution of Fossil Taxa in Time and Space.* University of Chicago Press. 259 pages.

See Also

This function mainly exists to look at the output from `getSpeciesSpecificRescaledKDE` for a fossil assemblage.

`plotHeatmapComparison` *Plots a Heatmap Comparison*

Description

This function is a complex wrapper of the functions `contour` and `filled.contour` which allows a user to combine colored contours for the surface of a third variables plotted across a two-dimensional space, with contour lines of the third variable's surface also plotted in that same two-dimensional space. This is often a common plotting need for this package, and thus is included here.

Usage

```
plotHeatmapComparison(
  x,
  y,
  z,
  xlim = range(x, finite = TRUE),
  ylim = range(y, finite = TRUE),
  zlim = range(z, finite = TRUE),
  xlog = FALSE,
  ylog = FALSE,
  xtick = pretty(x),
  ytick = pretty(y),
  contourLevels = NULL,
  nlevels = 10,
  contourLineLevels = NULL,
  contour.lwd = 2,
  additionalFeatures = NULL,
  palette = "plasma",
  xlab = "x",
  ylab = "y",
  main = "main title",
  gradientKeyLabel = "color gradient key",
  mtext_line = 3,
  margins = c(5, 6, 5, 5)
)
```

Arguments

`x, y`	The horizontal (`x`) and vertical (`y`) variables that determine the two-dimensional space within which the third variable (`z`) is plotted as a surface.
`z`	The values of the third variable (`z`) that will be used to define the plotted surface for contours, given as a matrix with the same number of rows as the length of `x`, and the same number of columns as the length of `y`.

`xlim`, `ylim`, `zlim` These are two-element vectors giving the minimum and maximum limits for the horizontal variable (`x`), vertical variable (`y`), and the variable defining the surface plotted (`z`) within the two dimensional space defined by `x` and `y`.

`xlog`, `ylog` Should the `x` or `y` axes be displayed with log-scaling?

`xtick`, `ytick` Vectors that give the positions of the tick-marks for `x` and `y` axes.

`contourLevels` A vector of values at which to put the breaks between the color-filled contours for `z`, Also determines the different levels show on the color-gradient key shown to the side of the contour plot.

`nlevels` The number of different color levels to use, if `contourLevels` is not defined.

`contourLineLevels`
A vector of values at which to put the distinct contour lines for `z`. This must be defined for contour lines to be plotted.

`contour.lwd` The thickness of plotted contour lines.

`additionalFeatures`
Additional features to add to the contour space, such as

`palette` The color palette to use for `filled.contour`. By default, the palette `"plasma"` is used.

`xlab`, `ylab` The labels for the `x` and `y` axes.

`main` The plot's main title.

`gradientKeyLabel`
The optional label text for the color gradient key for `z`, shown to the right of the main contour plot. This label will be shown to the right of the key.

`mtext_line` The distant in the margin away from the key at which the `gradientKeyLabel` is displayed. The default value is 3.

`margins` The size of the margins for the result plot. The default configuration gives some extra room on the left-hand size.

Details

The function `filled.contour` doesn't easy allow for sequential modifications, like adding additional contour lines to an existing contour plot, and so this function simplifies having to write the second `contour` plot as an argument for `plot.axes` in `filled.contour`.

Value

This function returns nothing at all as output. It just makes a plot.

`sampleFossilSeries` *Sample Fossil Assemblage Series*

Description

Given a time-series of 'true' fossil assemblages simulated in precise time, this function then chunks that 'true' ecological signal into sedimentary packages, which contain specimens from assemblages spanning the time interval during which that sediment accumulated. Further more, the inclusion of specimens from even more distant assemblages is used to model bioturbation.

Usage

```
sampleFossilSeries(
  bioturbIntensity,
  bioturbZoneDepth,
  distBetweenSamples,
  sampleWidth,
  simTimeVar,
  timestepAbundances,
  nSpecimens
)
```

Arguments

`bioturbIntensity`
: The degree of mixing within the bioturbation zone, as a value between 0 and 1. When intensity is 1, a given sample will consist only

`bioturbZoneDepth`
: The sediment depth to which bioturbation occurs. For example, Bioturbation depth varies considerably in the modern ocean, but is often around 10 centimeters – with the top ten centimeters of sediment (and the organic remains in those ten centimeters of sediment) being regularly moved up and down by organism activity. For the purposes of this model, a bioturbation zone depth of 10 centimeters means that sampling a centimeter of sediment at location X, the apparent fossil assemblage that would be recovered is just as likely to include specimens that were deposited five centimeters away as those deposited at location X.

`distBetweenSamples`
: The sedimentary thickness between successive samples, in the same units as `sampleWidth`.

`sampleWidth`
: The 'width' of a sample relative to core depth or outcrop height, usually given in linear units (usually centimeters). For taking sediment samples from a core, this is straightforward (how thick is each sediment sample taken?) but for outcrops this may be more difficult to determine (what is the thickness of a horizon in a shale unit?).

`simTimeVar`
: A data-frame specifying time-steps, sedimentary depth and environmental gradient values for simulating a time-series of sampled fossil assemblages.

timestepAbundances
: A matrix containing abundances for species as a series of simulated assemblages, output by getTimestepAbundances.

nSpecimens
: The number of specimens selected in each individual sample.

Details

This function is where bioturbation processes are handled, as well as time-averaging from samples capturing several sedimentary horizons reflecting multiple original fossil assemblages.

This function is generally run after running getTimestepAbundances. Most users will likely never run either function, instead running simulateFossilAssemblageSeries.

Value

A list composed of four components: simTimeVar, the input data-frame specifying time-steps, sedimentary depth and environmental gradient values; abundanceTable, a table of the abundances of species in each sample; sampleIntervals, a table specifying when in time each sample 'begins' and 'ends' in time (based on the sedimentation rate), and bioturbIntervals, a table specifying which intervals are 'included' in a sample

See Also

This function is generally run after running getTimestepAbundances. Most users will likely never run either function, instead running simulateFossilAssemblageSeries.

---

setupSimulatedGradientChange

Create a Stochastic Time-Series of Gradient Change For Use in Simulating Assemblage Change

---

Description

Given a series of inputs, simulates a sequence of gradient change against time for use in testing how environmental change alters the recovered sequence of fossil assemblages.

Usage

```
setupSimulatedGradientChange(
  nEvents,
  peakGradientValue,
  bgGradientValue,
  bgDurationRange,
  transitionDuration,
  eventDuration,
  halfGradientOnly = FALSE,
  includeInitialBackgroundPhase = TRUE,
  plot = FALSE
)
```

Arguments

nEvents	Number of events to occur in a simulated sequence of gradient change.
peakGradientValue	The gradient value at the 'peak' for an event that represents an excursion on that environmental gradient.
bgGradientValue	The gradient value expected during background intervals during which no notable excursion is occurring on that environmental gradient.
bgDurationRange	A vector of two values, representing the minimum and maximum duration (in time units) of a background interval between successive events.
transitionDuration	How long the transition between peak and background intervals should be. The longer this interval, the more chances of an assemblage being sampled that represents transitional gradient values.
eventDuration	The duration (in time-units) of a simulated event during which the environmental gradient is at an excursion 'peak' level.
halfGradientOnly	Whether to simulate only half of a background-event sequence, either beginning or terminating the simulation at the peak value. Only a single event can be simulated, so nEvents must be 1. The default is FALSE which signals to not simulated a half-gradient.
includeInitialBackgroundPhase	A logical indicating whether to include a lengthy background phase, for use in calibrating a simulation. This function is mainly for diagnostic purposes and may be removed in future updates.
plot	Should the simulated gradient be shown as a plot?

Details

This function is rather complicated and was written at a time when it was envisioned that simulations would involve time series of many repeated events with varying background intervals between them, rather than simulated sequences having only one event. In practice, use of paleoAM has tended to find the latter to be more useful.

Value

A list with five components: simGradient, a data frame giving the change in gradient values over time; approxGradientSeriesFunction, the simulated gradient given as an interpolated function; eventStartEndTimes, a vector of when each event and its preceding transition begin in time-units; eventPhaseStartTimes, a vector of when each new event phase (at the peak gradient value) begin in time-units; and backgroundStartEnd, a value indicating the time-step when the beginning background interval ends.

`simMixedAssemblageSample` *Simulate a Mixed Fossil Assemblage Composed of Communities at Different Gradient Values, and Sample the Lumped Assemblage*

Description

This function simulate a mixed fossil assemblage by simulating a series of communities across a defined range of gradient values, lumps them into a single mixed assemblage, and then samples that assemblage as defined by the user.

Usage

```
simMixedAssemblageSample(
  kdeRescaled,
  probSpeciesOccur,
  gradientValues,
  specimensPerTimestep,
  nSpecimens
)
```

Arguments

`kdeRescaled`	The list of modeled KDEs for species abundance, output from `getSpeciesSpecificRescaledKDE`.
`probSpeciesOccur`	The output from `getProbOccViaPresAbs`
`gradientValues`	A vector of gradient values to simulate over. A separate 'true' assemblage / community will be simulated for each value in the respective vector.
`specimensPerTimestep`	The number of specimens returned in a given time-step by `getTimestepAbundances`, usually set to an unrealistically high number to represent the true 'unsampled' fossil assemblage.
`nSpecimens`	The number of specimens selected in each individual sample.

Details

This function is mainly written for simulating what artificial mixtures of assemblages at different gradient values would look like if sampled and assumed to be a single cohesive assemblage.

Value

A matrix containing the species abundances in the resulting mixed assemblage.

simulateFossilAssemblageSeries *Simulate Time-Series of Successive Fossil Assemblages*

Description

Given a set of parameters and models describing species abundance, stochastically models changes in an underlying biotic gradient and simulates ecological change and a sequence of samples representing change in recovered fossil assemblages over that interval, including estimating the recovered gradient.

Usage

```
simulateFossilAssemblageSeries(
  kdeRescaled,
  probSpeciesOccur,
  origAbundData,
  eventChangeScale,
  bgGradientValue,
  fullGradientRange,
  eventSampleWidthRatio = NULL,
  sampleWidth = NULL,
  eventDuration = NULL,
  sedRatePerTimestep = NULL,
  samplingCompleteness,
  transitionDurationRatio,
  bioturbDepthRatio,
  bioturbIntensity,
  nEvents,
  nSpecimens,
  specimensPerTimestep = 10000,
  halfGradientOnly = FALSE,
  useTransformedRelAbundance = TRUE,
  projectIntoOrigDCA = TRUE,
  powerRootTransform = 1,
  maxSampleTimeStep = 500,
  minSampleTimeStep = 3,
  includeInitialBackgroundPhase = FALSE,
  plot = FALSE,
  thinOutput = FALSE
)
```

Arguments

kdeRescaled	The list of modeled KDEs for species abundance, output from getSpeciesSpecificRescaledKDE.
probSpeciesOccur	The output from getProbOccViaPresAbs

`origAbundData` The original matrix of abundance data, to be used to project the simulated data into the same detrended correspondence analysis (DCA) space.

`eventChangeScale`
A value indicating the amount relative to the background value (`bgGradientValue`) and the maximum possible change as indicated by `fullGradientRange` (in other words, simulated change must be within observed gradient, so `eventChangeScale` is a proportional multiplier of the total possible change).

`bgGradientValue`
The gradient value expected during background intervals during which no notable excursion is occurring on that environmental gradient.

`fullGradientRange`
A vector of two values giving the minimum and maximum gradient values observed in the empirical data.

`eventSampleWidthRatio`
How long should an event be relative to the amount of time (or sediment) captured within a sedimentary sample? This parameter is used for simulating event duration, sample width and sedimentation rate where any two of these three are defined and the third is not defined. This value is referred to as *Resolution Potential* in Belanger & Bapst (2023).

`sampleWidth` The 'width' of a sample relative to core depth or outcrop height, usually given in linear units (usually centimeters). For taking sediment samples from a core, this is straightforward (how thick is each sediment sample taken?) but for outcrops this may be more difficult to determine (what is the thickness of a horizon in a shale unit?).

`eventDuration` The duration (in time-units) of a simulated event during which the environmental gradient is at an excursion 'peak' level.

`sedRatePerTimestep`
The rate of sedimentation, given as a ratio of sediment thickness (given in linear dimensions, in the same units as `sampleWidth`), over time (given in the same time units as `eventDuration`.

`samplingCompleteness`
The relative completeness of stratigraphic sampling. For example, if two-centimeter wide samples of sediment are taken from a sediment core, every ten centimeters, then the `samplingCompleteness` is two over 10, or that `samplingCompleteness = 1/5`. A simulation with a sampling completeness of 1 would be comparable to exhaustively sampling a core that recorded no gaps in sedimentation over its history. Rocky outcrops are more complicated, as fossil-bearing horizons may be relatively thin compared to the thickness of the section, such that outcrop-based fossil records should be simulated as having *very low* `samplingCompleteness`.

`transitionDurationRatio`
The ratio of how long the transition between peak and background intervals should be, relative to the length of the peak 'event' duration (`eventDuration`). The longer this transition interval, the more chances of an assemblage being sampled that represents transitional gradient values.

`bioturbDepthRatio`
The ratio of the sediment depth to which bioturbation occurs, made relative to the width of a sediment sample (`sampleWidth`). A `sampleWidth` of 3 cm and a

biotubDepthRatio of 5 implies a bioturbation depth of 15 cm (3 * 5). Bioturbation depth varies considerably in the modern ocean, but is often the depth of active bioturbation is about 10 cm, such that the top ten centimeters of sediment (and the organic remains in those ten centimeters of sediment) are being regularly moved up and down by organism activity. For the purposes of this model, a bioturbation zone depth of 10 centimeters means that sampling a centimeter of sediment at location X, the apparent fossil assemblage that would be recovered is just as likely to include specimens that were deposited five centimeters away as those deposited at location X.

`bioturbIntensity`
: The degree of mixing within the bioturbation zone, as a value between 0 and 1. When intensity is 1, a given sample will consist only

`nEvents`
: Number of events to occur in a simulated sequence of gradient change.

`nSpecimens`
: The number of specimens selected in each individual sample.

`specimensPerTimestep`
: The number of specimens returned in a given time-step by `getTimestepAbundances`, usually set to an unrealistically high number to represent the true 'unsampled' fossil assemblage. Default is 10000.

`halfGradientOnly`
: Whether to simulate only half of a background-event sequence, either beginning or terminating the simulation at the peak value. Only a single event can be simulated, so `nEvents` must be 1. The default is `FALSE` which signals to not simulated a half-gradient.

`useTransformedRelAbundance`
: Should the DCA be analyzed

`projectIntoOrigDCA`
: Should the new simulated data be projected in the DCA generated by analyzing the original data? This is `TRUE` by default, which is what most users will likely use, as it is the preferable way to consider how the new simulated data relates to the original data.

`powerRootTransform`
: The power-root transform to be used on the abundance data before applying the DCA. By default this is 1, which means the data is not transformed at all. Note that the power-root transform is only performed if `useTransformedRelAbundance = TRUE`.

`maxSampleTimeStep`
: The maximum number of individual time-steps used for simulating a sample.

`minSampleTimeStep`
: The minimum number of individual time-steps used for simulating a sample.

`includeInitialBackgroundPhase`
: A logical indicating whether to include a lengthy background phase, for use in calibrating a simulation. This function is mainly for diagnostic purposes and may be removed in future updates.

`plot`
: Should the simulated time-series of fossil assemblages be shown as a sequence of generating and recovered gradient values against time? Default is `FALSE`.

`thinOutput`
: Should the output be thinned to just the sample properties and intrinsic variables? Default is FALSE.

Details

Different parameterizations may be given as input, allowing different parameters to be unspecified. Missing parameters are then calculated from the specified ones using `calculateImplicitParameters`.

Value

Returns a list, which by default has seven components: `implicitParameters`, the full list of parameters used for generating the simulated data; `simGradientChangeOut`, the simulated time-series of gradient change output by `setupSimulatedGradientChange`; `maxTime`, the total duration of the entire simulated time-series from start to end; `simTimeVar`, a data frame specifying time-steps, sedimentary depth and environmental gradient values for simulating a time-series of sampled fossil assemblages, used as input in `sampleFossilSeries`; `fossilSeries`, a list containing the simulated time-series of sampled fossil assemblages from `sampleFossilSeries`, `ecology`, the recovered ecological variables for each simulated sample, as returned by internal function `quantifyCommunityEcology`, and `sampleProperties`, a list containing a number of variables specific to individual .

If `thinList = TRUE` is used, then the output list contains only two components: `sampleProperties` and `implicitParameters`. The `implicitParameters` component is the same as in the full output, but the `sampleProperties` component only contains information on when (in both time and sedimentary depth) a given sample is located in the simulated time-series, and the variable `scoreDCA1_recovered`.

References

Belanger, Christina L., and David W. Bapst. 2023. "Simulating our ability to accurately detect abrupt changes in assemblage-based paleoenvironmental proxies." Palaeontologia Electronica 26 (2), 1-32

See Also

`calculateImplicitParameters`

Examples

```
# an example with Gulf of Alaska data

# load data
data(gulfOfAlaska)

alaskaKDEs <- getSpeciesSpecificRescaledKDE(
    gradientOrigDCA = DCA1_GOA,
    origAbundData = abundData_GOA,
    abundanceFloorRatio = 0.5,
    nBreaksGradientHist = 20,
    modeledSiteAbundance = 10000
    )

alaskaProbOccur <- getProbOccViaPresAbs(
   gradientOrigDCA = DCA1_GOA,
   origAbundData = abundData_GOA
   )
```

```
# Run the simulation of fossil assemblages
    # simulateFossilAssemblageSeries has lots of arguments...
    # below they are broken up into groups, seperate by #
    # matches scenarios from fig 13 of Belanger & Bapst

fossilSeriesOut <- simulateFossilAssemblageSeries(
      # inputs
      kdeRescaled = alaskaKDEs,
      probSpeciesOccur = alaskaProbOccur,
      origAbundData = abundData_GOA,
      fullGradientRange = c(min(DCA1_GOA), max(DCA1_GOA)),

      # let's make it relatively mild event
        # with a long transition
      eventChangeScale = 0.5,
      bgGradientValue = -1,
      transitionDurationRatio = 1,

      # don't need to define eventSampleWidthRatio
        # - only need to define three of eventSampleWidthRatio,
        # sampleWidth, eventDuration, sedRatePerTimestep
      sampleWidth = 3,
      eventDuration = 100,
      sedRatePerTimestep = 0.1,

      # sample every third sample-width worth of core
      samplingCompleteness = 1/3,
      # no bioturbation
      bioturbDepthRatio = 0,
      bioturbIntensity = 0,

      nEvents = 1,
      nSpecimens = 100,
      # let's plot it
      plot = TRUE
      )
```

`simulateGradientQuantile` *Repeatedly Simulate Sampled Assemblages at some Gradient Value, and Return a Quantile Based on Recovered Gradient Values*

Description

This function simulates assemblages at a single given gradient value, and returns a specified quantile on the recovered gradient values for the sake of defining an envelope around on recovered gradient values.

Usage

```
simulateGradientQuantile(
  quantileProbs = c(0.95),
  nSamplesSim,
  gradientValue,
  origAbundData,
  kdeRescaled,
  probSpeciesOccur,
  powerRootTransform = 1,
  specimensPerTimestep = 10000,
  nSpecimens
)
```

Arguments

quantileProbs	The quantile for which to return on the recovered gradient values from the simulated assemblages. (Technically multiple quantiles can be given, for which a value will be returned for each.
nSamplesSim	The number of samples to simulate.
gradientValue	The gradient value to simulate assemblages at.
origAbundData	The original matrix of abundance data, to be used to project the simulated data into the same detrended correspondence analysis (DCA) space.
kdeRescaled	The list of modeled KDEs for species abundance, output from getSpeciesSpecificRescaledKDE.
probSpeciesOccur	The output from getProbOccViaPresAbs
powerRootTransform	The power-root transform to be used on the abundance data before applying the DCA. By default this is 1, which means the data is not transformed at all. Note that the power-root transform is only performed if useTransformedRelAbundance = TRUE.
specimensPerTimestep	The number of specimens returned in a given time-step by getTimestepAbundances, usually set to an unrealistically high number to represent the true 'unsampled' fossil assemblage.
nSpecimens	The number of specimens selected in each individual sample.

Details

This function is most useful with applications like getRecoveredTransitionDuration which use envelope values to define features of a recovered sequence of gradient values for comparing simulated and empirical data.

Value

A value for each quantile specified in quantileProbs. May be multiple values if quantileProbs is a vector with more than one value.

Index

∗ **datasets**
gulfOfAlaska, 11
hirnantian, 12

abundData_GOA *(*gulfOfAlaska*)*, 11

calculateImplicitParameters, 2, *25*

DCA1_GOA *(*gulfOfAlaska*)*, 11

fullDataTable_GOA *(*gulfOfAlaska*)*, 11

getProbOccViaPresAbs, 4, *9, 10*, *21, 22*, *27*
getRecoveredTransitionDuration, 6
getSampleDCA, 7
getSpeciesSpecificRescaledKDE, *5*, 8, *10*, *15*, *21, 22*, *27*
getTimestepAbundances, 10, *19*
graptCommData *(*hirnantian*)*, 12
graptSampleInfo *(*hirnantian*)*, 12
gulfOfAlaska, 11, *13*

hirnantian, *12*, 12

plotFossilAssemblageSeriesDCA, 13
plotGradientKDE, *9*, 14
plotHeatmapComparison, 16

sampleFossilSeries, *10, 11*, 18, *25*
setupSimulatedGradientChange, 19
simMixedAssemblageSample, 21
simulateFossilAssemblageSeries, *4*, *10, 11*, *14*, *19*, 22
simulateGradientQuantile, *7*, 26