Package ‘S4DM’

May 11, 2026

Title Small Sample Size Species Distribution Modeling

Version 0.0.2

Description

Implements a set of distribution modeling methods that are suited to species with small sample sizes (e.g., poorly sampled species or rare species). While these methods can also be used on well-sampled taxa, they are united by the fact that they can be utilized with relatively few data points. More details on the currently implemented methodologies can be found in Maitner et al. (2026) <doi:10.1002/ecog.08112>, Drake and Richards (2018) <doi:10.1002/ecs2.2373>, Drake (2015) <doi:10.10
00202.1>.

Depends R (>= 4.1.0)

License MIT + file LICENSE

Encoding UTF-8

LazyData true

VignetteBuilder knitr

RoxygenNote 7.3.3

Imports corpcor, densratio (>= 0.3.1), flexclust, geometry, kernlab, maxnet, mvtnorm, np, pROC, robust, rvinecopulib, sf, terra, dplyr, Rdpack

Suggests geodata, BIEN, ggplot2, tidyterra, knitr, testthat, rmarkdown

RdMacros Rdpack

NeedsCompilation no

Author Brian S. Maitner [aut, cre] (ORCID:
<`https://orcid.org/0000-0002-2118-9880`>),
Robert L. Richards [aut],
Ben S. Carlson [aut],
John M. Drake [aut],
Cory Merow [aut]

Maintainer Brian S. Maitner <`bmaitner@usf.edu`>

Repository CRAN

Date/Publication 2026-05-11 06:50:13 UTC

Contents

`ensemble_range_map` *Generate ensemble predictions from S4DM range maps*

Description

This function evaluates model quality and creates an ensemble of the model outputs. This function uses 5-fold, spatially stratified, cross-validation to evaluate distribution model quality. Ensembles of algorithms spanning the sensitivity-specificity gradient can improve predictions for poorly sampled species (Maitner et al. 2026).

Usage

```
ensemble_range_map(
  occurrences,
  env,
  method = NULL,
  presence_method = NULL,
  background_method = NULL,
  bootstrap = "none",
  bootstrap_reps = 100,
  quantile = 0.05,
  constraint_regions = NULL,
  background_buffer_width = NULL,
  ...
)
```

Arguments

`occurrences`	Presence coordinates in long,lat format.
`env`	Environmental SpatRaster(s)
`method`	Optional. If supplied, both presence and background density estimation will use this method.
`presence_method`	Optional. Method for estimation of presence density.
`background_method`	Optional. Method for estimation of background density.
`bootstrap`	Character. One of "none" (the default, no bootstrapping), "numbag" (presence function is bootstrapped), or "doublebag" (presence and background functions are bootstrapped).
`bootstrap_reps`	Integer. Number of bootstrap replicates to use (default is 100)
`quantile`	Quantile to use for thresholding. Default is 0.05 (5 pct training presence). Set to 0 for minimum training presence (MTP).
`constraint_regions`	See get_env_bg documentation
`background_buffer_width`	Numeric or NULL. Width (meters or map units) of buffer to use to select background environment. If NULL, uses max dist between nearest occurrences.
`...`	Additional parameters passed to internal functions.

Details

Current plug-and-play methods include: "gaussian", "kde","vine","rangebagging", "lobagoc", and "none". Current density ratio methods include: "ulsif", "rulsif".

Value

List object containing elements (1) spatRaster ensemble layer showing the proportion of maps that are included in the range across the ensemble, (2) spatRasters for individual models, and (3) model quality information.

Note

Either `method` or both `presence_method` and `background_method` must be supplied.

Examples

```


# load in sample data

 library(S4DM)
 library(terra)

 # occurrence points
```

```
    data("sample_points")
    occurrences <- sample_points

  # environmental data
    env <- rast(system.file('ex/sample_env.tif', package="S4DM"))

  # rescale the environmental data

    env <- scale(env)

ensemble <- ensemble_range_map(occurrences = occurrences,
                               env = env,
                               method = NULL,
                               presence_method = c("gaussian", "kde"),
                               background_method = "gaussian",
                               quantile = 0.05,
                               background_buffer_width = 100000  )
```

`evaluate_range_map` *Evaluate S4DM range map quality*

Description

This function uses 5-fold, spatially stratified, cross-validation to evaluate distribution model quality.

Usage

```
evaluate_range_map(
  occurrences,
  env,
  method = NULL,
  presence_method = NULL,
  background_method = NULL,
  bootstrap = "none",
  bootstrap_reps = 100,
  quantile = 0.05,
  constraint_regions = NULL,
  background_buffer_width = NULL,
  standardize_preds = TRUE,
  ...
)
```

Arguments

`occurrences`	Presence coordinates in long,lat format.
`env`	Environmental SpatRaster(s)

`method`	Optional. If supplied, both presence and background density estimation will use this method.
`presence_method`	Optional. Method for estimation of presence density.
`background_method`	Optional. Method for estimation of background density.
`bootstrap`	Character. One of "none" (the default, no bootstrapping), "numbag" (presence function is bootstrapped), or "doublebag" (presence and background functions are bootstrapped).
`bootstrap_reps`	Integer. Number of bootstrap replicates to use (default is 100)
`quantile`	Quantile to use for thresholding. Default is 0.05 (5 pct training presence). Set to 0 for minimum training presence (MTP).
`constraint_regions`	See get_env_bg documentation
`background_buffer_width`	Numeric or NULL. Width (meters or map units) of buffer to use to select background environment. If NULL, uses max dist between nearest occurrences.
`standardize_preds`	Logical. Should environmental layers be scaled? Default is TRUE.
`...`	Additional parameters passed to internal functions.

Details

Current plug-and-play methods include: "gaussian", "kde","vine","rangebagging", "lobagoc", and "none". Current density ratio methods include: "ulsif", "rulsif".

Value

A list containing 1) a data.frame containing cross-validated model performance statistics (fold_results), and 2) a data.frame containing model performance statistics evaluated on the full dataset (overall_results).

Note

Either `method` or both `presence_method` and `background_method` must be supplied.

Examples

```
{

# load in sample data

 library(S4DM)
 library(terra)

 # occurrence points
   data("sample_points")
   occurrences <- sample_points
```

```
 # environmental data
   env <- rast(system.file('ex/sample_env.tif', package="S4DM"))

 # rescale the environmental data

   env <- scale(env)

# Evaluate a gaussian/gaussian model calculated with the numbag approach
# using 10 bootstrap replicates.

 evaluate_range_map(occurrences = occurrences,
                    env = env,
                    method = NULL,
                    presence_method = "gaussian",
                    background_method = "gaussian",
                    bootstrap = "numbag",
                    bootstrap_reps = 10,
                    quantile = 0.05,
                    constraint_regions = NULL,
                    background_buffer_width = 100000)



}
```

`fit_density_ratio` *Fit density-ratio distribution models in a plug-and-play framework.*

Description

This function fits density-ratio species distribution models for the specified density-ratio method (Drake and Richards 2018; Maitner et al. 2026).

Usage

```
fit_density_ratio(presence = NULL, background = NULL, method = NULL, ...)
```

Arguments

`presence`	dataframe of covariates at presence points
`background`	Dataframe of covariates at background points
`method`	Character. See "notes" for options.
`...`	Additional parameters passed to internal functions.

Details

Current methods include: "ulsif", "rulsif", "kliep", "maxnet"

Value

List of class "dr_model" containing model objects and metadata needed for projecting the fitted models.

References

Drake JM, Richards RL (2018). “Estimating environmental suitability.” *Ecosphere*, **9**(9), e02373. doi:10.1002/ecs2.2373.

Maitner BS, Richards RL, Carlson BS, Drake JM, Merow C (2026). “Flexible methods for species distribution modeling with small samples.” *Ecography*, **2026**(2), e08112. doi:10.1002/ecog.08112.

Examples

```
# load in sample data

 library(S4DM)
 library(terra)

 # occurrence points
   data("sample_points")
   occurrences <- sample_points

 # environmental data
   env <- rast(system.file('ex/sample_env.tif', package="S4DM"))

 # rescale the environmental data

   env <- scale(env)

 # Get presence environmental data

  pres_env <- get_env_pres(coords = occurrences,
                           env = env)

# Get background environmental data

 bg_env <- get_env_bg(coords = occurrences,
                      env = env,width = 100000)


# Note that the functions to get the environmental data return lists,
# and only the "env" element of these is used in the fit function

rulsif_fit <- fit_density_ratio(presence = pres_env$env,
                               background = bg_env$env,
                               method = "rulsif")
```

`fit_plug_and_play` *Fit presence-background distribution models in a plug-and-play framework.*

Description

This function fits presence-background species distribution models for the specified plug-and-play methods (Drake and Richards 2018; Drake 2015; Maitner et al. 2026).

Usage

```
fit_plug_and_play(
  presence = NULL,
  background = NULL,
  method = NULL,
  presence_method = NULL,
  background_method = NULL,
  bootstrap = "none",
  bootstrap_reps = 100,
  ...
)
```

Arguments

`presence`	dataframe of covariates at presence points
`background`	Optional. Dataframe of covariates at background points
`method`	Optional. If supplied, both presence and background density estimation will use this method.
`presence_method`	Optional. Method for estimation of presence density.
`background_method`	Optional. Method for estimation of background density.
`bootstrap`	Character. One of "none" (the default, no bootstrapping), "numbag" (presence function is bootstrapped), or "doublebag" (presence and background functions are bootstrapped).
`bootstrap_reps`	Integer. Number of bootstrap replicates to use (default is 100)
`...`	Additional parameters passed to internal functions.

Details

Current methods include: "gaussian", "kde","vine","rangebagging", "lobagoc", and "none".

Value

List of class "pnp_model" containing model objects and metadata needed for projecting the fitted models.

Note

Either `method` or both `presence_method` and `background_method` must be supplied.

References

Drake JM (2015). "Range bagging: a new method for ecological niche modelling from presence-only data." *J. R. Soc. Interface*, **12**(107). doi:10.1098/rsif.2015.0086.

Drake JM, Richards RL (2018). "Estimating environmental suitability." *Ecosphere*, **9**(9), e02373. doi:10.1002/ecs2.2373.

Maitner BS, Richards RL, Carlson BS, Drake JM, Merow C (2026). "Flexible methods for species distribution modeling with small samples." *Ecography*, **2026**(2), e08112. doi:10.1002/ecog.08112.

Examples

```
# load in sample data

 library(S4DM)
 library(terra)

 # occurrence points
   data("sample_points")
   occurrences <- sample_points

 # environmental data
   env <- rast(system.file('ex/sample_env.tif', package="S4DM"))

 # rescale the environmental data

   env <- scale(env)

 # Get presence environmental data

  pres_env <- get_env_pres(coords = occurrences,
                           env = env)

# Get background environmental data

 bg_env <- get_env_bg(coords = occurrences,
                      env = env,width = 100000)


# Note that the functions to get the environmental data return lists,
# and only the "env" element of these is used in the fit function

  kde_fit <- fit_plug_and_play (presence = pres_env$env,
                                background = bg_env$env,
                                method = "kde")
```

`get_env_bg` *Extract background data for SDM fitting.*

Description

This function extracts background data around known presence records.

Usage

```
get_env_bg(
  coords,
  env,
  method = "buffer",
  width = NULL,
  constraint_regions = NULL,
  standardize = TRUE
)
```

Arguments

`coords`	Coordinates (long,lat) to extract values for
`env`	Environmental SpatRaster(s) in any projection
`method`	Methods for getting bg points. Current option is buffer
`width`	Numeric or NULL. Width (meters or map units) of buffer. If NULL, uses max dist between nearest occurrences.
`constraint_regions`	An optional spatialpolygons* object that can be used to limit the selection of background points.
`standardize`	Logical. If TRUE, the variables will be scaled and centered

Value

A list containing 1) the background data (env), 2) the cell indices for which the background was taken (buffer_cells), 3) the environmental means (env_mean; NA if standardization not done), and 4) the environmental standard deviations (env_sds; NA if standardization not done).

Note

If supplying constraint_regions, any polygons in which the occurrences fall are considered fair game for background selection. This background selection is, however, still limited by the buffer as well.

Examples

```
{

# load in sample data

 library(S4DM)
 library(terra)

 # occurrence points
   data("sample_points")
   occurrences <- sample_points

 # environmental data
   env <- rast(system.file('ex/sample_env.tif', package="S4DM"))

 # rescale the environmental data

   env <- scale(env)

bg_data <- get_env_bg(coords = occurrences,
                      env = env,
                      method = "buffer",
                      width = 100000)


}
```

`get_env_pres` *Extract presence data for SDM fitting.*

Description

This function extracts presence data at known presence records.

Usage

```
get_env_pres(coords, env, env_bg = NULL)
```

Arguments

`coords`	Coordinates (long,lat) to extract values for
`env`	Environmental SpatRaster(s) in any projection
`env_bg`	Background data produced by `get_env_bg`, used for re-scaling

Value

A list containing 1) the environmental data at the presence locations (env), and 2) an sf data.frame containing the occurrence records(occurrence_sf).

Examples

```
 {

# load in sample data

 library(S4DM)
 library(terra)

 # occurrence points
   data("sample_points")
   occurrences <- sample_points

 # environmental data
   env <- rast(system.file('ex/sample_env.tif', package="S4DM"))

 # rescale the environmental data

   env <- scale(env)

env_pres <- get_env_pres(coords = occurrences,
                        env = env)

}
```

`get_response_curves` *Generate Response Curves*

Description

Given an environmental data set, fitted models, and a directory to output plots, this function generates response curves for each predictor in the model. The response curves depict the predicted change in probability of presence as a function of the environmental predictor while holding all other predictors constant at their mean values.

Usage

```
get_response_curves(
  env_bg,
  env_pres,
  pnp_model,
  n.int = 1000,
  envMeans = NULL,
  envSDs = NULL
)
```

Arguments

`env_bg`	Object returned by get_env_bg

`env_pres`	Object returned by get_env_pres
`pnp_model`	Object returned by `fit_plug_and_play` or `fit_density_ratio`
`n.int`	Number of points along which to calculate the response curve
`envMeans`	A vector of means for each environmental predictor in the dataset. (not used)
`envSDs`	A vector of standard deviations for each environmental predictor in the dataset.(not used)

Value

This function generates a set of marginal predictions for each environmental variable, holding other variables constant

Author(s)

Cory Merow, modified by Brian Maitner

`make_range_map` *Make a range map using plug-and-play modeling.*

Description

This function produces range maps using plug-and-play modeling with either presence-background or density-ratio approaches.

Usage

```
make_range_map(
  occurrences,
  env,
  method = NULL,
  presence_method = NULL,
  background_method = NULL,
  bootstrap = "none",
  bootstrap_reps = 100,
  quantile = 0.05,
  background_buffer_width = NULL,
  constraint_regions = NULL,
  verbose = FALSE,
  standardize_preds = TRUE,
  ...
)
```

Arguments

`occurrences`	Presence coordinates in long,lat format.
`env`	Environmental rasters
`method`	Optional. If supplied, both presence and background density estimation will use this method.
`presence_method`	Optional. Method for estimation of presence density.
`background_method`	Optional. Method for estimation of background density.
`bootstrap`	Character. One of "none" (the default, no bootstrapping), "numbag" (presence function is bootstrapped), or "doublebag" (presence and background functions are bootstrapped).
`bootstrap_reps`	Integer. Number of bootstrap replicates to use (default is 100)
`quantile`	Quantile to use for thresholding. Default is 0.05 (5 pct training presence). Set to 0 for minimum training presence (MTP), set to NULL to return continuous raster.
`background_buffer_width`	The width (in m for unprojected rasters and map units for projected rasters) of the buffer to use for background data. Defaults to NULL, which will take the maximum distance between occurrence records.
`constraint_regions`	See get_env_bg documentation
`verbose`	Logical. If TRUE, prints progress messages.
`standardize_preds`	Logical. Should environmental layers be scaled? Default is TRUE.
`...`	Additional parameters passed to internal functions.

Details

Current plug-and-play methods include: "gaussian", "kde","vine","rangebagging", "lobagoc", and "none". Current density ratio methods include: "ulsif", "rulsif",and "maxnet".

Value

A SpatRaster object containing a range map. Maps may be either binary or continuous, depending upon the `quantile` argument.

Note

Either `method` or both `presence_method` and `background_method` must be supplied.

Examples

```
{

# load in sample data
```

```
 library(S4DM)
 library(terra)

 # occurrence points
   data("sample_points")
   occurrences <- sample_points

 # environmental data
   env <- rast(system.file('ex/sample_env.tif', package="S4DM"))

 # rescale the environmental data

   env <- scale(env)

   map <- make_range_map(occurrences = occurrences,
                         env = env,
                         method = "gaussian",
                         presence_method = NULL,
                         background_method = NULL,
                         bootstrap = "none",
                         bootstrap_reps = 100,
                         quantile = 0.05,
                         background_buffer_width = 100000)

   plot(map)


}
```

`project_density_ratio` *Projects fitted density-ratio distribution models onto new covariates.*

Description

This function projects fitted density-ratio species distribution models onto new covariates.

Usage

```
project_density_ratio(dr_model, data)
```

Arguments

`dr_model`	A fitted density ratio model produced by `fit_density_ratio`
`data`	covariate data

Value

A vector of relative occurrence rates evaluated at the covariates supplied in the data object.

`project_plug_and_play` *Projects fitted plug-and-play distribution models onto new covariates.*

Description

This function projects fitted plug-and-play species distribution models onto new covariates.

Usage

```
project_plug_and_play(pnp_model, data, bg_zero_ror_zero = TRUE)
```

Arguments

`pnp_model`	A fitted plug-and-play model produced by `fit_plug_and_play`
`data`	covariate data
`bg_zero_ror_zero`	Logical. Default = TRUE. If the background is zero, set the ROR to zero. This will prevent NaN and Inf values which would otherwise interfere with AUC calculations.

Value

A vector of relative occurrence rates evaluated at the covariates supplied in the data object.

Note

The tsearchn function underlying rangebagging seems to fail sometimes with very uneven predictors. Rescaling helps.

`sample_points` *Example S4DM occurrence data*

Description

A sample dataset containing occurrence records.

Usage

```
sample_points
```

Format

A data.frame with 65 observations of 2 variables:

Longitude Longitude, in decimal degrees

Latitude Latitude, in decimal degrees ...

Source

https://biendata.org

`sdm_threshold` *Thresholds a continuous relative occurrence rate raster to create a binary raster.*

Description

This function thresholds a continuous relative occurrence rate raster to produce a binary presence/absence raster.

Usage

```
sdm_threshold(
  prediction_raster,
  occurrence_sf,
  quantile = 0.05,
  return_binary = TRUE
)
```

Arguments

`prediction_raster`	Raster containing continuous predictions of relative occurrence rate to be thresholded.
`occurrence_sf`	An sf object containing presence locations. Should be in the projection of the prediction raster
`quantile`	Numeric between 0 and 1. Quantile to use for thresholding (defaults to 0.05). Set to 0 for minimum training presence.
`return_binary`	LOGICAL. Should the raster returned be binary (presence/absence)? If FALSE, predicted presences will retain their 'suitability" scores.

Value

A SpatRaster object containing a range map. Maps may be either binary or continuous, depending upon the `return_binary` argument.

Author(s)

Cecina Babich Morrow (modified by Brian Maitner)

Examples

```
{

# load in sample data

library(S4DM)
library(terra)

# occurrence points
  data("sample_points")
  occurrences <- sample_points

# environmental data
  env <- rast(system.file('ex/sample_env.tif', package="S4DM"))

# rescale the environmental data

  env <- scale(env)

 bg_data <- get_env_bg(coords = occurrences,
                       env = env,
                       method = "buffer",
                       width = 100000)

 pres_data <- get_env_pres(coords = occurrences,
                           env = env)

 pnp_model <-fit_plug_and_play(presence = pres_data$env,
                   background = bg_data$env,
                   method = "gaussian")

 pnp_continuous <- project_plug_and_play(pnp_model = pnp_model,
                                         data = bg_data$env)

 #Make an empty raster to populate
 out_raster <- env[[1]]
 values(out_raster) <- NA

 # use the bg_data for indexing
 out_raster[bg_data$bg_cells] <- pnp_continuous

 plot(out_raster)

 #convert to a binary raster

 out_raster_binary <-
   sdm_threshold(prediction_raster = out_raster,
               occurrence_sf = pres_data$occurrence_sf,
               quantile = 0.05,
               return_binary = TRUE)

 plot(out_raster_binary)
```

```
}
```

`stratify_random` *Split data for k-fold spatially stratified cross validation*

Description

Splitting tool for cross-validation

Usage

```
stratify_random(occurrence_sf, nfolds = NULL)
```

Arguments

`occurrence_sf`	a sf object containing occurrence records
`nfolds`	number of desired output folds.

Details

See Examples.

Value

Returns a sf dataframe containing fold designation for each point.

Author(s)

Cory Merow cory.merow@gmail.com

Examples

```
{

# load in sample data

 library(S4DM)
 library(terra)
 library(sf)

 # occurrence points
   data("sample_points")
   occurrences <- sample_points


 occurrences <- st_as_sf(x = occurrences,coords = c(1,2))
```

```
random_folds <- stratify_random(occurrence_sf = occurrences,
                               nfolds = 5)


}
```

`stratify_spatial` *Split data for k-fold spatially stratified cross validation*

Description

Splitting tool for cross-validation

Usage

```
stratify_spatial(occurrence_sf, nfolds = NULL, nsubclusters = NULL)
```

Arguments

`occurrence_sf`	a sf object containing occurrence points
`nfolds`	number of desired output folds. Default value of NULL makes a reasonable guess based on sample size.
`nsubclusters`	intermediate number of clusters randomly split into nfolds. Default value of NULL makes a reasonable guess based on sample size. If you specify this manually, it should be an integer multiple of nfolds.

Details

See Examples.

Value

Returns a SpatialPoints dataframe with the data.frame containing fold designation for each point.

Author(s)

Cory Merow cory.merow@gmail.com

Examples

```
{

# load in sample data

 library(S4DM)
 library(terra)
 library(sf)
```

```
 # occurrence points
   data("sample_points")
   occurrences <- sample_points


 occurrences <- st_as_sf(x = occurrences,coords = c(1,2))

manual <- stratify_spatial(occurrence_sf = occurrences,nfolds = 5,nsubclusters = 5)
default <- stratify_spatial(occurrence_sf = occurrences)


}
```

Index