Package ‘nicheROVER’

October 13, 2022

Title Niche Region and Niche Overlap Metrics for Multidimensional Ecological Niches

Version 1.1.0

Date 2021-06-03

Description Implementation of a probabilistic method to calculate 'nicheROVER' (_niche_ _r_egion and niche _over_lap) metrics using multidimensional niche indicator data (e.g., stable isotopes, environmental variables, etc.). The niche region is defined as the joint probability density function of the multidimensional niche indicators at a user-defined probability alpha (e.g., 95%). Uncertainty is accounted for in a Bayesian framework, and the method can be extended to three or more indicator dimensions. It provides directional estimates of niche overlap, accounts for species-specific distributions in multivariate niche space, and produces unique and consistent bivariate projections of the multivariate niche region. The article by Swanson et al. (Ecology, 2015) provides a detailed description of the methodology. See the package vignette for a worked example using fish stable isotope data.

Depends R (>= 1.9.0)

Imports graphics, stats, utils

Suggests knitr, rmarkdown

VignetteBuilder knitr

License GPL-3

LazyData true

Encoding UTF-8

RoxygenNote 7.1.1

NeedsCompilation no

Author Martin Lysy [aut, cre],
Ashley D. Stasko [aut],
Heidi K. Swanson [aut]

Maintainer Martin Lysy <`mlysy@uwaterloo.ca`>

Repository CRAN

Date/Publication 2021-06-03 18:10:02 UTC

R topics documented:

`ellipse` *Point coordinates for a 2-D ellipse.*

Description

Calculates coordinates of points for plotting a 2-dimensional ellipse based on user-defined parameters. Can be used for exploratory data analysis to produce ellipses at a given niche region size (e.g., $\alpha = 95\%$).

Usage

```
ellipse(mu, V, alpha = 0.95, n = 100)
```

Arguments

`mu`	Centre of ellipse. A vector of length 2.
`V`	Scale of ellipse. A 2x2 matrix. See 'Details'.
`alpha`	Niche region size. See 'Details'.
`n`	Number of points to return for plotting.

Details

This function provides the coordinates needed to plot a 2-dimensional ellipse based on user-defined parameters, such that `X = c(x,y)` satisfies the equation

$$(X - \mu)' V^{-1} (X - \mu) = C,$$

where $C =$ `qchisq(alpha, df = 2)`.

Value

Returns a matrix of coordinates `cbind(x,y)` to plot a 2-dimensional ellipse.

See Also

`niche.plot()` for plotting.

Examples

```
mu <- rnorm(2)
V <- crossprod(matrix(rnorm(4), 2, 2))
ell.pts <- ellipse(mu = mu, V = V, alpha = .9, n = 100)
plot(ell.pts, col = rainbow(110)[1:100], type = "o")
points(mu[1], mu[2], pch = "+")
```

`fish` *Fish stable isotope dataset.*

Description

A dataset containing values for three stable isotopes measured in the muscle tissue of four species of arctic fish. For use in examples.

Usage

```
fish
```

Format

A data frame with 278 rows (observations) and 4 columns (species, delta^15N, delta^13C, and delta^34S).

Details

This dataset contains delta^15N, delta^13C, and delta^34S values for the following fish species:

- ARCS - Arctic Cisco (*Coregonus autumnalis*), $n = 69$.
- BDWF - Broad Whitefish (*Coregonus nasus*), $n = 71$.
- LKWF - Lake Whitefish (*Coregonus clupeaformis*), $n = 67$.
- LSCS - Least Cisco (*Coregonus sardinella*), $n = 70$

Source

Fish were collected between 2007 and 2008 from an estuarine area of the Beaufort Sea, North and West of the Mackenzie Delta at Phillips Bay, Yukon Territory, Canada (69.28 N, 138.49 W).

Examples

```
head(fish)
aggregate(fish[2:4], fish[1], mean)
```

`niche.par.plot` *Plot for niche parameters.*

Description

For one or more species, plots some or all of the niche parameters μ and Σ.

Usage

```
niche.par.plot(
  niche.par,
  plot.mu = TRUE,
  plot.Sigma = TRUE,
  plot.index,
  col,
  ndens = 512,
  ylab
)
```

Arguments

`niche.par`	List with `nspecies = length(niche.par)`, each element of which is a list with parameters `mu` and `Sigma`. See 'Details'.
`plot.mu`	Logical. If `TRUE`, plot the distribution of μ for each niche indicator (e.g., stable isotope). See 'Details'.
`plot.Sigma`	Logical. If `TRUE`, plot the distribution of Σ for each niche indicator. See 'Details'.
`plot.index`	Either a scalar of a numeric vector of length 2. If `plot.index = i` then plot the distribution of μ_i. If `plot.index = c(i,j)` then plot the distribution of Σ_{ij}.
`col`	Vector of colors in which to plot each species.
`ndens`	Number of points at which to evaluate density estimates.
`ylab`	Optional label for y-axis. If missing, defaults to $p(\mu_i\|X)$ and $p(\Sigma_{ij}\|X)$.

Details

Each element of the list `niche.par` is a distribution of niche parameters. That is, `names(niche.par[[1]]) = c("mu", "Sigma")`, and if `niso` is the number of niche indicators (e.g., stable isotopes), then `dim(niche.par[[1]]$mu) = c(nsamples, niso)` and `dim(niche.par[[1]]$Sigma) = c(niso, niso, nsamples)`.

Value

Returns a plot of the distribution of some or all niche parameters.

See Also

`niw.post()`, `niiw.post()` for niche parameter output, `stats::density()` for density estimation from sample data.

Examples

```
# fish data
data(fish)

# generate parameter draws from the "default" posteriors of each fish
nsamples <- 1e3
system.time({
  fish.par <- tapply(1:nrow(fish), fish$species,
                     function(ii) niw.post(nsamples = nsamples, X = fish[ii,2:4]))
})

# various parameter plots
clrs <- c("black", "red", "blue", "orange") # colors for each species

# mu1, mu2, and Sigma12
par(mar = c(4, 4, .5, .1)+.1, mfrow = c(1,3))
niche.par.plot(fish.par, col = clrs, plot.index = 1)
niche.par.plot(fish.par, col = clrs, plot.index = 2)
niche.par.plot(fish.par, col = clrs, plot.index = 1:2)
legend("topright", legend = names(fish.par), fill = clrs)

# all mu
niche.par.plot(fish.par, col = clrs, plot.mu = TRUE, plot.Sigma = FALSE)
legend("topright", legend = names(fish.par), fill = clrs)

# all mu and Sigma
par(mar = c(4.2, 4.2, 2, 1)+.1)
niche.par.plot(fish.par, col = clrs, plot.mu = TRUE, plot.Sigma = TRUE)
legend("topright", legend = names(fish.par), fill = clrs)
```

`niche.plot` *Plot for 2-d projection of niche regions.*

Description

For one or more species, creates a series of plots: (i) the raw niche indicators (e.g., stable isotope) data, (ii) their density estimates, and (iii) 2-dimensional projections of probabilistic niche regions based on n-dimensionsional data.

Usage

```
niche.plot(
  niche.par,
  niche.data,
  alpha = 0.95,
  species.names,
  iso.names,
  lims,
  col,
  ndens = 512,
  pfrac = 0,
  xlab
)
```

Arguments

`niche.par`	A list of length `nspecies`, each element of which in turn is a list with elements `mu` and `Sigma`. Each of these will correspond to an ellipse being drawn for that species in the corresponding 2-d plane. See 'Example'.
`niche.data`	A list of length `nspecies`, each element of which is a matrix with observations along the rows and niche indicators (e.g., stable isotopes) along the columns.
`alpha`	Size of the niche region to plot. Defaults to 0.95.
`species.names`	Names of the species. Defaults to `names(niche.par)`.
`iso.names`	Names of the niche indicators (or isotopes) to plot. Defaults to `colnames(niche.par)`.
`lims`	Two-row matrix of range limits for each niche indicator. Defaults to include all ellipses.
`col`	Vector of colours in which each species will be drawn.
`ndens`	Number of points at which to evaluate kernel density estimates.
`pfrac`	Fraction of the plot on which to display 1-dimensional raw niche indicator data. `pfrac = 0` means don't display the raw data in 1-d.
`xlab`	Title of plot, located on the bottom. Defaults to no title.

Details

A set of plots is created for each pairwise combination of niche indicators. Below the diagonal are scatterplots for each species, above the diagonal are ellipses corresponding to 2-d projections of the probabilistic niche regions. The diagonal displays density estimates for each indicator, and optionally the raw 1-d data. See Swanson et al. (2015) for detailed description of the probabilistic niche region.

Value

Returns a series of plots displaying niche indicator data and their probabilistic niche projections.

References

Swanson, H.K., Lysy, M., Stasko, A.D., Power, M., Johnson, J.D., and Reist, J.D. "A new probabilistic method for quantifying n-dimensional ecological niches and niche overlap." *Ecology: Statistical Reports* 96:2 (2015): 318-324. doi: 10.1890/140235.1.

See Also

`overlap.plot()`, `niw.post()`, `niiw.post()`.

Examples

```
data(fish) # 4 fish, 3 isotopes

# generate 10 parameter draws from the posteriors
# of each fish with default prior
nsamples <- 10
fish.par <- tapply(1:nrow(fish), fish$species,
                   function(ii) niw.post(nsamples = nsamples, X = fish[ii,2:4]))

# format data for plotting function
fish.data <- tapply(1:nrow(fish), fish$species, function(ii) X = fish[ii,2:4])

clrs <- c("black", "red", "blue", "orange") # colors for each species
niche.plot(niche.par = fish.par, niche.data = fish.data, pfrac = .1,
           iso.names = expression(delta^{15}*N, delta^{13}*C, delta^{34}*S),
           col = clrs, xlab = expression("Isotope Ratio (\u2030)"))
```

`niche.runif` *Uniform sampling from an elliptical niche region.*

Description

Uniform sampling from an elliptical niche region.

Usage

```
niche.runif(n, mu, Sigma, alpha = 0.95)
```

Arguments

`n`	Number of random draws.
`mu`	Mean vector.
`Sigma`	Variance matrix.
`alpha`	Probabilistic niche size

Value

IID draws from a uniform distribution on the elliptical niche region.

See Also

`ellipse()` and `niche.size()` for the definition of the elliptical niche region.

Examples

```
# 2d example
d <- 2 # number of dimensions
V <- crossprod(matrix(rnorm(4),d,d))
mu <- rnorm(d)
plot(ellipse(mu, V), type = "l")
points(niche.runif(1e4, mu, V), col = "brown", pch = ".")
```

`niche.size` *Calculate the size of an elliptical niche region.*

Description

Calculate the size of an elliptical niche region.

Usage

```
niche.size(Sigma, alpha = 0.95)
```

Arguments

`Sigma`	Variance matrix for normally distributed niche axes.
`alpha`	Probabilistic niche size.

Details

For a given niche region N_R, the niche size is defined as the hypervolume of this region: $N_S = \int_{x \in N_R} dx$.

Value

Hypervolume niche size.

Examples

```
# for each species, size of 95% niche region using sample variance
tapply(1:nrow(fish), fish$species, function(ind) {
  X <- fish[ind,2:4] # all measurements for given species
  Sighat <- var(X) # sample variance
  niche.size(Sigma = Sighat, alpha = .95)
})
```

`nicheROVER` *(Niche) (R)egion and Niche (Over)lap Metrics for Multidimensional Ecological Niches.*

Description

This package uses a probabilistic method to calculate niche regions and pairwise niche overlap using multidimensional niche indicator data (e.g., stable isotopes, environmental variables, etc.). The niche region is defined as the joint probability density function of the multidimensional niche indicators at a user-defined probability alpha (e.g., 95%). Uncertainty is accounted for in a Bayesian framework, and the method can be extended to three or more indicator dimensions. It provides directional estimates of niche overlap, accounts for species-specific distributions in multivariate niche space, and produces unique and consistent bivariate projections of the multivariate niche region. See Swanson et al. (2014) for a detailed description and worked example below using fish stable isotope data.

References

Swanson, H.K., Lysy, M., Stasko, A.D., Power, M., Johnson, J.D., and Reist, J.D. "A new probabilistic method for quantifying n-dimensional ecological niches and niche overlap." *Ecology: Statistical Reports* 96:2 (2015): 318-324. doi: 10.1890/140235.1.

Examples

```
# analysis for fish data

data(fish) # 4 fish, 3 isotopes
aggregate(fish[2:4], fish[1], mean) # isotope means per fish

# random draws from posterior distribution with default prior
nsamples <- 500
fish.par <- tapply(1:nrow(fish), fish$species,
                   function(ii) niw.post(nsamples = nsamples, X = fish[ii,2:4]))

# display p(mu | X) and p(Sigma | X) for each fish
clrs <- c("black", "red", "blue", "orange") # colors for each species
par(mar = c(4.2, 4.2, 2, 1)+.1)
niche.par.plot(fish.par, col = clrs)
legend(x = "topright", legend = names(fish.par), fill = clrs)

# 2-d projections of 10 niche regions
nsamples <- 10
fish.par <- tapply(1:nrow(fish), fish$species,
                   function(ii) niw.post(nsamples = nsamples, X = fish[ii,2:4]))

# format data for plotting function
fish.data <- tapply(1:nrow(fish), fish$species, function(ii) X = fish[ii,2:4])

niche.plot(niche.par = fish.par, niche.data = fish.data, pfrac = .05,
```

```
           iso.names = expression(delta^{15}*N, delta^{13}*C, delta^{34}*S),
           col = clrs, xlab = expression("Isotope Ratio (\u2030)"))

# niche overlap plots for 95% niche region sizes

# overlap calculation.  use nsamples = nprob = 1e4 for higher accuracy.
nsamples <- 500
over.stat <- overlap(fish.par, nreps = nsamples, nprob = nsamples, alpha = .95)

# overlap plot
overlap.plot(over.stat, col = clrs, mean.cred.col = "turquoise",
             equal.axis = TRUE,
             xlab = "Overlap Probability (%) -- Niche Region Size: 95%")
```

`niiw.post` *Random draws from the posterior distribution with Normal-Independent-Inverse-Wishart (NIIW) prior.*

Description

Given iid d-dimensional niche indicators $X = (X_1, \dots, X_N)$ with $X_i \sim N(\mu, \Sigma)$, this function generates random draws from $p(\mu, \Sigma | X)$ for the Normal-Independent-Inverse-Wishart (NIIW) prior.

Usage

```
niiw.post(nsamples, X, lambda, Omega, Psi, nu, mu0 = lambda, burn)
```

Arguments

`nsamples`	The number of posterior draws.
`X`	A data matrix with observations along the rows.
`lambda`	Mean of μ. See 'Details'.
`Omega`	Variance of μ. Defaults to `Omega = 0`. See 'Details'.
`Psi`	Scale matrix of Σ. Defaults to `Psi = 0`. See 'Details'.
`nu`	Degrees of freedom of Σ. Defaults to `nu = ncol(X)+1`. See 'Details'.
`mu0`	Initial value of μ to start the Gibbs sampler. See 'Details'.
`burn`	Burn-in for the MCMC sampling algorithm. Either an integer giving the number of initial samples to discard, or a fraction with `0 < burn < 1`. Defaults to `burn = floor(nsamples/10)`.

Details

The NIIW distribution $p(\mu, \Sigma|\lambda, \kappa, \Psi, \nu)$ is defined as

$$\Sigma \sim W^{-1}(\Psi, \nu), \quad \mu|\Sigma \sim N(\lambda, \Omega).$$

The default value `Omega = 0` uses the Lebesque prior on μ: $p(\mu) \propto 1$. In this case the NIW and NIIW priors produce identical resuls, but `niw.post()` is faster.

The default value `Psi = 0` uses the scale-invariant prior on Σ: $p(\Sigma) \propto |\Sigma|^{-(\nu+d+1)/2}$.

The default value `nu = ncol(X)+1` for `Omega = 0` and `Psi = 0` makes $E[\mu|X] = `colMeans(X)`$ and $E[\Sigma|X] = `var(X)`$.

Random draws are obtained by a Markov chain Monte Carlo (MCMC) algorithm; specifically, a Gibbs sampler alternates between draws from $p(\mu|\Sigma, X)$ and $p(\Sigma|\mu, X)$, which are Normal and Inverse-Wishart distributions respectively.

Value

Returns a list with elements `mu` and `Sigma` of sizes `c(nsamples, length(lambda))` and `c(dim(Psi), nsamples)`.

See Also

`niw.post()`, `rwish()`.

Examples

```
# simulate normal data with mean and variance (mu0, Sigma0)
d <- 4
mu0 <- rnorm(d)
Sigma0 <- matrix(rnorm(d^2), d, d)
Sigma0 <- Sigma0 %*% t(Sigma0)
N <- 1e2
X <- matrix(rnorm(N*d), N, d) # iid N(0,1)
X <- t(t(X %*% chol(Sigma0)) + mu0) # each row is N(mu0, Sigma)

# prior parameters
# flat prior on mu
lambda <- 0
Omega <- 0
# informative prior on Sigma
Psi <- crossprod(matrix(rnorm(d^2), d, d))
nu <- 5

# sample from NIIW posterior
nsamples <- 2e3
system.time({
  siiw <- niiw.post(nsamples, X, lambda, Omega, Psi, nu, burn = 100)
})

# sample from NIW posterior
kappa <- 0
```

```
system.time({
  siw <- niw.post(nsamples, X, lambda, kappa, Psi, nu)
})

# check that posteriors are the same

# p(mu | X)
clrs <- c("black", "red")
par(mar = c(4.2, 4.2, 2, 1)+.1)
niche.par.plot(list(siiw, siw), col = clrs, plot.mu = TRUE, plot.Sigma = FALSE)
legend(x = "topright", legend = c("NIIW Prior", "NIW Prior"), fill = clrs)

# p(Sigma | X)
par(mar = c(4.2, 4.2, 2, 1)+.1)
niche.par.plot(list(siiw, siw), col = clrs, plot.mu = FALSE, plot.Sigma = TRUE)
legend(x = "topright", legend = c("NIIW Prior", "NIW Prior"), fill = clrs)
```

`niw.coeffs`

Posterior coefficients of the Normal-Inverse-Wishart distribution with its conjugate prior.

Description

Given iid d-dimensional niche indicators $X = (X_1, \ldots, X_N)$ with $X_i \sim N(\mu, \Sigma)$, this function calculates the coefficients of the Normal-Inverse-Wishart (NIW) posterior $p(\mu, \Sigma|X)$ for a conjugate NIW prior. Together with `niw.mom()`, this can be used to rapidly compute the point estimates $E[\mu|X]$ and $E[\Sigma|X]$.

Usage

```
niw.coeffs(X, lambda, kappa, Psi, nu)
```

Arguments

`X`	A data matrix with observations along the rows.
`lambda`	Location parameter. See 'Details'.
`kappa`	Scale parameter. Defaults to `kappa = 0`. See 'Details'.
`Psi`	Scale matrix. Defaults to `Psi = 0`. See 'Details'.
`nu`	Degrees of freedom. Defaults to `nu = ncol(X)+1`. See 'Details'.

Details

The NIW distribution $p(\mu, \Sigma|\lambda, \kappa, \Psi, \nu)$ is defined as

$$\Sigma \sim W^{-1}(\Psi, \nu), \quad \mu|\Sigma \sim N(\lambda, \Sigma/\kappa).$$

The default value `kappa = 0` uses the Lebesque prior on μ: $p(\mu) \propto 1$.

The default value `Psi = 0` uses the scale-invariant prior on Σ: $p(\Sigma) \propto |\Sigma|^{-(\nu+d+1)/2}$.

The default value `nu = ncol(X)+1` for `kappa = 0` and `Psi = 0` makes $E[\mu|X] =$ ‘$colMeans(X)$‘ and $E[\Sigma|X] =$ ‘$var(X)$‘.

Value

Returns a list with elements `lambda`, `kappa`, `Psi`, `nu` corresponding to the coefficients of the NIW posterior distribution $p(\mu, \Sigma|X)$.

See Also

`rniw()`, `niw.mom()`, `niw.post()`.

Examples

```
# NIW prior coefficients
d <- 3
lambda <- rnorm(d)
kappa <- 5
Psi <- crossprod(matrix(rnorm(d^2), d, d))
nu <- 10

# data
data(fish)
X <- fish[fish$species == "ARCS",2:4]

# NIW posterior coefficients
post.coef <- niw.coeffs(X, lambda, kappa, Psi, nu)

# compare
mu.mean <- niw.mom(post.coef$lambda, post.coef$kappa, post.coef$Psi, post.coef$nu)$mu$mean
mu.est <- rbind(prior = niw.mom(lambda, kappa, Psi, nu)$mu$mean,
                data = colMeans(X),
                post = mu.mean)
round(mu.est, 2)
```

`niw.mom` *Mean and variance of the Normal-Inverse-Wishart distribution.*

Description

This function computes the mean and variance of the Normal-Inverse-Wishart (NIW) distribution. Can be used to very quickly compute Bayesian point estimates for the conjugate NIW prior.

Usage

```
niw.mom(lambda, kappa, Psi, nu)
```

Arguments

`lambda`	Location parameter. See 'Details'.
`kappa`	Scale parameter. See 'Details'.
`Psi`	Scale matrix. See 'Details'.
`nu`	Degrees of freedom. See 'Details'.

Details

The NIW distribution $p(\mu, \Sigma|\lambda, \kappa, \Psi, \nu)$ is defined as

$$\Sigma \sim W^{-1}(\Psi, \nu), \quad \mu|\Sigma \sim N(\lambda, \Sigma/\kappa).$$

Note that $\mathrm{cov}(\mu, \Sigma) = 0$.

Value

Returns a list with elements `mu` and `Sigma`, each containing lists with elements `mean` and `var`. For `mu` these elements are of size `length(lambda)` and `c(length(lambda),length(lambda))`. For `Sigma` they are of size `dim(Psi)` and `c(dim(Psi), dim(Psi))`, such that $\mathrm{cov}(\Sigma_{ij}, \Sigma_{kl})$ =`Sigma$var[i,j,k,l]`.

See Also

`rniw()`, `niw.coeffs()`, `niw.post()`.

Examples

```
# NIW parameters
d <- 3 # number of dimensions
lambda <- rnorm(d)
kappa <- 2
Psi <- crossprod(matrix(rnorm(d^2), d, d))
nu <- 10

# simulate data
niw.sim <- rniw(n = 1e4, lambda, kappa, Psi, nu)

# check moments
niw.mV <- niw.mom(lambda, kappa, Psi, nu)

# mean of mu
ii <- 1
c(true = niw.mV$mu$mean[ii], sim = mean(niw.sim$mu[,ii]))

# variance of Sigma
II <- c(1,2)
JJ <- c(2,3)
c(true = niw.mV$var[II[1],II[2],JJ[1],JJ[2]],
  sim = cov(niw.sim$Sigma[II[1],II[2],], niw.sim$Sigma[JJ[1],JJ[2],]))
```

`niw.post` *Random draws from the posterior distribution with Normal-Inverse-Wishart (NIW) prior.*

Description

Given iid d-dimensional niche indicators $X = (X_1, \ldots, X_N)$ with $X_i \sim N(\mu, \Sigma)$, this function generates random draws from $p(\mu, \Sigma|X)$ for the Normal-Inverse-Wishart (NIW) prior.

Usage

```
niw.post(nsamples, X, lambda, kappa, Psi, nu)
```

Arguments

`nsamples`	The number of posterior draws.
`X`	A data matrix with observations along the rows.
`lambda`	Location parameter. See 'Details'.
`kappa`	Scale parameter. Defaults to `kappa = 0`. See 'Details'.
`Psi`	Scale matrix. Defaults to `Psi = 0`. See Details.
`nu`	Degrees of freedom. Defaults to `nu = ncol(X)+1`. See Details.

Details

The NIW distribution $p(\mu, \Sigma | \lambda, \kappa, \Psi, \nu)$ is defined as

$$\Sigma \sim W^{-1}(\Psi, \nu), \quad \mu|\Sigma \sim N(\lambda, \Sigma/\kappa).$$

The default value `kappa = 0` uses the Lebesque prior on μ: $p(\mu) \propto 1$.

The default value `Psi = 0` uses the scale-invariant prior on Σ: $p(\Sigma) \propto |\Sigma|^{-(\nu+d+1)/2}$.

The default value `nu = ncol(X)+1` for `kappa = 0` and `Psi = 0` makes $E[\mu|X] =$ `colMeans(X)` and $E[\Sigma|X] =$ `var(X)`.

Value

Returns a list with elements `mu` and `Sigma` of sizes `c(nsamples, length(lambda))` and `c(dim(Psi), nsamples)`.

See Also

`rniw()`, `niiw.post()`.

Examples

```
# compare the default non-informative prior to an arbitrary informative prior
# for simulated data

# simulate normal data with mean and variance (mu0, Sigma0)
d <- 4
mu0 <- rnorm(d)
Sigma0 <- matrix(rnorm(d^2), d, d)
Sigma0 <- Sigma0 %*% t(Sigma0)
N <- 1e2
X <- matrix(rnorm(N*d), N, d) # iid N(0,1)
X <- t(t(X %*% chol(Sigma0)) + mu0) # each row is N(mu0, Sigma)

# informative prior parameters
lambda <- rnorm(d)
kappa <- 20
```

```
Psi <- crossprod(matrix(rnorm(d^2), d, d))
nu <- 5

# iid draws from informative prior pi(mu, Sigma)
nsamples <- 2e3
siw0 <- rniw(nsamples, lambda, kappa, Psi, nu)

# iid draws from posterior p(mu, Sigma | X) with informative prior
siw1 <- niw.post(nsamples, X, lambda, kappa, Psi, nu)

# iid draws from posterior p(mu, Sigma | X) with default noninformative prior
siw2 <- niw.post(nsamples, X)

# compare

# prior and posterior densities of mu
clrs <- c("orange", "red", "blue", "black")
ii <- 1
par(mar = c(4.2, 4.2, 2, 1)+.1)
niche.par.plot(list(siw0, siw1, siw2), col = clrs[1:3],
               plot.index = ii, ylab = "Density")
abline(v = mu0[ii], col = clrs[4]) # true value of mu
legend(x = "topright",
       legend = c(parse(text = paste0("pi(mu[", ii, "])")),
                  parse(text = paste0("p(mu[", ii, "]*\" | \"*X)*\", Informative Prior\"")),
                parse(text = paste0("p(mu[", ii, "]*\" | \"*X)*\", Noninformative Prior\"")),
                     parse(text = paste0("\"True value of \"*mu[", ii, "]"))),
       fill = clrs)

# prior and posterior densities of Sigma
ii <- 1
jj <- 2
par(mar = c(4.2, 4.2, 2, 1)+.1)
niche.par.plot(list(siw0, siw1, siw2), col = clrs[1:3],
               plot.index = c(ii,jj), ylab = "Density")
abline(v = Sigma0[ii,jj], col = clrs[4])
legend(x = "topright",
       legend = c(parse(text = paste0("pi(Sigma[", ii, "*", jj, "])")),
                  parse(text = paste0("p(Sigma[", ii, "*", jj,
                                      "]*\" | \"*X)*\", Informative Prior\"")),
                  parse(text = paste0("p(Sigma[", ii, "*", jj,
                                      "]*\" | \"*X)*\", Noninformative Prior\"")),
                  parse(text = paste0("\"True value of \"*Sigma[", ii, "*", jj, "]"))),
       fill = clrs)
```

`overlap` *Monte Carlo calculation of niche region overlap metrics.*

Description

Calculates the distribution of a niche region overlap metric for each pairwise species combination and user-specified niche region sizes.

Usage

```
overlap(
  niche.par,
  nreps,
  nprob,
  alpha = 0.95,
  species.names,
  norm.redraw = TRUE
)
```

Arguments

`niche.par`	A list with `nspecies = length(niche.par)`, each element of which in turn is a list with elements `mu` and `Sigma`. See Details.
`nreps`	The number of overlap metric calculations for each species. Defaults to the smallest number of parameter samples supplied by `niche.par`. See 'Details'.
`nprob`	The number of normal draws for each Monte Carlo overlap metric calculation. See 'Details'.
`alpha`	Scalar or vector of niche region sizes for calculating the niche overlap metric. Defaults to 0.95.
`species.names`	Names of the species. Defaults to `names(niche.par)`.
`norm.redraw`	Logical. If `FALSE`, the same `nprob*nspecies` iid $N(0,1)$ draws are used for each calculation of the overlap metric. This increases the Monte Carlo error, but the procedure is about 1.5x faster. Defaults to `TRUE`.

Details

The overlap metric is the probability that a randomly drawn individual from species A will be found within the niche region of species B (for a given niche region size, e.g., `alpha = .95`). It is a single number which is a function of the parameters for each species, $\Theta_A = (\mu_A, \Sigma_A)$ and $\Theta_B = (\mu_B, \Sigma_B)$. This number is difficult to calculate directly, but easy to approximate stochastically by generating `nprob` draws from the distribution of species A and counting the fraction of them which fall in the niche region of species B.

Typically the true values of Θ_A and Θ_B are unknown and must be estimated from the data. Thus, the overlap metric is calculated for `nreps` combinations of samples from $p(\Theta_A|X)$ and $p(\Theta_B|X)$ which are supplied in `niche.par`.

See Swanson et al. (2015) for a detailed description of niche overlap and its calculation.

Value

Returns an array of size `c(nspecies, nspecies, nreps, nlevels)`, where `nlevels` is the number of alpha levels at which to calculate the overlap metric. For each of the last two dimensions of the output array, the first two dimensions form an `nspecies` by `nspecies` matrix giving each pairwise calculation of overlap metric between two species for given Θ_A, Θ_B, and `alpha`. In each of these matrices, Species A is along the rows of this matrix and Species B is along the columns.

References

Swanson, H.K., Lysy, M., Stasko, A.D., Power, M., Johnson, J.D., and Reist, J.D. "A new probabilistic method for quantifying n-dimensional ecological niches and niche overlap." *Ecology: Statistical Reports* 96:2 (2015): 318-324. doi: 10.1890/140235.1.

See Also

`overlap.plot()`, `niw.post()`, `niiw.post()`.

Examples

```
# fish data
data(fish)

# generate parameter draws from the "default" posteriors of each fish
nsamples <- 500
system.time({
  fish.par <- tapply(1:nrow(fish), fish$species,
                     function(ii) niw.post(nsamples = nsamples, X = fish[ii,2:4]))
})

# overlap calculation. use nsamples = nprob = 1e4 for more accurate results.
system.time({
  over <- overlap(fish.par, nreps = nsamples, nprob = nsamples,
                  alpha = c(.95, .99))
})

# posterior expectations of overlap metrics
over.mean <- apply(over*100, c(1:2, 4), mean)
round(over.mean)

# posterior 95% credible intervals of overlap metrics
over.cred <- apply(over*100, c(1:2, 4), quantile,
                   prob = c(.025, .975), na.rm = TRUE)
round(over.cred[,,,1]) # display alpha = .95 niche region
```

`overlap.plot` *Plot the overlap metric.*

Description

Plots the posterior distribution of the niche region overlap metric calculated for each pairwise combination of species.

Usage

```
overlap.plot(
  over.stat,
  nbreaks = 50,
  equal.axis = FALSE,
  species.names,
  col,
  mean.cred = TRUE,
  mean.cred.col = "green",
  xlab
)
```

Arguments

`over.stat`	An array with `dim(over.stat) = c(nspecies, nspecies, nreps)` containing `nreps` calculations of the overlap metric for each pair of species. See 'Details'.
`nbreaks`	Number of breaks in the histogram. Defaults to 50.
`equal.axis`	Logical. If `TRUE`, all histograms in a given column of the output (corresponding to different Species A for the same Species B) are plotted on the same range.
`species.names`	A vector of species names. Defaults to `dimnames(over.stat)[[1]]`.
`col`	A vector of the colours in which each species will be drawn.
`mean.cred`	Logical. If `TRUE`, vertical lines for mean and 95% credible intervals will be included in the historgram of each overlap metric.
`mean.cred.col`	Colour of the mean and credible interval lines in the histogram.
`xlab`	Optional plot title, located on the bottom. Default is no title.

Details

This function uses the overlap metric information in `over.stat` calculated by `overlap()` to create 2-dimensional plots of interspecific niche region overlap.

Value

Returns a series of histograms illustrating the probability of pairwise species overlap.

See Also

`overlap()`, `niw.post()`, `niiw.post()`.

Examples

```
# fish data
data(fish)

# parameter draws from the "default" posteriors of each fish
nsamples <- 500
system.time({
```

```
  fish.par <- tapply(1:nrow(fish), fish$species,
                     function(ii) niw.post(nsamples = nsamples,
                                           X = fish[ii,2:4]))
})

# overlap calculation
system.time({
  over <- overlap(fish.par, nreps = nsamples, nprob = nsamples,
                  alpha = c(.95, .99))
})

# overlap plot
clrs <- c("black", "red", "blue", "orange") # color for each species
ii <- 1 # which niche region alpha level to use
overlap.plot(over[,,,ii], col = clrs, mean.cred.col = "turquoise",
             xlab = paste0("Overlap Probability (%) -- Niche Region Size: ",
                           dimnames(over)[[4]][ii]))
```

`overlap.unif` *Overlap calculation for uniform niche regions.*

Description

Overlap calculation for uniform niche regions.

Usage

```
overlap.unif(muA, SigmaA, muB, SigmaB, alphaA = 0.95, alphaB = 0.95, nprob)

overlap.sphere(muA, sigmaA, muB, sigmaB, alphaA = 0.95, alphaB = 0.95)
```

Arguments

`muA, muB`	Mean of niche regions.
`SigmaA, SigmaB`	Variance matrix of elliptical niche regions.
`alphaA, alphaB`	Probabilistic size of niche regions.
`nprob`	Number of uniform draws from niche region `A`.
`sigmaA, sigmaB`	standard deviations (scalars) of spherical niche regions.

Details

The overlap between niche regions A and B is defined as $vol(A \cap B)/vol(A \cup B)$, where the hypervolume of an n-dimensional region S is $vol(S) = \int_S dx$. For elliptical niche regions, there are simple formulas for $vol(A)$ and $vol(B)$. Thus, we need only determine the volume of the intersection $vol(A \cap B)$, as the volume of the union is given by the formula $vol(A \cup B) = vol(A) + vol(B) - vol(A \cap B)$.

For spherical niche regions, $vol(A \cap B)$ has a closed-form expression (see 'References'). For elliptical regions, no such formula exists and a Monte Carlo method is used instead. That is, $vol(A \cap$

$B)$ is calculated by sampling uniformly from A, then multiplying $vol(A)$ by the fraction of sampled points which fall into B.

While the uniform overlap metric is invariant to permutation of niche regions A and B, the accuracy of the Monte Carlo calculation of $vol(A \cap B)$ is not: higher accuracy is obtained when a higher fraction of sampled points are in the opposite niche region. `overlap.unif()` does not attempt to determine for which region this is the case, though the choice can be informed by plotting the niche regions, e.g., with `niche.plot()`.

Value

A Monte Carlo estimate of the niche overlap for `overlap.unif()`, and an analytic calculation for `overlap.sphere()`.

References

Li, S. "Concise formulas for the area and volume of a hyperspherical cap." *Asian Journal of Mathematics & Statistics* 4.1 (2011): 66-70. doi: 10.3923/ajms.2011.66.70.

Examples

```
# spherical case: compare Monte Carlo method to analytic formula

d <- 2 # 2D example
mA <- rnorm(d)
mB <- rnorm(d)
sigA <- rexp(1)
SigA <- sigA^2 * diag(d)
sigB <- rexp(1)
SigB <- sigB^2 * diag(d)

# plot circles
ellA <- ellipse(mA, SigA)
ellB <- ellipse(mB, SigB)
plot(0, type = "n",
     xlim = range(ellA[,1], ellB[,1]),
     ylim = range(ellA[,2], ellB[,2]), xlab = "x", ylab = "y")
lines(ellA, col = "red")
lines(ellB, col = "blue")
legend("topright", legend = c("niche A", "niche B"),
       fill = c("red", "blue"), bg = "white")

# compare niche calculations
overlap.sphere(mA, sigA, mB, sigB)
overlap.unif(mA, SigA, mB, SigB, nprob = 1e5)
```

`rniw` *Random draws from a Normal-Inverse-Wishart distribution.*

Description

Generates random draws from a Normal-Inverse-Wishart (NIW) distribution. Can be used to compare prior to posterior parameter distributions.

Usage

```
rniw(n, lambda, kappa, Psi, nu)
```

Arguments

`n`	Number of samples to draw.
`lambda`	Location parameter. See 'Details'.
`kappa`	Scale parameter. See 'Details'.
`Psi`	Scale matrix. See 'Details'.
`nu`	Degrees of freedom. See 'Details'.

Details

The NIW distribution $p(\mu, \Sigma | \lambda, \kappa, \Psi, \nu)$ is defined as

$$\Sigma \sim W^{-1}(\Psi, \nu), \quad \mu | \Sigma \sim N(\lambda, \Sigma/\kappa).$$

Value

Returns a list with elements `mu` and `Sigma` of sizes `c(n,length(lambda))` and `c(nrow(Psi),ncol(Psi),n)`.

See Also

`rwish()`, `niw.mom()`, `niw.coeffs()`.

Examples

```
d <- 4 # number of dimensions
nu <- 7 # degrees of freedom
Psi <- crossprod(matrix(rnorm(d^2), d, d)) # scale
lambda <- rnorm(d)
kappa <- 2
n <- 1e4

niw.sim <- rniw(n, lambda, kappa, Psi, nu)

# diagonal elements of Sigma^{-1} are const * chi^2
S <- apply(niw.sim$Sigma, 3, function(M) diag(solve(M)))
```

```
ii <- 2
const <- solve(Psi)[ii,ii]
hist(S[ii,], breaks = 100, freq = FALSE,
     main = parse(text = paste0("\"Histogram of \"*(Sigma^{-1})[", ii,ii,"]")),
     xlab = parse(text = paste0("(Sigma^{-1})[", ii,ii,"]")))
curve(dchisq(x/const, df = nu)/const,
      from = min(S[ii,]), to = max(S[ii,]), col = "red", add = TRUE)

# elements of mu have a t-distribution
mu <- niw.sim$mu

ii <- 4
const <- sqrt(Psi[ii,ii]/(kappa*(nu-d+1)))
hist(mu[,ii], breaks = 100, freq = FALSE,
     main = parse(text = paste0("\"Histogram of \"*mu[", ii, "]")),
     xlab = parse(text = paste0("mu[", ii, "]")))
curve(dt((x-lambda[ii])/const, df = nu-d+1)/const, add = TRUE, col = "red")
```

`rwish` *Random draws from a Wishart (or Inverse-Wishart) distribution.*

Description

Generates a random samples from a Wishart distribution defined as $W(\Psi, \nu)$, or an Inverse-Wishart distribution defined as $W^{-1}(\Psi, \nu)$.

Usage

```
rwish(n, Psi, nu, inv = FALSE)
```

Arguments

`n`	Number of samples to draw.
`Psi`	Scale matrix.
`nu`	Degrees of freedom.
`inv`	Logical. Setting `inv = TRUE` returns random matrices from an Inverse-Wishart distribution. See 'Details'.

Details

Setting `inv = TRUE` replaces Ψ by Psi^{-1} and inverts the output random matrices, such that they are being generated from an Inverse-Wishart $W^{-1}(\Psi, \nu)$ distribution.

Value

Returns an array of Wishart (or Inverse-Wishart) draws of size `c(nrow(Psi),ncol(Psi),n)`.

See Also

`rniw()`

Examples

```
d <- 4 # number of dimensions
nu <- 7 # degrees of freedom
Psi <- crossprod(matrix(rnorm(d^2), d, d)) # scale matrix
n <- 1e4

Sigma <- rwish(n, Psi, nu)

# for any vector a, X = (a' Sigma a) has a const * chi^2 distribution
a <- rnorm(d)
X <- apply(Sigma, 3, function(S) crossprod(a, S %*% a))
const <- a %*% Psi %*% a

hist(X, breaks = 100, freq = FALSE,
     main = parse(text = "\"Histogram of \"*X==a*minute*Sigma*a"),
     xlab = parse(text = "X==a*minute*Sigma*a"))
curve(dchisq(x/const, df = nu)/const,
      from = min(X), to = max(X), col = "red", add = TRUE)
```

Index