Package 'dsm'

July 4, 2017

Maintainer Laura Marshall <lhm@st-andrews.ac.uk>

License GPL (>= 2)

Title Density Surface Modelling of Distance Sampling Data

LazyLoad yes

Author David L. Miller, Eric Rexstad, Louise Burt, Mark V. Bravington, Sharon Hedley.

Description Density surface modelling of line transect data. A Generalized Additive Model-based approach is used to calculate spatially-explicit estimates of animal abundance from distance sampling (also presence/absence and strip transect) data. Several utility functions are provided for model checking, plotting and variance estimation.

Version 2.2.15

URL http://github.com/DistanceDevelopment/dsm

BugReports https://github.com/DistanceDevelopment/dsm/issues

Depends R (>= 3.0), mgcv (>= 1.7), mrds (>= 2.1.16), numDeriv

Imports nlme, ggplot2, statmod

Suggests Distance, tweedie

RoxygenNote 5.0.1

NeedsCompilation no

Repository CRAN

Date/Publication 2017-07-03 23:15:15 UTC

R topics documented:

`dsm-package` *Density surface modelling*

Description

`dsm` implements spatial models for distance sampling data.

Details

Further information on distance sampling methods and example code is available at `http://distancesampling.org/R/`.

For help with distance sampling and this package, there is a Google Group `https://groups.google.com/forum/#!forum/distance-sampling`.

A example analysis is available at `http://distancesampling.org/R/vignettes/mexico-analysis.html`.

block.info.per.su *Find the block information*

Description

Takes the transect data and works out how many blocks of a given size (in segment terms) fit into each.

Usage

```
block.info.per.su(block.size, data, name.su)
```

Arguments

block.size	number of segments per block
data	data used to build the model
name.su	names of the sampling units (ie. transects)

Value

a data.frame with the following columns

name	the sample unit name (e.g. transect label)
num.seg	number of segments in that transect
num.block	number of blocks available
start.block	block # for first block
end.block	block # for last block
num.req	number of blocks needed for the unit

check.cols *Check column names exist*

Description

Internal function to check that supplied ‘data.frames‘ have the correct columns and checks that sample labels are all unique.

Usage

```
check.cols(ddf.obj, segment.data, observation.data, strip.width, segment.area)
```

Arguments

ddf.obj	a ddf object from 'mrds'
segment.data	segment data as defined in dsm
observation.data	observation data as defined in dsm
strip.width	strip width if strip transects are being used
segment.area	area of segments

Value

nothing, but throws an error if something went wrong

Author(s)

David Lawrence Miller

---

dsm *Fit a density surface model to segment-specific estimates of abundance or density.*

Description

Fits a density surface model (DSM) to detection adjusted counts from a spatially-referenced distance sampling analysis. dsm takes observations of animals, allocates them to segments of line (or strip transects) and optionally adjusts the counts based on detectability using a supplied detection function model. A generalized additive model, generalized mixed model or generalized linear model is then used to model these adjusted counts based on a formula involving environmental covariates.

Usage

```
dsm(formula, ddf.obj, segment.data, observation.data, engine = "gam",
  convert.units = 1, family = quasipoisson(link = "log"), group = FALSE,
  gamma = 1.4, control = list(keepData = TRUE), availability = 1,
  strip.width = NULL, segment.area = NULL, weights = NULL,
  transect = "line", method = "REML", ...)
```

Arguments

formula	formula for the surface. This should be a valid glm/gam/gamm formula. See "Details", below, for how to define the response.
ddf.obj	result from call to ddf or ds. If ddf.obj is NULL then strip transects are assumed.
segment.data	segment data, see dsm-data.
observation.data	observation data, see dsm-data.

engine	which fitting engine should be used for the DSM (glm/gam/gamm/bam).
convert.units	conversion factor to multiply the area of the segments by. See 'Units' below.
family	response distribution (popular choices include quasipoisson, Tweedie/tw and negbin/nb). Defaults to quasipossion.
group	if TRUE the abundance of groups will be calculated rather than the abundance of individuals. Setting this option to TRUE is equivalent to setting the size of each group to be 1.
gamma	parameter to gam() set to a value of 1.4 (from advice in Wood (2006)) such that the gam() is inclined to not 'overfit' when GCV is used to select the smoothing parameter (ignored for REML, see link{gam} for further details).
control	the usual control argument for a gam; keepData must be TRUE for variance estimation to work (though this option cannot be set for GLMs or GAMMs.
availability	an availability bias used to scale the counts/estimated counts by. If we have N animals in a segment, then N/availability will be entered into the model. Uncertainty in the availability is not handled at present.
strip.width	if ddf.obj, above, is NULL, then this is where the strip width is specified (i.e. for a strip transect survey). This is sometimes (and more correctly) referred to as the half-width, i.e. right truncation minus left truncation.
segment.area	if 'NULL' (default) segment areas will be calculated by multiplying the 'Effort' column in 'segment.data' by the (right minus left) truncation distance for the 'ddf.obj' or by 'strip.width'. Alternatively a vector of segment areas can be provided (which must be the same length as the number of rows in 'segment.data') or a character string giving the name of a column in 'segment.data' which contains the areas. If segment.area is specified it takes precident.
weights	weights for each observation used in model fitting. The default, weights=NULL, weights each observation by its area (see Details). Setting a scalar value (e.g. weights=1) all observations are equally weighted.
transect	type of transect ("line", the default or "point"). This is overridden by the detection function transect type, this is usually only necessary when no detection function is specified.
method	The smoothing parameter estimation method. Default is "REML", using RE-stricted Maximum Likelihood. See gam for other options. Ignored for engine="glm".
...	anything else to be passed straight to glm/gam/gamm/bam.

Details

The response (LHS of 'formula') can be one of the following:

n, count, N	count in each segment
Nhat, abundance.est	estimated abundance per segment, estimation is via a Horvitz-Thompson estimator. This sh
presence	interpret the data as presence/absence (remember to change the family argument to binom
D, density, Dhat, density.est	density per segment

The offset used in the model is dependent on the response:

count	area of segment multiplied by average probability of detection in the segment
estimated count	area of the segment
presence	zero
density	zero

In the latter two cases (density and presence estimation) observations can be weighted by segment areas via the `weights=` argument. By default (`weights=NULL`), when density or presence are estimated the weights are set to the segment areas (using `segment.area` or by calculating `2*`(strip width)`*Effort`) Alternatively `weights=1` will set the weights to all be equal. A third alternative is to pass in a vector of length equal to the number of segments, containing appropriate weights.

Value

a `glm`/`gam`/`gamm` object, with an additional element, `ddf` which holds the detection function object.

Units

It is often the case that distances are collected in metres and segment lengths are recorded in kilometres. `dsm` allows you to provide a conversation factor (`convert.units`) to multiply the areas by. For example: if distances are in metres and segment lengths are in kilometres setting `convert.units=1000` will lead to the analysis being in metres. Setting `convert.units=1/1000` will lead to the analysis being in kilometres. The conversion factor will be applied to 'segment.area' if that is specified.

Large models

For large models, `engine="bam"` with `method="fREML"` may be useful. Models specified for `bam` should be as `gam`. READ `bam` before using this option; this option is considered EXPERIMENTAL at the moment. In particular note that the default basis choice (thin plate regression splines) will be slow and that in general fitting is less stable than when using `gam`. For negative binomial response, theta must be specified when using `bam`.

Author(s)

David L. Miller

References

Hedley, S. and S. T. Buckland. 2004. Spatial models for line transect sampling. JABES 9:181-199.

Miller, D. L., Burt, M. L., Rexstad, E. A., Thomas, L. (2013), Spatial models for distance sampling data: recent developments and future directions. Methods in Ecology and Evolution, 4: 1001-1010. doi: 10.1111/2041-210X.12105 (Open Access, available at http://onlinelibrary.wiley.com/doi/10.1111/2041-210X.12105/abstract)

Wood, S.N. 2006. Generalized Additive Models: An Introduction with R. CRC/Chapman & Hall.

Examples

```
## Not run:
library(Distance)
library(dsm)

# load the Gulf of Mexico dolphin data (see ?mexdolphins)
data(mexdolphins)

# fit a detection function and look at the summary
hr.model <- ds(distdata, max(distdata$distance),
               key = "hr", adjustment = NULL)
summary(hr.model)

# fit a simple smooth of x and y to counts
mod1 <- dsm(count~s(x,y), hr.model, segdata, obsdata)
summary(mod1)

# predict over a grid
mod1.pred <- predict(mod1, preddata, preddata$area)

# calculate the predicted abundance over the grid
sum(mod1.pred)

# plot the smooth
plot(mod1)

## End(Not run)
```

`dsm-data` *Data format for DSM*

Description

Two `data.frames` must be provided to `dsm`. They are referred to as `observation.data` and `segment.data`.

Details

The `segment.data` table has the sample identifiers which define the segments, the corresponding effort (line length) expended and the environmental covariates that will be used to model abundance/density. `observation.data` provides a link table between the observations used in the detection function and the samples (segments), so that we can aggregate the observations to the segments (i.e. `observation.data` is a "look-up table" between the observations and the segments).

`observation.data` - the observation `data.frame` must have (at least) the following columns:

`object`	unique object identifier
`Sample.Label`	the identifier for the segment that the observation occurred in
`size`	the size of each observed group (e.g 1 if all animals occurred individually)
`distance`	distance to observation

One can often also use `observation.data` to fit a detection function (so additional columns for detection function covariates are allowed in this table).

`segment.data`: the segment `data.frame` must have (at least) the following columns:

`Effort`	the effort (in terms of length of the segment)
`Sample.Label`	identifier for the segment (unique!)
???	environmental covariates, for example location (projected latitude and longitude), and other relevant covariat

---

`dsm.cor` *Check for autocorrelation in residuals*

Description

Once a DSM has been fitted to data, this function can be used to check for autocorrelation in the residuals.

Usage

```
dsm.cor(dsm.obj, Transect.Label = "Transect.Label",
  Segment.Label = "Segment.Label", max.lag = 10,
  resid.type = "scaled.pearson", fun = cor, ylim = c(0, 1),
  subset = "all", ...)
```

Arguments

`dsm.obj`	a fitted dsm object.
`Transect.Label`	label for the transect (default: `Transect.Label`). Using different labels can be useful when transects are split over geographical features or when transects are surveyed multiple times.
`Segment.Label`	label for the segments (default: `Segment.Label`).The result of calling `order()` must make sense.
`max.lag`	maximum lag to calulate at.
`resid.type`	the type of residuals used, see `residuals.gam` and `residuals.gam`. Defaults to `"scaled.pearson"` in the GAM case and `"normalized"` in the GAMM case (which are equivalent).
`fun`	the function to use, by default `cor`, must take two column vectors as arguments.
`ylim`	user defined limits in y direction.
`subset`	which subset of the data should the correlation function be calculated on?
`...`	other options to pass to `plot`.

Value

a plot or a vector of `fun` applied at the lags.

Details

Within each `Transect.Label`, segments will be sorted according to their `Segment.Labels`. This may require some time to get right for your particular data. If one has multiple surveys where transects are revisited, for example, one may want to make `Transect.Label` a unique transect-survey id. Neither label need to be included in the model, they must just be present in the `$data` field in the model. This usually means that they have to be in the segment data passed to `dsm`.

The current iteration of this function will only plot correlations nicely, other things are up to you but you can get the function to return the data (by assigning the result to an object).

If there are NA values in the residuals then the correlogram will not be calculated. This usually occurs due to NA values in the covariates (so the smoother will not have fitted values there). Code like 'any(is.na(dsm.obj$data))' might be helpful.

Author(s)

David L. Miller

Examples

```
library(Distance)
library(dsm)

# load the data, see ?mexdolphins
data(mexdolphins)

# fit a model
hr.model <- ds(distdata, max(distdata$distance),
               key = "hr", adjustment = NULL)
mod1<-dsm(count~s(x,y), hr.model, segdata, obsdata)

# look at lag 1 differences up to a maximum of lag 9, using deviance
# residuals
dsm.cor(mod1, resid.type="deviance", max.lag=9,
        Segment.Label="Sample.Label")
```

`dsm.var.gam` *Prediction variance estimation assuming independence*

Description

If one is willing to assume the the detection function and spatial model are independent, this function will produce estimates of variance of predictions of abundance, using the result that squared coefficients of variation will add.

Usage

```
dsm.var.gam(dsm.obj, pred.data, off.set, seglen.varname = "Effort",
  type.pred = "response")
```

Arguments

dsm.obj	a model object returned from running dsm.
pred.data	either: a single prediction grid or list of prediction grids. Each grid should be a data.frame with the same columns as the original data.
off.set	a a vector or list of vectors with as many elements as there are in pred.data. Each vector is as long as the number of rows in the corresponding element of pred.data. These give the area associated with each prediction cell. If a single number is supplied it will be replicated for the length of pred.data.
seglen.varname	name for the column which holds the segment length (default value "Effort").
type.pred	should the predictions be on the "response" or "link" scale? (default "response").

Details

This is based on dsm.var.prop taken from code by Mark Bravington and Sharon Hedley.

Value

a list with elements

model	the fitted model object
pred.var	variance of the regions given in pred.data.
bootstrap	logical, always FALSE
model	the fitted model with the extra term
dsm.object	the original model, as above

Author(s)

David L. Miller

Examples

```
## Not run:
 library(Distance)
 library(dsm)

 # load the Gulf of Mexico dolphin data (see ?mexdolphins)
 data(mexdolphins)

 # fit a detection function and look at the summary
 hr.model <- ds(distdata, max(distdata$distance),
                key = "hr", adjustment = NULL)
 summary(hr.model)

 # fit a simple smooth of x and y
 mod1 <- dsm(count~s(x, y), hr.model, segdata, obsdata)

 # Calculate the variance
 # this will give a summary over the whole area in mexdolphins$preddata
 mod1.var <- dsm.var.gam(mod1, preddata, off.set=preddata$area)
```

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

`dsm.var.movblk` *Variance estimation via parametric moving block bootstrap*

Description

Estimate the variance in abundance over an area using a moving block bootstrap. Two procedures are implemented, one incorporating detection function uncertainty, one not.

Usage

```
dsm.var.movblk(dsm.object, pred.data, n.boot, block.size, off.set,
  ds.uncertainty = FALSE, samp.unit.name = "Transect.Label",
  progress.file = NULL, bs.file = NULL, bar = TRUE)
```

Arguments

`dsm.object`	object returned from `dsm`.
`pred.data`	either: a single prediction grid or list of prediction grids. Each grid should be a `data.frame` with the same columns as the original data.
`n.boot`	number of bootstrap resamples.
`block.size`	number of segments in each block.
`off.set`	a a vector or list of vectors with as many elements as there are in `pred.data`. Each vector is as long as the number of rows in the corresponding element of `pred.data`. These give the area associated with each prediction cell. If a single number is supplied it will be replicated for the length of `pred.data`.
`ds.uncertainty`	incorporate uncertainty in the detection function? See Details, below. Note that this feature is EXPERIMENTAL at the moment.
`samp.unit.name`	name sampling unit to resample (default 'Transect.Label').
`progress.file`	path to a file to be used (usually by Distance) to generate a progress bar (default `NULL` – no file written).
`bs.file`	path to a file to store each boostrap round. This stores all of the bootstrap results rather than just the summaries, enabling outliers to be detected and removed. (Default `NULL`).
`bar`	should a progress bar be printed to screen? (Default `TRUE`).

Details

Setting ds.uncertainty=TRUE will incorporate detection function uncertainty directly into the bootstrap. This is done by generating observations from the fitted detection function and then re-fitting a new detection function (of the same form), then calculating a new effective strip width. Rejection sampling is used to generate the observations (except in the half-normal case) so the procedure can be rather slow. Note that this is currently not supported with covariates in the detection function.

Setting ds.uncertainty=FALSE will incorporate detection function uncertainty using the delta method. This assumes that the detection function and the spatial model are INDEPENDENT. This is probably not reasonable.

Examples

```
## Not run:
library(Distance)
library(dsm)

# load the Gulf of Mexico dolphin data (see ?mexdolphins)
data(mexdolphins)

# fit a detection function and look at the summary
hr.model <- ds(distdata, max(distdata$distance),
               key = "hr", adjustment = NULL)
summary(hr.model)

# fit a simple smooth of x and y
mod1 <- dsm(count~s(x, y), hr.model, segdata, obsdata)
summary(mod1)

# calculate the variance by 500 moving block bootstraps
mod1.movblk <- dsm.var.movblk(mod1, preddata, n.boot = 500,
   block.size = 3, samp.unit.name = "Transect.Label",
   off.set = preddata$area,
   bar = TRUE, bs.file = "mexico-bs.csv", ds.uncertainty = TRUE)

## End(Not run)
```

---

dsm.var.prop *Prediction variance propagation for DSMs*

Description

To ensure that uncertainty from the detection function is correctly propagated to the final variance estimate of abundance, this function uses a method first detailed in Williams et al (2011).

Usage

```
dsm.var.prop(dsm.obj, pred.data, off.set, seglen.varname = "Effort",
  type.pred = "response")
```

Arguments

dsm.obj	a model object returned from running dsm.
pred.data	either: a single prediction grid or list of prediction grids. Each grid should be a data.frame with the same columns as the original data.
off.set	a a vector or list of vectors with as many elements as there are in pred.data. Each vector is as long as the number of rows in the corresponding element of pred.data. These give the area associated with each prediction cell. If a single number is supplied it will be replicated for the length of pred.data.
seglen.varname	name for the column which holds the segment length (default value "Effort").
type.pred	should the predictions be on the "response" or "link" scale? (default "response").

Details

The idea is to refit the spatial model but including an extra random effect. This random effect has zero mean and hence to effect on point estimates. Its variance is the Hessian of the detection function. Variance estimates then incorporate detection function uncertainty. Further mathematical details are given in the paper in the references below.

Many prediction grids can be supplied by supplying a list of data.frames to the function.

Note that this routine simply calls dsm_varprop. If you don't require multiple prediction grids, the other routine will probably be faster.

Value

a list with elements

model	the fitted model object
pred.var	variance of each region given in pred.data.
bootstrap	logical, always FALSE
pred.data	as above
off.set	as above
model	the fitted model with the extra term
dsm.object	the original model, as above
model.check	simple check of subtracting the coefficients of the two models to see if there is a large difference
deriv	numerically calculated Hessian of the offset

Diagnostics

The summary output from the function includes a simply diagnostic that shows the average probability of detection from the "original" fitted model (the model supplied to this function; column Fitted.model) and the probability of detection from the refitted model (used for variance propagation; column Refitted.model) along with the standard error of the probability of detection from the fitted model (Fitted.model.se), at the unique values of any covariates used in the detection function. If there are large differences between the probabilities of detection then there are potentially problems with the fitted model, the variance propagation or both. This can be because the fitted model does not account for enough of the variability in the data and in refitting the variance model accounts for this in the random effect.

Limitations

Note that this routine is only useful if a detection function has been used in the DSM. It cannot be used when the `Nhat`, `abundance.est` responses are used. Importantly this requires that if the detection function has covariates, then these do not vary within a segment (so, for example covariates like sex cannot be used).

Negative binomial models fitted using the `nb` family will give strange results (overly big variance estimates due to scale parameter issues) so `nb` models are automatically refitted with `negbin` (with a warning). It is probably worth refitting these models with `negbin` manually (perhaps giving a smallish range of possible values for the negative binomial parameter) to check that convergence was reached.

Author(s)

Mark V. Bravington, Sharon L. Hedley. Bugs added by David L. Miller.

References

Williams, R., Hedley, S.L., Branch, T.A., Bravington, M.V., Zerbini, A.N. and Findlay, K.P. (2011). Chilean Blue Whales as a Case Study to Illustrate Methods to Estimate Abundance and Evaluate Conservation Status of Rare Species. Conservation Biology 25(3), 526-535.

Examples

```
## Not run:
 library(Distance)
 library(dsm)

 # load the Gulf of Mexico dolphin data (see ?mexdolphins)
 data(mexdolphins)

 # fit a detection function
 df <- ds(distdata, max(distdata$distance),
          key = "hn", adjustment = NULL)

 # fit a simple smooth of x and y
 mod1 <- dsm(count~s(x, y), df, segdata, obsdata, family=tw())

 # Calculate the variance
 # this will give a summary over the whole area in mexdolphins$preddata
 mod1.var <- dsm.var.prop(mod1, preddata, off.set=preddata$area)
 summary(mod1.var)

## End(Not run)
```

dsm_varprop *Variance propagation for density surface models*

Description

Calculate the uncertainty in predictions from a fitted DSM, including uncertainty from the detection function.

Usage

```
dsm_varprop(model, newdata, trace = FALSE, var_type = "Vp")
```

Arguments

model	a fitted dsm
newdata	the prediction grid
trace	for debugging, see how the scale parameter estimation is going
var_type	which variance-covariance matrix should be used ("Vp" for variance-covariance conditional on smoothing parameter(s), "Vc" for unconditional). See gamObject for an details/explanation. If in doubt, stick with the default, "Vp".

Details

When we make predictions from a spatial model, we also want to know the uncertainty about that abundance estimate. Since density surface models are 2 (or more) stage models, we need to incorporate the uncertainty from the earlier stages (i.e. the detection function) into our "final" uncertainty estimate.

This function will refit the spatial model but include the Hessian of the offset as an extra term. Variance estimates using this new model can then be used to calculate the variance of predicted abundance estimates which incorporate detection function uncertainty. Importantly this requires that if the detection function has covariates, then these do not vary within a segment (so, for example covariates like sex cannot be used).

For more information on how to construct the prediction grid data.frame, newdata, see predict.dsm.

This routine is only useful if a detection function has been used in the DSM.

Note that we can use var_type="Vc" here (see gamObject), which is the variance-covariance matrix for the spatial model, corrected for smoothing parameter uncertainty. See Wood, Pya & S\"afken (2016) for more information.

Negative binomial models fitted using the nb family will give strange results (overly big variance estimates due to scale parameter issues) so nb models are automatically refitted with negbin (with a warning). It is probably worth refitting these models with negbin manually (perhaps giving a smallish range of possible values for the negative binomial parameter) to check that convergence was reached.

Value

a list with elements

old_model	fitted model supplied to the function as model
refit	refitted model object, with extra term
pred	point estimates of predictions at newdata
var	total variance calculated over all of newdata
ses	standard error for each prediction cell in newdata

Diagnostics

The summary output from the function includes a simply diagnostic that shows the average probability of detection from the "original" fitted model (the model supplied to this function; column Fitted.model) and the probability of detection from the refitted model (used for variance propagation; column Refitted.model) along with the standard error of the probability of detection from the fitted model (Fitted.model.se), at the unique values of any covariates used in the detection function. If there are large differences between the probabilities of detection then there are potentially problems with the fitted model, the variance propagation or both. This can be because the fitted model does not account for enough of the variability in the data and in refitting the variance model accounts for this in the random effect.

Author(s)

David L. Miller, based on code from Mark V. Bravington and Sharon L. Hedley.

References

Williams, R., Hedley, S.L., Branch, T.A., Bravington, M.V., Zerbini, A.N. and Findlay, K.P. (2011). Chilean Blue Whales as a Case Study to Illustrate Methods to Estimate Abundance and Evaluate Conservation Status of Rare Species. Conservation Biology 25(3), 526-535. Wood, S.N., Pya, N. and S\"afken, B. (2016) Smoothing parameter and model selection for general smooth models. Journal of the American Statistical Association, 1-45.

Examples

```
## Not run:
library(Distance)
library(dsm)

# load the Gulf of Mexico dolphin data (see ?mexdolphins)
data(mexdolphins)

# fit a detection function
df <- ds(distdata, max(distdata$distance),
         key = "hn", adjustment = NULL)

# fit a simple smooth of x and y
mod1 <- dsm(count~s(x, y), df, segdata, obsdata, family=tw())

# Calculate the variance
preddata$off.set <- preddata$area
mod1.varp <- dsm_varprop(mod1, preddata)
summary(mod1.varp)
```

```
# this will give a summary over the whole area in mexdolphins$preddata

## End(Not run)
```

`generate.ds.uncertainty` *Generate data from a fitted detection function*

Description

When `ds.uncertainty` is `TRUE`, this procedure generates data from the fitted detection function (assuming that it is correct).

Usage

```
generate.ds.uncertainty(ds.object)
```

Arguments

`ds.object`	a fitted detection function object (as returned by a call to `ddf.ds()`.

Note

This function changes the random number generator seed. To avoid any potential side-effects, use something like: `seed <- get(".Random.seed",envir=.GlobalEnv)` before running code and `assign(".Random.seed",seed,envir=.GlobalEnv)` after.

Author(s)

David L. Miller

`generate.mb.sample` *Moving block bootstrap sampler*

Description

Not usually used on its own, called from within `dsm.var.movblk`.

Usage

```
generate.mb.sample(num.blocks.required, block.size, which.blocks, dsm.data,
  unit.info, n.units)
```

Arguments

num.blocks.required	number of blocks that we need.
block.size	number of segments per block.
which.blocks	which blocks should be sampled.
dsm.data	the $data element of the result of a call to dsm.
unit.info	result of calling block.info.per.su.
n.units	number of sampling units.

Value

vector of log-residuals

---

latlong2km *Convert latitude and longitude to Northings and Eastings*

Description

Convert longitude and latitude co-ordinates to kilometres west-east and south-north from axes through (lon0,lat0) using the "spherical law of cosines".

Usage

```
latlong2km(lon, lat, lon0 = sum(range(lon))/2, lat0 = sum(range(lat))/2)
```

Arguments

lon	longitude
lat	latitude
lon0	longitude reference point (defaults to mean longitude)
lat0	latitude reference point (defaults to mean latitude)

Details

WARNING: This is an approximate procedure for converting between latitude/ longitude and Northing/Easting. Consider using projection conversions available in packages sp and rgdal for better results.

Value

list with elements km.e and km.n.

Author(s)

Simon N. Wood

make.soapgrid *Create a knot grid for the internal part of a soap film smoother.*

Description

This routine simply creates a grid of knots (in the correct format) to be used as in the "internal" part of the soap film smoother

Usage

```
make.soapgrid(bnd, n.grid)
```

Arguments

bnd	list with elements x and y which give the locations of the boundary vertices. The first and last elements should be the same.
n.grid	either one number giving the number of points along the x and y axes that should be used to create the grid, or a vector giving the number in the x direction, then y direction.

Value

a list with elements x and y, containing the knot locations.

Author(s)

David L Miller

matrixnotposdef.handler *Handler to suppress the "matrix not positive definite" warning*

Description

Internal function to suppress an annoying warnings from chol()

Usage

```
matrixnotposdef.handler(w)
```

Arguments

w	a warning

Details

See: https://stat.ethz.ch/pipermail/r-help/2012-February/302407.html See: http://romainfrancois.blog.free.fr/index.php?post/specific-warnings

Value

not a warning if the warning was "matrix not positive definite" or "the matrix is either rank-deficient or indefinite"

Author(s)

David L. Miller

`mexdolphins` *Pan-tropical spotted dolphins in the Gulf of Mexico*

Description

Data from a combination of several NOAA shipboard surveys conducted on pan-tropical spotted dolphins in the Gulf of Mexico. 47 observations of groups of dolphins The group size was recorded, as well as the Beaufort sea state at the time of the observation. Coordinates for each observation and bathymetry data were also available as covariates for the analysis. A complete example analysis (and description of the data) is provided at http://distancesampling.org/R/vignettes/mexico-analysis.html.

References

Halpin, P.N., A.J. Read, E. Fujioka, B.D. Best, B. Donnelly, L.J. Hazen, C. Kot, K. Urian, E. LaBrecque, A. Dimatteo, J. Cleary, C. Good, L.B. Crowder, and K.D. Hyrenbach. 2009. OBIS-SEAMAP: The world data center for marine mammal, sea bird, and sea turtle distributions. Oceanography 22(2):104-115

NOAA Southeast Fisheries Science Center. 1996. Report of a Cetacean Survey of Oceanic and Selected Continental Shelf Waters of the Northern Gulf of Mexico aboard NOAA Ship Oregon II (Cruise 220)

`offsets` *Offsets*

Description

This will be documentation on calculating offsets.

plot.dsm *Plot a density surface model.*

Description

See plot.gam.

Usage

```
## S3 method for class 'dsm'
plot(x, ...)
```

Arguments

x	a dsm object
...	other arguments passed to plot.gam.

Value

a plot!

Author(s)

David L. Miller

See Also

dsm plot.gam

plot.dsm.var *Create plots of abundance uncertainty*

Description

Note that the prediction data set must have x and y columns even if these were not used in the model.

Usage

```
## S3 method for class 'dsm.var'
plot(x, poly = NULL, limits = NULL, breaks = NULL,
  legend.breaks = NULL, xlab = "x", ylab = "y", observations = TRUE,
  plot = TRUE, boxplot.coef = 1.5, x.name = "x", y.name = "y",
  gg.grad = NULL, ...)
```

Arguments

x	a dsm.var object
poly	a list or data.frame with columns x and y, which gives the coordinates of a polygon to draw. It may also optionally have a column group, if there are many polygons.
limits	limits for the fill colours
breaks	breaks for the colour fill
legend.breaks	breaks as they should be displayed
xlab	label for the x axis
ylab	label for the y axis
observations	should observations be plotted?
plot	actually plot the map, or just return a ggplot2 object?
boxplot.coef	control trimming (as in summary.dsm.var), only has an effect if the bootstrap file was saved.
x.name	name of the variable to plot as the x axis.
y.name	name of the variable to plot as the y axis.
gg.grad	optional ggplot gradient object.
...	any other arguments

Value

a plot

Details

In order to get plotting to work with dsm.var.prop and dsm.var.gam, one must first format the data correctly since these functions are designed to compute very general summaries. One summary is calculated for each element of the list pred supplied to dsm.var.prop and dsm.var.gam.

For a plot of uncertainty over a prediction grid, pred (a data.frame), say, we can create the correct format by simply using `pred.new <- split(pred,1:nrow(pred))`.

Author(s)

David L. Miller

See Also

dsm.var.prop, dsm.var.gam, dsm.var.movblk

plot_pred_by_term *Spatially plot predictions per model term*

Description

Plot the effect of each smooth in the model spatially. For each term in the model, plot its effect in space. Plots are made on the same scale, so that the relative influence of each smooth can be seen.

Usage

```
plot_pred_by_term(dsm.obj, data, location_cov = c("x", "y"))
```

Arguments

dsm.obj	fitted dsm object
data	data to use to plot (often the same as the precition grid), data should also include width and height columns for plotting
location_cov	which covariates to plot by (usually 2, spatial covariates, by default =c("x","y")

Value

a ggplot2 plot

Author(s)

David L Miller (idea taken from inlabru)

Examples

```
## Not run:
library(Distance)
library(dsm)

# load the Gulf of Mexico dolphin data and fit a model
data(mexdolphins)
hr.model <- ds(distdata, max(distdata$distance),
               key = "hr", adjustment = NULL)
mod1 <- dsm(count~s(x,y) + s(depth), hr.model, segdata, obsdata)

preddata$width <- preddata$height <- sqrt(preddata$area)

# make the plot
plot_pred_by_term(mod1, preddata, c("x","y"))

# better plot would be
# library(viridis)
# plot_pred_by_term(mod1, preddata, c("x","y")) + scale_fill_viridis()

## End(Not run)
```

predict.dsm *Predict from a fitted density surface model*

Description

Make predictions outside (or inside) the covered area.

Usage

```
## S3 method for class 'dsm'
predict(object, newdata = NULL, off.set = NULL,
  type = "response", ...)
```

Arguments

object	a fitted dsm object as produced by dsm().
newdata	spatially referenced covariates e.g. altitude, depth, distance to shore, etc. Covariates in the data.frame must have names *identical* to variable names used in fitting the DSM.
off.set	area of each of the cells in the prediction grid. Should be in the same units as the segments/distances given to dsm. Ignored if there is already a column in newdata called off.set.
type	what scale should the results be on. The default is "response", see predict.gam for an explanation of other options (usually not necessary).
...	any other arguments passed to predict.gam.

Details

If newdata is not supplied, predictions are made for the data that built the model. Note that the order of the results will not necessarily be the same as the segdata (segment data) data.frame that was supplied (it will be sorted by the Segment.Label field).

Value

predicted values on the response scale (density/abundance).

Author(s)

David L. Miller

See Also

predict.gam dsm.var.gam dsm.var.prop dsm.var.movblk

print.dsm *Print a description of a density surface model object*

Description

This method just gives a short description of the fitted model. Use the summary.dsm method for more information.

Usage

```
## S3 method for class 'dsm'
print(x, ...)
```

Arguments

x	a dsm object
...	unspecified and unused arguments for S3 consistency

Author(s)

David L. Miller

See Also

summary.ds

print.dsm.var *Print a description of a density surface model variance object*

Description

This method only provides a short summary, use the summary.dsm.var method for information.

Usage

```
## S3 method for class 'dsm.var'
print(x, ...)
```

Arguments

x	a dsm variance object
...	unspecified and unused arguments for S3 consistency

Author(s)

David L. Miller

See Also

summary.dsm.var

`print.dsm_varprop` *Print a description of a density surface model variance object*

Description

This method only provides a short summary, see summary.dsm_varprop.

Usage

```
## S3 method for class 'dsm_varprop'
print(x, ...)
```

Arguments

x	a dsm variance object
...	unspecified and unused arguments for S3 consistency

Author(s)

David L. Miller

See Also

summary.dsm_varprop

`print.summary.dsm.var` *Print summary of density surface model variance object*

Description

See summary.dsm.var for information.

Usage

```
## S3 method for class 'summary.dsm.var'
print(x, ...)
```

Arguments

x	a summary of dsm variance object
...	unspecified and unused arguments for S3 consistency

Author(s)

David L. Miller

See Also

summary.dsm.var

---

print.summary.dsm_varprop *Print summary of density surface model variance object*

Description

See summary.dsm_varprop for information.

Usage

```
## S3 method for class 'summary.dsm_varprop'
print(x, ...)
```

Arguments

x	a summary of dsm variance object
...	unspecified and unused arguments for S3 consistency

Author(s)

David L. Miller

See Also

summary.dsm.var

---

rqgam.check *Randomised quantile residuals check plot for GAMs/DSMs*

Description

Reproduces the "Resids vs. linear pred" plot from gam.check but using randomised quantile residuals, a la Dunn and Smyth (1996). Checks for heteroskedasticity as as usual, looking for "funnel"-type structures in the points, which is much easier with randomised quantile residuals than with deviance residuals, when your model uses a count distribution as the response.

Usage

```
rqgam.check(gam.obj, ...)
```

Arguments

gam.obj	a gam, glm or dsm object.
...	arguments passed on to all plotting functions

Details

Note that this function only works with negative binomial and Tweedie response distributions.

Earlier versions of this function produced the full gam.check output, but this was confusing as only one of the plots was really usedul. Checks of k are not computed, these need to be done using gam.check.

Value

just plots!

Author(s)

Based on code provided by Natalie Kelly, bugs added by Dave Miller

Examples

```
library(Distance)
library(dsm)
library(tweedie)

# load the Gulf of Mexico dolphin data (see ?mexdolphins)
data(mexdolphins)

# fit a detection function and look at the summary
hr.model <- ds(distdata, max(distdata$distance),
               key = "hr", adjustment = NULL)

# fit a simple smooth of x and y with a Tweedie response with estimated
#  p parameter
mod1 <- dsm(count~s(x, y), hr.model, segdata, obsdata, family=tw())
rqgam.check(mod1)
```

---

summary.dsm *Summarize a fitted density surface model*

Description

Gives a brief summary of a fitted dsm object.

Usage

```
## S3 method for class 'dsm'
summary(object, ...)
```

Arguments

object	a dsm object
...	other arguments passed to summary.gam.

Value

a summary object

Author(s)

David L. Miller

See Also

dsm

---

summary.dsm.var *Summarize the variance of a density surface model*

Description

Gives a brief summary of a fitted dsm variance object.

Usage

```
## S3 method for class 'dsm.var'
summary(object, alpha = 0.05, boxplot.coef = 1.5,
  bootstrap.subregions = NULL, ...)
```

Arguments

object	a dsm.var object
alpha	alpha level for confidence intervals (default 0.05 to give a 95% confidence internal, i.e. we generate 100*c(alpha/2, 1-alpha/2) confidence intervals)
boxplot.coef	the value of coef used to calculate the outliers see boxplot.
bootstrap.subregions	list of vectors of logicals or indices for subregions for which variances need to be calculated (only for bootstraps (see dsm.var.prop for how to use subregions with variance propagation).
...	unused arguments for S3 compatibility

Value

a summary object

Author(s)

David L. Miller

See Also

dsm.var.movblk dsm.var.prop

`summary.dsm_varprop` *Summarize the variance of a density surface model*

Description

Gives a brief summary of a fitted `dsm_varprop` variance object.

Usage

```
## S3 method for class 'dsm_varprop'
summary(object, alpha = 0.05, ...)
```

Arguments

`object`	a `dsm.var` object
`alpha`	alpha level for confidence intervals (default 0.05 to give a 95% confidence internal)
`...`	unused arguments for S3 compatibility

Value

a summary object

Author(s)

David L. Miller

See Also

dsm_varprop summary.dsm.var

trim.var *Trimmed variance*

Description

Trim the variance estimates from the bootstrap. This is defined as the percentage defined as amount necessary to bring median and trimmed mean within 8

Usage

```
trim.var(untrimmed.bootstraps, boxplot.coef = 1.5)
```

Arguments

untrimmed.bootstraps	(usually the $study.area.total element of a returned dsm bootstrap object.
boxplot.coef	the value of coef used to calculate the outliers see boxplot.

Details

Code originally by Louise Burt.

Value

trimmed variance

Author(s)

Louise Burt

vis.concurvity *Visualise concurvity between terms in a GAM*

Description

Plot measures of how much one term in the model could be explained by another. When values are high, one should consider re-running variable selection with one of the offending variables removed to check for stability in term selection.

Usage

```
vis.concurvity(model, type = "estimate")
```

Arguments

model	fitted model
type	concurvity measure to plot, see concurvity

Details

These methods are considered somewhat experimental at this time. Consult concurvity for more information on how concurvity measures are calculated.

Author(s)

David L Miller

Examples

```
## Not run:
library(Distance)
library(dsm)

# load the Gulf of Mexico dolphin data (see ?mexdolphins)
data(mexdolphins)

# fit a detection function and look at the summary
hr.model <- ds(distdata, max(distdata$distance),
               key = "hr", adjustment = NULL)

# fit a simple smooth of x and y to counts
mod1 <- dsm(count~s(x,y)+s(depth), hr.model, segdata, obsdata)

# visualise concurvity using the "estimate" metric
vis.concurvity(mod1)

## End(Not run)
```

Index