Package 'IceCast'

June 24, 2019

Type Package

Title Apply Statistical Post-Processing to Improve Sea Ice Predictions

Version 2.1.0

Author Hannah M. Director, Adrian E. Raftery, and Cecilia M. Bitz

Maintainer Hannah M. Director <direch@uw.edu>

Description Tools for correcting biases and calibrating sea ice predictions obtained from dynamic ensemble models. Implements and extends Director et al. (2017) <doi:10.1175/JCLI-D-17-0185.1> This package depends on the 'ncdf4' and 'rgeos' R packages. These packages require installing externally from R Unidata's 'NetCDF' library and Geometry Engine - Open Source ('GEOS'). (See the 'rgeos' and 'ncdf4' packages for details.)

License GPL (>= 2)

Depends R (>= 2.10), rgeos, maptools, sp

Imports ncdf4, MASS, raster, methods, utils, Rcpp (>= 0.12.17), coda

Suggests geosphere, fields, knitr, rmarkdown, viridis

VignetteBuilder knitr

Encoding UTF-8

LazyData true

RoxygenNote 6.1.1

LinkingTo Rcpp, RcppArmadillo

NeedsCompilation yes

Repository CRAN

Date/Publication 2019-06-24 07:40:03 UTC

R topics documented:

`all_regions` *Polygon of all regions*

Description

All regions except the non-regional ocean converted into a single `SpatialPolygons` object. Regions are modified from the region masks provided by the National Snow and Ice Data Center (NSIDC)

Usage

```
all_regions
```

Format

`SpatialPolygons` object

References

Region Mask: National Snow and Ice Data Center, 2017: Region mask for the northern hemisphere. `http://nsidc.org/data/polar-stereo/tools_masks.html`.

Examples

```
data(all_regions)
plot(all_regions)
```

any_intersect *Check if a line has intersecting segments*

Description

Determine if there are any intersecting line segments in a matrix of coordinates representing a line

Usage

```
any_intersect(line)
```

Arguments

`line`	matrix of coordinates corresponding to the line of interest

Value

list where `list$any` is a boolean indicating if there are any intersections and `list$val` is an index corresponding to the first intersection found

Examples

```
check_results <- any_intersect(currSecEx)
check_results$any #true/false
check_results$val #indices of first intersection found
```

bg_water *Polygon of the non-regional ocean*

Description

The non-regional ocean converted into a single `SpatialPolygons` object. The boundaries of the non-regional ocean were defined by modifying the region masks provided by the National Snow and Ice Data Center (NSIDC).

Usage

```
bg_water
```

Format

`SpatialPolygons` object

References

Region Mask: National Snow and Ice Data Center, 2017: Region mask for the northern hemisphere. http://nsidc.org/data/polar-stereo/tools_masks.html.

Examples

```
data(bg_water)
plot(bg_water)
```

bound_info *Get boundary info*

Description

Determine which y values are on the boundaries and what the corresponding bounds of those y values are

Usage

```
bound_info(y_obs, dist, loop)
```

Arguments

y_obs	matrix of observed distances (dimension: number of lines by number of years)
dist	a list of the lengths for the corresponding lines
loop	boolean which if true TRUE indicates that the lines extend outward from a single point forming a circle and if FALSE indicates that the lines are mapped along a fixed contour such as a land boundary

Value

list of 3 matrices each of dimension number of lines by number of years giving the lower bound for hidden x values, the upper bound for hidden x values, and an indicator of whether the value is bounded at all. The values in the list are named ub, lb, and xUnObs respectively.

calc_pars *Compute Parameters Estimates*

Description

Compute parameter estimates for the contour model using MCMC output from `fit_cont_pars`

Usage

```
calc_pars(res_r, burn_in, w)
```

Arguments

`res_r`	output of MCMC run from function `fit_cont_pars` for one region
`burn_in`	number of iterations to discard as burn-in. This is the number before thinning. Value will be divided by `w`.
`w`	integer specifying the thinning used. Samples from every w-th iteration are stored.

Value

List of a list of parameters for each region. Each list contains two elements, `muEst` and `sigmaEst`. These which give estimates for the `mu` and `sigma` parameters used to generate contours.

Examples

```
## Not run:
y_obs <- y_obs(maps = obs_maps, reg_info)
res <- fit_cont_pars(r = 3, n_iter = 1000, y_obs, reg_info)
calc_pars(res, burn_in = 100, w = res$w)

## End(Not run)
```

censor *Truncate simulated line lengths based on a list of bounds*

Description

Take in unbounded line lengths (x-values) and truncate them based on provided bounds to create bounded lengths (y-values)

Usage

```
censor(x, bounds)
```

Arguments

x a vector of generated line lengths

bounds a vector which gives the lengths of the end and breakpoints for each x value

Value

vector of new line lengths

check_intersect *Check if line segments intersect*

Description

Find if two line segments intersect

Usage

```
check_intersect(a, b, c, d, seq = FALSE)
```

Arguments

a first coordinate of first line segment

b second coordinate of first line segment

c first coordinate of second line segment

d second coordinate of second line segment

seq indicator for whether the two line segments are intersecting

Value

boolean indicating if there is an intersection

Examples

```
check_intersect(c(0, 0), c(1, 1), c(2, 2), c(3, 3))
check_intersect(c(0, 0), c(1, 1), c(0.5, 0.5), c(2, 2))
```

`clim_9_2005_2007` *Climatology forecast for 2005-2007.*

Description

Proportion of times in the preceding ten years that sea ice concentration of at least 15% was observed in each grid box. Array of dimension year by longitude by latitude. Computed from NASA Boostrap sea ice concentration product.

Usage

```
data(clim_9_2005_2007)
```

Format

array

References

Comiso, J., 2017: Bootstrap sea ice concentrations from Nimbus-7 SMMR and DMSP SSM/I-SSMIS. version 3. Boulder, Colorado USA: NASA National Snow and Ice Data Center Distributed Active Archive Center. doi: https://doi.org/10.5067/7Q8HCCWS4I0R

`clim_9_2008` *Climatology forecast for 2008.*

Description

Proportion of times in the preceding ten years that sea ice concentration of at least 15% was observed in each grid box. Array of dimension year by longitude by latitude. Computed from NASA Boostrap sea ice concentration product.

Usage

```
data(clim_9_2008)
```

Format

array

References

Comiso, J., 2017: Bootstrap sea ice concentrations from Nimbus-7 SMMR and DMSP SSM/I-SSMIS. version 3. Boulder, Colorado USA: NASA National Snow and Ice Data Center Distributed Active Archive Center. doi: https://doi.org/10.5067/7Q8HCCWS4I0R

`cond_prob` *Compute conditional probability of observed event*

Description

Computes conditional probability of the observed event under some probability model

Usage

```
cond_prob(obs, mod)
```

Arguments

`obs` matrix with binary observations indicating if sea ice concentration of at least 15$%$ was observed. Dimension is lon x lat.

`mod` matrix with estimated sea ice probability from a model. Dimension is lon x lat.

`contig_zero` *Find indices of sequences of contiguous zeros*

Description

Identify the indices of sequences of repeated zero values and indices of sequences of non-zero values

Usage

```
contig_zero(y)
```

Arguments

`y` vector to consider

Value

Matrix of three columns where each row gives the first and last index of a sequence of numbers. The third column is a boolean. If `TRUE`, the indices are for a sequence zeros. If `FALSE`, the indices are for a sequence non-zero values.

contour_shift *Apply contour-shifting to bias correct*

Description

Apply contour-shifting to bias correct a predicted contour using existing mappings.

Usage

```
contour_shift(maps, predicted, bc_year, pred_start_year, reg_info, level,
  dat_type_pred, my_land_mat = land_mat, my_land = land,
  n_train_years = NULL)
```

Arguments

maps	object obtained from the create_mapping function (see details)
predicted	array of predicted values of dimension year x month x longitude x latitude
bc_year	year to be bias-corrected
pred_start_year	year prediction array starts in
reg_info	a reg_info list (see documentation for reg_info)
level	concentration level for which to build contour
dat_type_pred	string indicating the format of the prediction: either "gfdl" or "simple" (see details)
my_land_mat	binary matrix specifying land locations
my_land	SpatialPolygons corresponding to the land
n_train_years	number of years prior to the current year used in fitting the bias correction

Details

The map parameter is a list of length four that has the form of a list obtained from running the create_mapping function. The values start_year and endYear give the first and last year that were mapped. The variables obs_list and pred_list are lists of arrays with one 3-dimensional array for each region. The first dimension of each array coresponds to the year, the second dimension corresponds to the lines on which the region is being mapped, and the third dimension corresponds to the variables of interest. The first and second dimension are indexed sequentially. The variables for the third dimension are for the fixed points' x-coordinates, the fixed points' y-coordinates, the mapped points' x-coordinates, the mapped points' y-coordinates, the length of the mapping vectorsin the x-direction, the length of the vectors in the y-direction, and the angles of the mapping vectors. The predicted data array, predicted, should be a single array of dimension: years x longitude (304) x latitude (448). If dat_type_pred = ``simple"}, the values in the array should indicate whether each grid box is categorized to contain ice (1: ice-covered, 0: no ice, NA: la
\code{dat_type_pred =``gfdl" the values in the predicted array correspond to the raw ice concentrations values predicted (including indicators for missing data, land etc.) formatted as in the

CM2.5 Forecast-oriented Low-Ocean Resolution (FLOR) model produced by the National Oceanic and Atmospheric Administration's Geophysical Fluid Dynamics Laboratory and converted to a Polar Stereographic grid (Vecchi et al. 2014; Msadek et al. 2014). Weights for converting to a polar stereograhic grid were obtained from the spherical coordinate remapping and interpolation package (SCRIP) (Jones 1997).

Value

SpatialPolygons object of the adjusted region

References

Jones, P.W. "A user's guide for SCRIP: A spherical coordinate remapping and interpolation package." Los Alamos National Laboratory, Los Alamos, NM (1997).

Msadek, R., et al. "Importance of initial conditions in seasonal predictions of Arctic sea ice extent." Geophysical Research Letters 41.14 (2014): 5208-5215.

Vecchi, Gabriel A., et al. "On the seasonal forecasting of regional tropical cyclone activity." Journal of Climate 27.21 (2014): 7994-8016.

Examples

```
## Not run:
adj <- contour_shift(maps = discrep, predicted = emFeb2012, bc_year = 2012,
                     pred_start_year = 1980, reg_info, level = 15,
                     dat_type_pred = "gfdl")
plot(land, col = "grey", border = FALSE)
plot(adj, add = TRUE, col = "blue")

## End(Not run)
```

conv_to_grid *Convert* SpatialPolygons *object to a grid*

Description

Convert SpatialPolygons object to a binary grid. Grid boxes whose centers are part ofthe SpatialPolygons are given value 1 and all other grid boxes are given value 0. Land values are set to NA.

Usage

```
conv_to_grid(x, my_land_mat = land_mat)
```

Arguments

x	SpatialPolygons object
my_land_mat	binary matrix specifying land locations

`create_mapping` *Map a set of observations and predictions*

Description

Finds all the mappings for a set of observations and predictions often over multiple years

Usage

```
create_mapping(start_year, end_year, obs_start_year, pred_start_year,
  observed, predicted, reg_info, month, level, dat_type_obs, dat_type_pred,
  plotting = FALSE, obs_only = FALSE, pred_only = FALSE, nX = 304,
  nY = 448, xmn = -3850, xmx = 3750, ymn = -5350, ymx = 5850)
```

Arguments

`start_year`	first year to be mapped
`end_year`	last year to be mapped
`obs_start_year`	year in which observation array starts
`pred_start_year`	year in which prediction array starts
`observed`	array of observed values of dimension year x longitude x latitude
`predicted`	array of predicted values of dimension year x longitude x latitude
`reg_info`	a `reg_info` list (see documentation for `reg_info`)
`month`	month under consideration
`level`	concentration level for which to build contour
`dat_type_obs`	string of either "bootstrap" or "simple" indicating the file type of the observation (see details)
`dat_type_pred`	string of either "gfdl" or "simple" indicating the file type of the prediction (see details)
`plotting`	boolean indicatng whether maps should be plotted (defaults to false)
`obs_only`	indicator to run mapping only for observations
`pred_only`	indicator to run mapping only for predictions
`nX`	dimension in the x (defaults to value for Northern Polar stereographic grid: 304)
`nY`	dimension in the y (defaults to value for Northern Polar stereographic grid: 448)
`xmn`	min x value (defaults to value for Northern Polar stereographic grid: -3850)
`xmx`	max x value (defaults to value for Northern Polar stereographic grid: 3750)
`ymn`	min y value (defaults to value for Northern Polar stereographic grid: -5350)
`ymx`	max y value (defaults to value for Northern Polar stereographic grid: 5850)

Details

The object `maps` is obtained from running the `create_mapping` function. It is a list of four objects. The first two items in the list, `start_year` and `end_year`, give the first and last year that were mapped. The second two items, `obs_list` and `pred_list`, are lists of arrays with one 3-dimensional array for each region. The first dimension is for the year. The other two dimensions are for the fixed points' y-coordinates, the mapped points' x-coordinates, the mapped points' y-coordinates, the length of the mapping vectors in the x-direction, the length of the vectors in the y-direction, and the angles of the mapping vectors.

For `dat_type_obs = "simple"` and `dat_type_pred = "simple"` the values in the `observed` and `predicted` arrays are indicators of whether the grid box contains ice (1: ice-covered, 0: no ice, NA: land). If `datTypePred = "gfdl"` or `dat_type_obs = "bootstrap"`, the values in the `observed` and `predicted` arrays correspond to the raw ice concentrations values observed or predicted (including indicators for missing data, land etc.). If `datTypePred = "gfdl"`, the predictions are formatted as in the CM2.5 Forecast-oriented Low-Ocean Resolution (FLOR) model produced by the National Oceanic and Atmospheric Administration's Geophysical Fluid Dynamics Laboratory and converted to a Polar Stereographic grid (Vecchi et al. 2014; Msadek et al. 2014). If `datTypeObs = "bootstrap"` the array values are assumed to be from the monthly sea ice concentration obtained from the National Aeronautics and Space Administration (NASA) satellites Nimbus-7 SMMR and DMSP SSM/I-SSMIS and processed by the bootstrap algorithm. Weights for converting to a polar stereograhic grid were obtained from the spherical coordinate remapping and interpolation package (SCRIP) (Jones 1997).

Value

`map` object (see details)

References

Comiso, J., 2017: Bootstrap sea ice concentrations from Nimbus-7 SMMR and DMSP SSM/I-SSMIS. version 3. Boulder, Colorado USA: NASA National Snow and Ice Data Center Distributed Active Archive Center. doi: https://doi.org/10.5067/7Q8HCCWS4I0R

CM2.5 Forecast-oriented Low-Ocean Resolution (FLOR) model: Vecchi, Gabriel A., et al. "On the seasonal forecasting of regional tropical cyclone activity." Journal of Climate 27.21 (2014): 7994-8016.

Msadek, R., et al. "Importance of initial conditions in seasonal predictions of Arctic sea ice extent." Geophysical Research Letters 41.14 (2014): 5208-5215.

National Center for Atmospheric Research, 2017: Earth system grid at NCAR. https://www.earthsystemgrid.org/home.html.

Jones, P.W. "A user's guide for SCRIP: A spherical coordinate remapping and interpolation package." Los Alamos National Laboratory, Los Alamos, NM (1997).

Examples

```
## Not run:
create_mapping(start_year = 1981, end_year = 1981, obs_start_year = 1981,
               pred_start_year = 1980, observed = obsFeb19811982,
               predicted = emFeb19811982, reg_info = reg_info, month = 2,
               level = 15, dat_type_obs = "bootstrap", dat_type_pred = "gfdl",
               plotting = TRUE)
## End(Not run)
```

---

currSecEx *Coordinates of a line segment with self-intersections*

Description

Example of a line segment with self-intersections. We will use it to demonstrate the `untwistSec` function.

Usage

```
currSecEx
```

Format

n x 2 matrix of coordiantes

Examples

```
data(currSecEx)
head(currSecEx)
```

---

discrep *Discrepancy maps for September 1993-2007 (lead time 2.5-months)*

Description

The object `discrep` is obtained from running the `createMapping` function for September 1993-2007. The predictions used are from European Center for Medium-Range Weather Forecasts (ECMWF) at a 2.5-month lead time and are converted to a Polar Stereographic grid. Model output is available from the Sea Ice Prediction Network Predicatability Portal or the Copernicus Climate Change Service data store. The observations used are from the monthly sea ice concentration obtained fromthe National Aeronautics and Space Administration (NASA) satellites Nimbus-7 SMMR and DMSP SSM/I-SSMIS and processed by the bootstrap algorithm. The results are distributed by the National Snow and Ice Data Center (Comiso 2017).

Usage

```
discrep
```

Format

Object obtained from the `createMapping` function (see details)

Details

The object `discrep` is obtained from running the `createMapping` function. It is a list of four objects where `startYear` and `endYear` give the first year and last year that were mapped. The variables `obsList` and `predList` are lists of arrays with one 3-dimensional array for each region. The first dimension is for the year. The other two dimensions are for the fixed points' y-coordinates, the mapped points' x-coordinates, the mapped points' y-coordinates, the length of the mapping vectors in the x-direction, the length of the vectors in the y-direction, and the angles of the mapping vectors.

References

Comiso, J., 2017: Bootstrap sea ice concentrations from Nimbus-7 SMMR and DMSP SSM/I-SSMIS. version 3. Boulder, Colorado USA: NASA National Snow and Ice Data Center Distributed Active Archive Center. doi: https://doi.org/10.5067/7Q8HCCWS4I0R

Copernicus Climate Change Service (2019). Description of the c3s seasonal multi-system.https://confluence.ecmwf.int/display/COPSRV/Description+of+the+C3S+seasonal+multi-system

Sea Ice Prediction Network (2019). Sea ice prediction network predictability portal. https://atmos.uw.edu/sipn/.

Examples

```
data(discrep)
names(discrep)
```

`dist_mat` *Compute 'distances' among $n$ lines*

Description

Creates a matrix specifying how difference indices are among an ordered set of indices. For lines with indices i and j, the 'distance' computed is $|i - j|$. Results are used to define covariance.

Usage

```
dist_mat(n_lines)
```

Arguments

`n_lines`	number of lines in matrix

Value

Matrix of 'distances' among the indices

ecmwf_bin *Binary predictions from ECMWF ensemble, September 1993-2018*

Description

The object ensemble_bin is a binary array indicating if at least half of the ensemble members have sea ice concentrations of at least 15% from September 1993-2018. The predictions are from the European Center for Medium-Range Weather Forecasts (ECMWF) ensemble at a 2.5-month lead time. They have been converted to a Polar stereographic grid.

Usage

```
ecmwf_bin
```

Format

array of dimension of 26 x 304 x 448 (corresponding to year x longitude x 448 latitude)

References

Copernicus Climate Change Service (2019). Description of the c3s seasonal multi-system. https://confluence.ecmwf.int/display/COPSRV/Description+of+the+C3S+seasonal+multi-system

Sea Ice Prediction Network (2019). Sea ice prediction network predictability portal. https://atmos.uw.edu/sipn/.

Examples

```
data(ecmwf_bin)
```

extract_coords *Function to extract coordinates.*

Description

Function to extract coordinates from a SpatialLines object. If there are breaks in the line, this function connects the closest points to create one line. Note: This differs from the function getCoords in that the ordering of the points is considered.

Usage

```
extract_coords(x)
```

Arguments

x	SpatialLines or SpatialPolygons object

Value

n x 2 matrix of coordinates

Examples

```
coords <- extract_coords(reg_info$regions[[3]])
par(mfrow = c(1, 2))
plot(reg_info$regions[[3]], main = "Polygon Object")
plot(coords, type = "p", main = "Coordinates", pch = 20)
```

`find_trans` *Find transition points (points at the start/end of polygons)*

Description

Find transition points (points at the start/end of polygons)

Usage

```
find_trans(pts, r, reg_info, close = 12.5)
```

Arguments

`pts`	matrix with two columns giving coordinates of the points
`r`	integer specifying the region
`reg_info`	a `reg_info` list (see documentation for `reg_info`)
`close`	how close a point must be to the line to count as being on it, defaults to 12.5

`fit_cont_pars` *Sets up MCMC to fit the parameters of the contour Model in R, then runs the sampler in C++*

Description

Sets up MCMC to fit the parameters of the contour Model in R, then runs the sampler in C++

Usage

```
fit_cont_pars(r, n_iter, y_obs, reg_info, dists = NULL,
  sigma_min = 0.01, sigma0_lb = NULL, sigma0_ub = NULL,
  xU_prop_sd_def = 0.03, mu_ini = NULL, mu0 = NULL, lambda0 = NULL,
  sigma_ini = NULL, sigma_prop_cov = NULL, sigma_sp = 25,
  rho_ini = 0.5, rho0_lb = 0, rho0_ub = 0.99, rho_prop_sd = 0.01,
  w = 20)
```

Arguments

`r` number indicating which region in the `reg_info` list is being considered

`n_iter` number of iterations to run the MCMC, must be a multiple of `w`

`y_obs` output of `y_obs` function. This is a list of matrices, one per region, giving the observed y values. Each row corresponds to the lines and each column corresponds to a training year

`reg_info` a `reg_info` list (see documentation for `reg_info`)

`dists` symmetric matrix of the same dimension as the number of lines being used, specifying distances among indices. Defaults to `NULL`, which means atrix will be computed by the `dist_mat` function

`sigma_min` minimum value for all σ parameters. Typically close to but not exactly zero (defaults to 0.01). Not used if `sigma0_lb` is set to `NULL`

`sigma0_lb` vector of the same length as the number of lines which specifies the lower bound of the uniform prior for each sigma value. Defaults to `NULL`, meaning `sigma0_lb` is set to be a vector with all values set to `sigmaMin`

`sigma0_ub` vector of the same length as the number of lines which specifies the upper bound of the uniform prior for each sigma value. Defaults to `NULL`

`xU_prop_sd_def` Standard deviation for proposals for xU when xU can take on an infinite set of values

`mu_ini` vector of the same length as the number of lines which specifies the values from which each element of μ will be initialized in the MCMC. Defaults to `NULL`, meaning μ will be initialized with the mean of the observed y's

`mu0` vector of the same length as the number of lines which specifies the prior mean for μ. Defaults to `NULL`, meaning each element in `mu0` will be set to be in the middle of its corresponding line

`lambda0` matrix of the same dimension as the number of lines which specifices the prior covariance matrix for μ. Defaults to `NULL`, which gives a diagonal matrix with diagonal elements corresponding to the variance that would be required for 80 values of the corresponding line if the data were normally distributed.

`sigma_ini` vector of the same length as the number of lines which specifies the values from which each element in Σ will be initialized from. Defaults to `NULL`, meaning each element of Σ will be initialized with the observed standard deviation of its corresponding y's, bounded by `sigma0_lb` and `sigma0_ub`.

`sigma_prop_cov` covariance matrix of the same length as the number of lines that is used in sampling Σ values. Defaults to `NULL`, meaning a diagonal matrix is used. The elements on the diagonal of this matrix are generally set to have value `sigma_ini`/20 unless the corresponding observed y's have zero variance, in which case these values are set to 0.1.

`sigma_sp` integer specifying how many elements in the Σ matrix should be sampled together in the MCMC. Defaults to 25.

`rho_ini` double between 0 and 1 from which the value of `rho` will be initialized. Defaults to 0.5

`rho0_lb` double between 0 and 1 which gives the lower bound of the uniform prior for `rho`. Detauls to 0.

`rho0_ub` double between 0 and 1 which gives the upper bound of the uniform prior for `rho`. Defaults to 1.

`rho_prop_sd` standard deviation for the normal proposal distribution used when proposing value for `rho` in the sampler. Defaults to 0.01

`w` integer specifying how many samples of the parameters will be maintained. Samples from every w-th iteration is stored.

Value

List that gives the values of the MCMC chain for `xU`, `mu`, `sigma` and `rho` along with indicators of acceptance on each iteration: `xURate`, `sigmaRate`, and `rhoRate`. Background information is also outputted including the upper and lower bounds for unobserved x's (`xU_lb`, `xU_ub`), vectors giving the first and last indices of each grouping in sampling Σ (`sigma_ind_1`,`sigma_ind_2`), the distance matrix (`dists`), and the integer specifying how many samples of the parameters will be maintained `w`

Examples

```
## Not run:
y_obs <- y_obs(maps = obs_maps, reg_info)
res <- fit_cont_pars(r = 3, n_iter = 1000, y_obs, reg_info)

## End(Not run)
```

`fit_weights` *Compute weighting between two models*

Description

Compute weighting between two models based on accuracy in predicting a set of observations. Computation is via the Expectation-Maximization algorithm.

Usage

```
fit_weights(mod1, mod2, obs, prop_area, w_ini = 0.5, z_ini = 0.5,
  eps = 0.01)
```

Arguments

`mod1` array with estimated sea ice probability from model 1. Dimensions are nuumber of training years x lon x lat.

`mod2` array with estimated sea ice probability from model 2. Dimensions are nuumber of training years x lon x lat.

`obs` array with observations of sea ice presence (1) and absence (0). Dimensions are nuumber of training years x lon x lat.

`prop_area` matrix that gives the proportion of area in each grid box. Should sum to 1. Dimensions are lon x lat.

w_ini initial value of all w, defaults to 0.5.

z_ini initial value of all z, defaults to 0.5.

eps tolerance for EM algorithm to reach convergence, defaults to 0.01.

Value

value between 0 and 1 giving the weight on the first model

Examples

```
## Not run:
weight <- fit_weights(mod1 = clim_9_2005_2007, mod2 = ppe_9_2005_2007,
obs = obs_9_2005_2007, prop_area = prop_area)

## End(Not run)
```

full *Sample collection of completely ice-filled regions*

Description

Example of a collection of several completely ice-filled regions stored as a single SpatialPolygons object

Usage

```
data(full)
```

Format

An object of class SpatialPolygons of length 1.

Examples

```
data(full)
```

gen_cont *Generate contours*

Description

Generate the contours for a particular region given the model prediction

Usage

```
gen_cont(r, pars_r, reg_info, n_gen = NULL, map_pred_r = NULL,
  stat_only = FALSE, mean_only = FALSE, eff_zero = 12.5,
  stat_only_trend = TRUE)
```

Arguments

`r`	integer indicating the number of the region in which the contours should be generated
`pars_r`	List of parameter information for region r. The list should contain two elements, `muEst` and `sigmaEst`, which give estimates for the μ and Σ parameters used in generating contours. Typically obtained from the `calc_pars` function
`reg_info`	a `reg_info` list (see documentation for `reg_info`)
`n_gen`	integer specifying the number of contours to be generated, must be at least 2
`map_pred_r`	output of `get_map` function applied to `SpatialPolygons` object corresponding to an intial forecast (typically a bias-corrected dynamic ensemble forecast)
`stat_only`	boolean indicating that forecast is purely statistical (no dynamic ensemble model forecast considered)
`mean_only`	boolean indicating that only the mean contour will be computed rather than distribution
`eff_zero`	how close a generated vector needs to be to zero to be counted as a zero, defaults to 12.5
`stat_only_trend`	boolean indicating if a trend adjustment should be applied when `stat_only = TRUE`. Defaults to true

Examples

```
## Not run:
#statistical binary, region 1
stat_bin_1 <- gen_cont(r = 1, pars_r = pars_1, reg_info,
                       stat_only = TRUE, mean_only = TRUE)

#statistical probabilistic, region 1, 2 generated contours
stat_prob_1 <- gen_cont(r = 1, pars_r = pars_1, reg_info,
                        n_gen = 2, stat_only = TRUE)

 #hybrid probabilistic, region 1, 2 generated contours
 hybrid_prob_1 <- gen_cont(r = 1, pars_r = pars_1, reg_info,
                          n_gen = 2, map_pred_r = map_curr_1)

## End(Not run)
```

get_area *Calculate geodesic area*

Description

Caclulate the geodesic areas of SpatialPolygons object on the Nothern Hemisphere Polar Stereographic grid projection

Usage

```
get_area(poly, byid = FALSE)
```

Arguments

poly	SpatialPolygons object for which to calculate area
byid	boolean indicating whether areas should be calculated for each polygon individually or for the whole object together

Details

Area calculations are for the Polar stereographic grid with major axis of 6378273m and ellipsoid flattening of 1/298.2794111.

Value

Area of polygon (or vector of areas if byid is set to TRUE)

References

Information on Polar Stereographic North projection: https://nsidc.org/data/polar-stereo/ps_grids.html

Examples

```
get_area(reg_info$regions[[1]])
get_area(land, byid = TRUE)
```

get_coords *Extract coordinates from a spatial object of lines and points*

Description

Get coordinates from a spatial object of lines and points. There is no ordering of points returned. Note: This differs from `extract_coords` in that the ordering of the points is NOT considered.

Usage

```
get_coords(my_points)
```

Arguments

`my_points`	spatial object of type `SpatialCollections`, `SpatialPoints`, or `SpatialLines`

Value

n x 2 matrix of coordinates

Examples

```
#Load sample line
ex_line <- as(rm_holes(bg_water[2]), "SpatialLines")
get_coords(ex_line)
```

get_dist *Find euclidean distance*

Description

Finds the euclidean distance between two points (ignoring projection)

Usage

```
get_dist(p1, p2)
```

Arguments

`p1`	vector giving the x and y coordinate pair for the first point
`p2`	vector giving the x and y coordinate pair for the second point

Value

distance value

Examples

```
get_dist(c(1, 2), c(3, 4))
```

get_ind *Find indices in matrix*

Description

Function to find to which matrix indices coordinates correspond (on a 304 x 448 grid)

Usage

```
get_ind(coords, xmn = -3850, ymn = -5350)
```

Arguments

coords	coordinates of interest
xmn	min x (defaults to value for Northern Polar stereographic grid: -3850)
ymn	min y (defaults to value for Northern Polar stereographic grid: -5350)

Value

n x 2 matrix of coordinates on a 304 x 448 grid

Examples

```
dat <- matrix(nrow = 2, ncol = 2, data = c(-2000, 0, 300, 1000))
get_ind(dat)
```

get_init_month *Get initialization month*

Description

Determine initialization month based on month being forecast and lag. Considers lags up to 11 months in advance.

Usage

```
get_init_month(month, lag)
```

Arguments

month	forecast month (integer from 1 to 12 corresponing to month of year)
lag	months in advance prediction is being made (integer from 1 to 11).

Details

Note that this calculation assumes that the prediction for a month is its on first day. This differs from the labeling used in Director et al. (2017) which rounds up to the nearest full month.

Value

integer corresponding to the initialization month

Examples

```
init_month <- get_init_month(month = 10, lag = 4)
init_month
```

get_map *Map one observation or prediction*

Description

Find the mapping vectors for one observation or prediction.

Usage

```
get_map(ice, reg_info, plotting = FALSE, main = "", my_land = land)
```

Arguments

ice	SpatialPolygons object corresponding to the region of ice
reg_info	reg_info list (see reg_info documentation)
plotting	boolean indicating if map should be plotted
main	string specifying the name for the plot
my_land	SpatialPolygons object corresponding to the land

Value

List of the length of the number of regions. Each item in the list is a matrix. Each row of each matrix corresponds to a point in the region's line. The six columns give the fixed point's x-coordinate, the fixed point's y-coordinate, the mapped point's x-coordinate, the mapped point's y-coordinate, the length of the mapping fvectors in the x-direction, and the length of the vectors in the y-direction.

Examples

```
## Not run:
obs <- get_region(dat = obsFeb19811982[1,,], dat_type = "bootstrap",
               level = 15)
obs_map <- get_map(ice = obs, plotting = TRUE, reg_info,
                 main = "Observed Mapping \n February 1985")

## End(Not run)
```

get_region *Get polygons corresponding to regions*

Description

Takes in a matrix and returns a SpatialPolygon object representing regions fitting some criteria. Typically these regions are either where the sea ice concentration is above a certain level or where there is land.

Usage

```
get_region(dat, dat_type, level = NULL, my_land_mat = land_mat,
  my_all_regions = all_regions, use_all = FALSE, land_ind = FALSE,
  xmn = -3850, xmx = 3750, ymn = -5350, ymx = 5850)
```

Arguments

dat	matrix of one of the allowed data types ("gfdl", "bootstrap", or "simple) (see details)
dat_type	string indicating the format of the data: either "gfdl", "bootstrap", or "simple" (see details)
level	concentration level of interest
my_land_mat	binary matrix specifying land locations
my_all_regions	SpatialPolygons object specifying region that will be considered
use_all	boolean, if true indicates to use the full area (overrides land_mat)
land_ind	boolean, if true indicates that the region of interest is the land
xmn	min x dimension (defaults to value for polar stereographic grid: -3850)
xmx	max x dimension (defaults to value for polar stereographic grid: 3750)
ymn	min y dimension (defaults to value for polar stereographic grid: -5350)
ymx	max y dimension (defaults to value for polar stereographic grix: 5850)

Details

For datType = "simple" the values in the dat matrix are indicators of whether the grid box contains ice (1: ice-covered, 0: no ice, NA: land). If datType = "gfdl" or datType = "bootstrap", the values in the matrix correspond to the raw ice concentrations values observed or predicted (including indicators for missing data, land etc.). If datType = "gfdl", the predictions are formatted as in the CM2.5 Forecast-oriented Low-Ocean Resolution (FLOR) model produced by the National Oceanic and Atmospheric Administration's Geophysical Fluid Dynamics Laboratory converted to a Polar Stereographic grid (Vecchi et al. 2014; Msadek et al. 2014). If datType = "bootstrap" the array values are formatted the same as the ice concentration values obtained from the National Aeronautics and Space Administration (NASA) satellites Nimbus-7 SMMR and DMSP SSM/I-SSMIS and processed by the bootstrap algorithm.

Value

region of interest as a `SpatialPolygons` object

References

Bootstrap sea ice concentration: Comiso, J., 2017: Bootstrap sea ice concentrations from Nimbus-7 SMMR and DMSP SSM/I-SSMIS. version 3. Boulder, Colorado USA: NASA National Snow and Ice Data Center Distributed Active Archive Center

CM2.5 Forecast-oriented Low-Ocean Resolution (FLOR) model:Vecchi, Gabriel A., et al. "On the seasonal forecasting of regional tropical cyclone activity." Journal of Climate 27.21 (2014): 7994-8016.

Msadek, R., et al. "Importance of initial conditions in seasonal predictions of Arctic sea ice extent." Geophysical Research Letters 41.14 (2014): 5208-5215.

Examples

```
## Not run:
obs_example <-  get_region(dat = obsFeb2012, dat_type = "bootstrap", level = 15)
plot(land, col = 'grey', border = FALSE)
plot(obs_example, col = "lightblue", add = TRUE)

## End(Not run)
```

`indiv_poly` *Generate an individual polygon from a set of points*

Description

Generate an individual polygon from a set of points

Usage

```
indiv_poly(pts, r, loop_r, reg_info, t1 = NULL, t2 = NULL,
  poly_name = "unspecified")
```

Arguments

`pts`	matrix with two columns giving the coordinates of the generated points
`r`	integer specifying the region
`loop_r`	boolean indicating whether the points are going in a loop
`reg_info`	a `reg_info` list (see documentation for `reg_info`)
`t1`	index of first transition point under consideration in `pts` matrix, NULL if `loop_r == TRUE`
`t2`	index of second transition point under consideration in `pts` matrix, NULL if `loop_r == TRUE`
`poly_name`	string giving name of polygon

interEx *Example of a line that contains self-intersections*

Description

Example of a line that contains self-intersections. We will use it to demonstrate the functions that address these intersections.

Usage

```
interEx
```

Format

n x 2 matrix of coordinates

Examples

```
data(interEx)
plot(interEx)
```

interp_new_pts *Interpolate along region boundaries*

Description

Interpolate contour points that are very close or on the region boundaries.

Usage

```
interp_new_pts(r, new_pts, reg_info, end = TRUE, close = 12.5)
```

Arguments

r	integer indicating for which region the contours are being generated
new_pts	coordinates of the contour
reg_info	a reg_info list (see documentation for reg_info)
end	indicator determining if the points are being interpolated on the ending coordinates or the starting coordinates. Defaults to TRUE.
close	how close a point must be to the line to count as being on it, defaults to 12.5

`inter_start_line` *Add points where line connecting point sequence crosses start_line*

Description

Add points where line connecting point sequence crosses start_line

Usage

```
inter_start_line(r, pts, reg_info)
```

Arguments

`r`	region number
`pts`	matrix with two columns giving coordinates of the points
`reg_info`	a `reg_info` list (see documentation for `reg_info`)

`int_line` *Space points along a line*

Description

The function evenly spaces the number of points that are on one line, `pred_l`, on a different line, `obs_l`

Usage

```
int_line(pred_l, obs_l, plotting = FALSE)
```

Arguments

`pred_l`	predicted line (n1 x 2 matrix of coordinates)
`obs_l`	predicted line (n2 x 2 matrix of coordinates)
`plotting`	boolean indicating whether maps should be plotted

Value

n x 2 matrix of evenly-spaced coordinates

Examples

```
line_space <- int_line(predLEx, obsLEx, plotting = TRUE)
```

keep_line *Keep only spatial lines*

Description

Keep only SpatialLines from a spatial object.

Usage

```
keep_line(my_poly)
```

Arguments

my_poly	SpatialCollections, SpatialPolygons, SpatialPoints, or SpatialLines object

Value

SpatialPolygons object

Examples

```
par(mfrow = c(1, 2))
plot(spatialCollEx, col = "blue", main = "Spatial Collections Object")
line_only <- keep_line(spatialCollEx)
plot(line_only, col = "blue", main = "Spatial Line Only")
```

keep_poly *Keep only spatial polygons*

Description

Keep only SpatialPolygons from a spatial object.

Usage

```
keep_poly(my_poly)
```

Arguments

my_poly	SpatialCollections, SpatialPolygons, SpatialPoints, or SpatialLines object

Value

SpatialPolygons object

Examples

```
par(mfrow = c(1, 2))
plot(spatialCollEx, col = "blue", main = "Spatial Collections Object")
poly_only <- keep_poly(spatialCollEx)
plot(poly_only, col = "blue", main = "Spatial Polygon Only")
```

`land` *Polygon of land*

Description

Land mask as a single `SpatialPolygons` object. The land mask was obtained from the CM2.5 Forecast-oriented Low-Ocean Resolution (FLOR) model produced by the National Oceanic and Atmospheric Administration's Geophysical Fluid Dynamics Laboratory converted to a Polar Stereographic grid. (Vecchi et al. 2014; Msadek et al. 2014). Weights for converting to a polar stereograhic grid were obtained from the spherical coordinate remapping and interpolation package (SCRIP) (Jones 1997).

Usage

```
land
```

Format

`SpatialPolygons` object

References

Vecchi, Gabriel A., et al. "On the seasonal forecasting of regional tropical cyclone activity." Journal of Climate 27.21 (2014): 7994-8016.

Msadek, R., et al. "Importance of initial conditions in seasonal predictions of Arctic sea ice extent." Geophysical Research Letters 41.14 (2014): 5208-5215.

Jones, P.W. "A user's guide for SCRIP: A spherical coordinate remapping and interpolation package." Los Alamos National Laboratory, Los Alamos, NM (1997).

Region Mask: National Snow and Ice Data Center, 2017: Region mask for the northern hemisphere. `http://nsidc.org/data/polar-stereo/tools_masks.html`.

Examples

```
data(land)
plot(land)
```

land_mat *Binary matrix indicating where there is land*

Description

Binary matrix of dimension 304 x 448 with value for 1 for land grid boxes and 0 otherwise. Data are on a north Polar Stereographic grid with the land mask simplified to match model output from the CM2.5 Forecast-oriented Low-Ocean Resolution (FLOR) model produced by the National Oceanic and Atmospheric Administration's Geophysical Fluid Dynamics Laboratory converted to a Polar Stereographic grid (Vecchi et al. 2014; Msadek et al. 2014). Weights for converting to a polar stereograhic grid were obtained from the spherical coordinate remapping and interpolation package (SCRIP) (Jones 1997).

Usage

```
land_mat
```

Format

304 x 448 matix

References

Vecchi, Gabriel A., et al. "On the seasonal forecasting of regional tropical cyclone activity." Journal of Climate 27.21 (2014): 7994-8016.

Msadek, R., et al. "Importance of initial conditionsin seasonal predictions of Arctic sea ice extent." Geophysical Research Letters 41.14 (2014): 5208-5215.

National Center for Atmospheric Research, 2017: Earth system grid at NCAR. `https://www.earthsystemgrid.org/home.html`.

Examples

```
data(land_mat)
image(land_mat, xaxt = "n", yaxt = "n")
```

make_polygons *Create polygon from mapped points*

Description

Create a new polygon from the coordinates of mapped points

Usage

```
make_polygons(r, my_end, poly_name = "unspecified", loop_r)
```

Arguments

`r` integer specifying the region of current interest

`my_end` n x 2 list of mapped points, i.e. the points to which the polygon should extend

`poly_name` character string to name the new polygon (defaults to "unspecified")

`loop_r` boolean indicating whether the points are going in a loop

Value

`SpatialPolygons` object created from the mapped points

Examples

```
new_poly <- make_polygons(r = 5, my_end = mappedPoints, loop_r = FALSE)
plot(new_poly)
```

`mappedPoints` *Example of mapped points*

Description

Example of a set of mapped points organized as an n x 2 matrix of coordinates. This is used to demonstrate the `makePolygons` function.

Usage

```
mappedPoints
```

Format

matrix of 1027 x 2

Examples

```
data(mappedPoints)
head(mappedPoints)
plot(mappedPoints, type = "l")
```

`mapxy` *Get geodetic latitudes and longitudes*

Description

Get corresponding latitude and longitude values for coordinates on a Polar Stereographic North projection grid

Usage

```
mapxy(X, Y, sgn = 1, slat = 70, re = 6378.273, e2 = 0.006693883,
  degrees = TRUE)
```

Arguments

`X`	Polar Stereographic X Coordinate (km)
`Y`	Polar Stereographic Y Coordinate (km)
`sgn`	indicator for Northern hemisphere (defaults to 1)
`slat`	standard latitude (defaults to 70)
`re`	Earth's radius (defaults to 6378.273)
`e2`	eccentricity squared (defaults to 0.006693883)
`degrees`	boolean indicating whether result should be returned in degrees or radians

Value

list with elements `coords$aLat`, the geodetic latitude (degrees, +90 to -90), and `coords$aLon`, the geodetic longitude (degrees, -180 to 180)

References

The equations for this calculation are from Snyder, J. P., 1982, Map Projections Used by the U.S. Geological Survey, Geological Survey Bulletin 1532, U.S. Government Printing Office. See JPL Technical Memorandum 3349-85-101 for further details.

Examples

```
new <- mapxy(100, 300)
new$alat
new$alon
```

map_curr_1 *Sample output from* get_map

Description

Example output from the get_map function for the Central Arctic region.

Usage

```
data(map_curr_1)
```

Format

An object of class matrix with 130 rows and 6 columns.

Examples

```
data(map_curr_1)
```

merged *Sample list of contours*

Description

Example list of ten contours in the form of SpatialPolygons objects

Usage

```
data(merged)
```

Format

An object of class list of length 10.

Examples

```
data(merged)
```

merge_conts *Merge contours*

Description

Merge generated contours for all regions together

Usage

```
merge_conts(conts, full)
```

Arguments

`conts`	list of contours organized as a list of regions by a list of years by a list of samples
`full`	`SpatialPolygons` object for area to be included in all generated contours

Value

Returns a list of contours organized as a list of years by a list of samples

obsLEx *Coordinates of an observed line segment*

Description

Example of the coordinates for an observed line segment. We will use it to demonstrate the `intLine` function.

Usage

```
obsLEx
```

Format

n x 2 matrix of coordinates

Examples

```
data(obsLEx)
head(obsLEx)
```

`obsSep2006_2007` *Observed sea ice September 2006-2007*

Description

The object `observed` is an array obtained from the function `readMonthlyBS`for startYear = 2006 and endYear = 2007. It gives the observed sea ice concentrations arranged in an array of dimension of year x month x lon x lat. The observations are from the monthly sea ice concentration obtained from the National Aeronautics and Space Administration (NASA) satellites Nimbus-7 SMMR and DMSP SSM/I-SSMIS and processed by the bootstrap algorithm. The results are distributed by the National Snow and Ice Data Center (Comiso 2017).

Usage

```
obsSep2006_2007
```

Format

array of dimension of 2 x 12 x 304 x 448 (year x month x longitude x latitude)

References

Comiso, J., 2017: Bootstrap sea ice concentrations from Nimbus-7 SMMR and DMSP SSM/I-SSMIS. version 3. Boulder, Colorado USA: NASA National Snow and Ice Data Center Distributed Active Archive Center

Examples

```
data(obsSep2006_2007)
dim(obsSep2006_2007)
```

`obsSep2008` *Observed sea ice September 2008*

Description

The object `observed` is an binary matrix of dimension lon x lat that indicates whether sea ice concentration was at least 15%. The observations are from the monthly sea ice concentration obtained from the National Aeronautics and Space Administration (NASA) satellites Nimbus-7 SMMR and DMSP SSM/I-SSMIS and processed by the bootstrap algorithm. The results are distributed by the National Snow and Ice Data Center (Comiso 2017).

Usage

```
obsSep2008
```

Format

array of dimension of 2 years x 12 months x longitude x latitude

References

Comiso, J., 2017: Bootstrap sea ice concentrations from Nimbus-7 SMMR and DMSP SSM/I-SSMIS. version 3. Boulder, Colorado USA: NASA National Snow and Ice Data Center Distributed Active Archive Center. doi: https://doi.org/10.5067/7Q8HCCWS4I0R

Examples

```
data(obsSep2008)
dim(obsSep2008)
```

`obs_9_2005_2007` *Observed sea ice for September 2005-2007*

Description

Array of dimension year x longitude by latitude. Binary indicate of whether sea ice concentration of at least 15% was observed. Computed from NASA Boostrap sea ice concentration product (Comiso 2017).

Usage

```
data(obs_9_2005_2007)
```

Format

array

References

Comiso, J., 2017: Bootstrap sea ice concentrations from Nimbus-7 SMMR and DMSP SSM/I-SSMIS. version 3. Boulder, Colorado USA: NASA National Snow and Ice Data Center Distributed Active Archive Center. doi: https://doi.org/10.5067/7Q8HCCWS4I0R

obs_9_2008 *Observed sea ice for September 2008*

Description

Array of dimension longitude by latitude. Binary indicate of whether sea ice concentration of at least 15% was observed. Computed from NASA Boostrap sea ice concentration product (Comiso 2017).

Usage

```
data(obs_9_2008)
```

Format

array

References

Comiso, J., 2017: Bootstrap sea ice concentrations from Nimbus-7 SMMR and DMSP SSM/I-SSMIS. version 3. Boulder, Colorado USA: NASA National Snow and Ice Data Center Distributed Active Archive Center. doi: `https://doi.org/10.5067/7Q8HCCWS4I0R`

obs_maps *Output from* `create_mapping` *function*

Description

Sample output from the `create_mapping` function for observations from September 1993-2007. It is a list of four objects. The first two items in the list, `start_year` and `end_year`, give the first and last year that were mapped. The second two items, `obs_list` and `pred_list`, are lists of arrays with one 3-dimensional array for each region. The first dimension is for the year. The other two dimensions are for the fixed points' y-coordinates, the mapped points' x-coordinates, the mapped points' y-coordinates, the length of the mapping vectors in the x-direction, the length of the vectors in the y-direction, and the angles of the mapping vectors.

Usage

```
data(obs_maps)
```

Format

An object of class `list` of length 4.

Examples

```
data(obs_maps)
```

pars_1 *Sample parameter information for generating a contour*

Description

Example list with two elements, mu_est and sigma_est, which give the mean and covariance from which an example contour can be generated

Usage

```
data(pars_1)
```

Format

An object of class list of length 2.

Examples

```
data(pars_1)
```

ppe_9_2005_2007 *Post-procesed ensemble forecast for 2005-2007*

Description

Array of dimension year by longitude by latitude that gives example forecasts post-processed with a contour model. The initial forecasts are from the European Center for Medium-Range Weather Forecasts (ECMWF) ensemble at a 2.5-month lead time. They have been converted to a Polar stereographic grid. Model output is available from the Sea Ice Prediction Network Predicatability Portal or the Copernicus Climate Change Service data store.

Usage

```
data(ppe_9_2005_2007)
```

Format

array

References

Copernicus Climate Change Service (2019). Description of the c3s seasonal multi-system. https://confluence.ecmwf.int/display/COPSRV/Description+of+the+C3S+seasonal+multi-system

Sea Ice Prediction Network (2019). Sea ice prediction network predictability portal. https://atmos.uw.edu/sipn/.

`ppe_9_2008` *Post-procesed ensemble forecast for 2008*

Description

Array of dimension year by longitude by latitude.The initial forecasts are from the European Center for Medium-Range Weather Forecasts (ECMWF) ensemble at a 2.5-month lead time. They have been converted to a Polar stereographic grid. Model output is available from the Sea Ice Prediction Network Predicatability Portal or the Copernicus Climate Change Service data store.

Usage

```
data(ppe_9_2008)
```

Format

array

References

Copernicus Climate Change Service (2019). Description of the c3s seasonal multi-system. https://confluence.ecmwf.int/display/COPSRV/Description+of+the+C3S+seasonal+multi-system

Sea Ice Prediction Network (2019). Sea ice prediction network predictability portal. https://atmos.uw.edu/sipn/.

`predLEx` *Coordinates of a predicted line segment*

Description

Example of the coordinates for a predicted line segment. We will use it to demonstrate the `intLine` function.

Usage

```
predLEx
```

Format

n x 2 matrix of coordinates

Examples

```
data(predLEx)
head(predLEx)
```

pred_maps *Computed mappings for predictions for September 1993-2008*

Description

Output of the `create_mapping` function for September 1993-2008 using predictions from the European Center for Medium-Range Weather Forecasts (ECMWF) ensemble converted to a Polar stereographic grid.

Usage

```
data(pred_maps)
```

Format

An object of class `list` of length 4.

References

Copernicus Climate Change Service (2019). Description of the c3s seasonal multi-system.`https://confluence.ecmwf.int/display/COPSRV/Description+of+the+C3S+seasonal+multi-system`

Sea Ice Prediction Network (2019). Sea ice prediction network predictability portal. `https://atmos.uw.edu/sipn/.`

Examples

```
data(pred_maps)
```

prob_map *Get probabilities on a grid from contours*

Description

Takes in list of polygon objects from merged function and produces a map of probabilities

Usage

```
prob_map(merged, nX = 304, nY = 448)
```

Arguments

`merged` list of contours organized as a list of years by a list of samples

`nX` dimension in the x (defaults to value for Northern Polar stereographic grid: 304)

`nY` dimension in the y (defaults to value for Northern Polar stereographic grid: 448)

Value

array of dimension number of years by longitude by latitude that gives the proportion of contours in which the grid box is ice-covered

Examples

```
## Not run:  probs <- prob_map(merged)
```

prop_area *Proportion of total area by grid box*

Description

Matrix of dimension longitude by latitude. Elements give the proportion of the total area (within the seas of the Arctic) in each grid box. The sum of all elements is 1.

Usage

```
data(prop_area)
```

Format

array

pt_line_inter *Check if a point crosses a line segment*

Description

Check if a point crosses a line segment

Usage

```
pt_line_inter(pt_to_test, fixed_pts)
```

Arguments

`pt_to_test`	numeric vector of length two giving the point to test
`fixed_pts`	matrix of dimension 2 by 2 giving the line segment to test

quick_run *Simple evaluation of contour-shifting*

Description

Reads in netCDF files of observations and predictions, performs bias correction, and exports a new netCDF file with bias-corrected predictions

Usage

```
quick_run(obs_NCDF, pred_NCDF, pred_years, start_year, month, output_file,
  level, dat_type_obs = "bootstrap", n_train_years = NULL)
```

Arguments

`obs_NCDF`	filepath for observed data array (see details for info about array structure)
`pred_NCDF`	filepath for predicted data array (see details for info about array structure)
`pred_years`	vectors of years for which to make prediction
`start_year`	first year to use when learning model
`month`	month of prediction
`output_file`	filepath for where bias-corrected netCDF file should be stored
`level`	concentration level for which to build contour
`dat_type_obs`	string of either "bootstrap" or "simple" indicating the file type of the observation (see details for info about array structure)
`n_train_years`	number of prior years used in training bias correction

Details

The predicted data array, `pred_NCDF`, should be a netCDF file with a single array of dimension: years x longitude (304) x latitude (448). The variable should be named `ice_ind`. The values in the array should indicate whether each grid box is categorized to contain ice (1: ice-covered, 0: no ice, NA: land). The observed data array, `obs_NCDF`, should be a netCDF file with a single array of dimension: years x longitude (304) x latitude (448). The observed data array, `obs_NCDF`, can be formatted the same as `pred_NCDF` if `dat_type_obs = "simple"`. Alternatively, if `dat_type_obs = "bootstrap"` the array values can be ice concentration values obtained from the National Aeronautics and Space Administration (NASA) satellites Nimbus-7 SMMR and DMSP SSM/I-SSMIS and processed by the bootstrap algorithm. Data should be retained in the same format as given by bootstrap (including indicators for missing data, land etc.). The variable should be named "conc".

Value

netCDF file of dimension years by longitude (304) by latitude (448) with indicators for where ice is predicted after bias correction. (1: ice-covered, 0: not ice, NA: land). Grid boxes will be categorized as ice if their centers are ice covered (within R the bias-corrected contours are not restricted to align to a grid).

References

Comiso, J., 2017: Bootstrap sea ice concentrations from Nimbus-7 SMMR and DMSP SSM/I-SSMIS. version 3. Boulder, Colorado USA: NASA National Snow and Ice Data Center Distributed Active Archive Center

Examples

```
## Not run:
quick_run(obs_NCDF = "/obs.nc", pred_NCDF = "/pred.nc",
         pred_years = c(2001:2013), start_year = 1980, month = 2,
         output_file = "/outputFile.nc", level = 15, dat_type_obs = "simple")

## End(Not run)
```

read_bootstrap *Read individual bootstrap binary file*

Description

Read in individual binary files of monthly observation data. The observations are from the monthly sea ice concentration obtained from the National Aeronautics and Space Administration (NASA) satellites Nimbus-7 SMMR and DMSP SSM/I-SSMIS and processed by the bootstrap algorithm. The results are distributed by the National Snow and Ice Data Center (NSIDC) (Comiso 2017). Functions assume file name conventions are the same as used by NSIDC.

Usage

```
read_bootstrap(file_name, nX = 304, nY = 448)
```

Arguments

`file_name`	file name for binary bootstrap data
`nX`	dimension in the x (defaults to value for Northern Polar stereographic grid: 304)
`nY`	dimension in the y (defaults to value for Northern Polar stereographic grid: 448)

Value

numeric vector of concentrations

References

Comiso, J., 2017: Bootstrap sea ice concentrations from Nimbus-7 SMMR and DMSP SSM/I-SSMIS. version 3. Boulder, Colorado USA: NASA National Snow and Ice Data Center Distributed Active Archive Center. doi: https://doi.org/10.5067/7Q8HCCWS4I0R

Examples

```
## Not run:
#fileName should be the binary file
rawData <- read_bootstrap(file_name)

## End(Not run)
```

`read_monthly_BS` *Read in a set of bootstrap observations over a set of year*

Description

Function to process monthly bootstrap data over multiple years. The observations are from the monthly sea ice concentration obtained from the National Aeronautics and Space Administration (NASA) satellites Nimbus-7 SMMR and DMSP SSM/I-SSMIS and processed by the bootstrap algorithm. The resultsare distributed by the National Snow and Ice Data Center (NSIDC) (Comiso 2017). Functions assume file name conventions are the same as used by NSIDC.

Usage

```
read_monthly_BS(start_year, end_year, file_folder, version, nX = 304,
  nY = 448)
```

Arguments

`start_year`	first year to read in
`end_year`	last year to read in
`file_folder`	folder in which binary files are stored
`version`	either 2 or 3 indicating which version of the bootstrap data you are using
`nX`	longitude dimension
`nY`	latitude dimension

Details

raw binary files for 2012-2013 are included in the package as an example

Value

bootstrap observations sorted into array of dimension: year x month x lon x lat

References

Bootstrap sea ice concentration: Comiso, J., 2017: Bootstrap sea ice concentrations from Nimbus-7 SMMR and DMSP SSM/I-SSMIS. version 3. Boulder, Colorado USA: NASA National Snow and Ice Data Center Distributed Active Archive Center

Examples

```
## Not run:
#my_file_path should be a file path where the 1983 binary files are stored
observed_demo <- read_monthly_BS(start_year = 1983, end_year = 1983,
                                file_folder = my_file_path)

## End(Not run)
```

reg_info *List of information about each region*

Description

A region information list, reg_info, is a list of ten items regions,start_lines, start_lines_coords, start_coords, end_coords, out, lines, dist, loop, and angs. The package contains a reg_info object which is typically what is used for all analyses. However, it would be possible to redefine the regions if desired by making a new reg_info object.

Usage

```
reg_info
```

Format

An object of class list of length 10.

Details

regions: list of SpatialPolygons objects corresponding to each region.

start_lines: list of SpatialLines object giving the line from which each mapping or contour generation will start. For the central Arctic region, a single SpatialPoint is used instead. List ordered the same as reg_info$regions

start_lines_coords: list of matrices giving the coordinates that approximately match reg_info$start_lines, except that they extend to touch the end point of the first and last fixed line. For the central Arctic region, the coordinate of the start_lineis just repeated. List ordered the same as reg_info$regions

start_coords: list of matrices giving the coordinates from which the lines start. List ordered the same as reg_info$regions

end_coords: list of matrices giving the coordinates between the end points of the first and last fixed line. List ordered the same as reg_info$regions

out: list of SpatialPolygons object that border reg_info$start_lines, but are outside the region. These are used when building new polygons to determine if points are outside the region of interest. List ordered the same as reg_info$regions

lines: list giving the SpatialLines objects that correspond to the line on which contours are mapped and built.

`dist`: list for each region with one item for each line `reg_info$lines` giving the lengths at which restrictions on the line lengths occur. The first element for all entries is 0 and the last element is the length of the line. Elements in between refer to the starting and ending lengths on which points cannot be placed. The first list index is ordered the same as `reg_info$regions` and the second list index is ordered as the corresponding lines in `reg_info$lines`

`loop`: vector gives a Boolean for each region. The value `TRUE` indicates that the `lines` are mapped in a circle around a fixed point. The value `FALSE` indicates that the lines are mapped along a line on land. The first element, corresponding to the central Arctic region is `TRUE`. All others are `FALSE`. Elements ordered the same as `reg_info$regions`

`angs`: list of vectors giving the angles of the corresponding `reg_info$lines`. Elements ordered the same as `reg_info$regions`

References

The regions in this object have been substantially modified from the following region mask:

National Snow and Ice Data Center, 2017: Region mask for the northern hemisphere `http://nsidc.org/data/polar-stereo/tools_masks.html`.

Examples

```
data(reg_info)
names(reg_info)
```

---

rm_holes *Remove holes in a polygon*

Description

Remove holes from a `SpatialPolygons` object. Note that this function differs from the function `findHoles` in that it only removes holes contained within the polygon itself, not gaps between the polygon and region boundaries

Usage

```
rm_holes(my_poly, poly_name = "notSpecified")
```

Arguments

`my_poly`	`SpatialPolygon` object
`poly_name`	character string to name polygon (defaults to "notSpecified")

Value

`SpatialPolygon` object with holes removed

Examples

```
with_holes <- bg_water[2]
plot(with_holes, col = "blue", main = "Polygon with Holes")
no_holes <- rm_holes(with_holes)
plot(no_holes, col = "blue", main = "Holes removed")
```

RunMCMC *Run MCMC to Fit Contour Model*

Description

Run MCMC to Fit Contour Model

Usage

```
RunMCMC(n_iter, dists, x, xU_vecs, xU_years, xU_prop_sd, xU_lb, xU_ub, mu,
  mu0, lambda0, sigma, sigma_ind_1, sigma_ind_2, sigma_prop_cov, rho,
  rho0_lb, rho0_ub, rho_prop_sd, sigma0_lb, sigma0_ub, w)
```

Arguments

n_iter	number of iterations to run the MCMC
dists	symmetric matrix of the same dimension as the number of lines being used, specifying distances among starting locations or angles.
x	a matrix of observed distances (y) of dimension number of vectors by number of years
xU_vecs	vector giving the indices of each x value vector in each year that is not observed (vector indices and year indices are paired, so must be ordered the same as xU_years)
xU_years	vector giving the indices of each year in which each x value vector is not observed (vector indices and year indices are paired, so must be ordered the same as xU_vecs)
xU_prop_sd	Standard deviation for proposals for xU
xU_lb	Lower bounds for xU values being sampled (order must match orded of xU_vecs and xU_years)
xU_ub	Upper bounds for xU values being sampled (order must match orded of xU_vecs and xU_years)
mu	vector of the same length as the number of lines which specifies the values from which each element of mu will be initialized in the MCMC.
mu0	vector of the same length as the number of lines which specifies the prior mean for mu.
lambda0	matrix of the same dimension as the number of lines which specifices the prior covariance matrix for mu.

sigma vector of the same length as the number of lines which specifies the values from which each element in sigma will be initialized from

sigma_ind_1 vector giving the first index of each section of sigma's to be sampled together

sigma_ind_2 vector giving the last index of each section of sigma's to be sampled together

sigma_prop_cov covariance matrix of the same length as the number of lines that is used in sampling sigma values

rho double between 0 and 1 from which the value of rho will be initialized

rho0_lb double between 0 and 1 which gives the lower bound of the uniform prior for rho

rho0_ub double between 0 and 1 which gives the upper bound of the uniform prior for rho.

rho_prop_sd standard deviation for the normal proposal distribution used when proposing value for rho in the sampler. Defaults to 0.01

sigma0_lb vector of the same length as the number of lines which specifies the lower bound of the uniform prior for each sigma value

sigma0_ub vector of the same length as the number of lines which specifies the upper bound of the uniform prior for each sigma value.

w Integer specifying how many samples of the parameters will be maintained. Samples from every wth iteration is stored.

Value

List of length 7 that gives the values of the MCMC chain for xU, mu, sigma and rho along with indicators of acceptance on each iteration: xURate, sigmaRate, and rhoRate.

sec_to_interp *Interpolate a section of line*

Description

Interpolate a section of line

Usage

```
sec_to_interp(p1 = NULL, p2 = NULL, bd_r, loop_r = FALSE)
```

Arguments

p1 vector of length two giving the coordinates of the first point

p2 vector length two giving the coordinates of the second point

bd_r matrix with two columns giving the fixed line on which to interpolate

loop_r boolean indicating whether the points are going in a loop

Details

If only p1 is given the point is assumed to be the first in the sequence. If only p2 is given the point is assumed to be the last point in the sequence

---

sipSep2006_2007 *Ensemble sea ice probability for September 2006-2007 (lead time 2.5 months)*

Description

The object `sipSep2006_2007` is an array of the proportion of ensemble members that have sea ice concentrations of at least 15%. The predictions are are from the European Center for Medium-Range Weather Forecasts (ECMWF) ensemble and converted to a Polar stereographic grid.

Usage

```
sipSep2006_2007
```

Format

array of dimension of 2 x 304 x 448 (corresponding to year x longitude x latitude)

References

Comiso, J., 2017: Bootstrap sea ice concentrations from Nimbus-7 SMMR and DMSP SSM/I-SSMIS. version 3. Boulder, Colorado USA: NASA National Snow and Ice Data Center Distributed Active Archive Center. doi: https://doi.org/10.5067/7Q8HCCWS4I0R

Copernicus Climate Change Service (2019). Description of the c3s seasonal multi-system. https://confluence.ecmwf.int/display/COPSRV/Description+of+the+C3S+seasonal+multi-system

Sea Ice Prediction Network (2019). Sea ice prediction network predictability portal. https://atmos.uw.edu/sipn/.

Examples

```
data(sipSep2006_2007)
dim(sipSep2006_2007)
```

sipSep2008 *Ensemble estimated sea ice probability September 2008 (lead time 2.5 months)*

Description

The object `sip2006_2007` is an array of the sea ice probability predicted from the European Center for Medium-Range Weather Forecasts (ECMWF) ensemble converted to a Polar stereographic grid.

The object `sipSep2008` is an array of the proportion of ensemble members that have sea ice concentrations of at least 15%. The predictions are from the European Centerfor Medium-Range Weather Forecasts (ECMWF) ensemble converted to the Polar stereographic grid.

Usage

```
sipSep2008

sipSep2008
```

Format

matrix of dimension 304 x 448 (longitude x latitude)

References

Copernicus Climate Change Service (2019). Description of the c3s seasonal multi-system. https://confluence.ecmwf.int/display/COPSRV/Description+of+the+C3S+seasonal+multi-system

Sea Ice Prediction Network (2019). Sea ice prediction network predictability portal. https://atmos.uw.edu/sipn/.#' @examples data(sipSep2008) dim(sipSep2008)

Copernicus Climate Change Service (2019). Description of the c3s seasonal multi-system.https://confluence.ecmwf.int/display/COPSRV/Description+of+the+C3S+seasonal+multi-system

Sea Ice Prediction Network (2019). Sea ice prediction network predictability portal. https://atmos.uw.edu/sipn/.

spatialCollEx *Spatial collection example*

Description

Example of a `SpatialCollections` object that contains a `SpatialPolygons` object and a `SpatialLines` object

Usage

```
spatialCollEx
```

Format

SpatialCollections object

Examples

```
data(SpatialCollEx)
plot(spatialCollEx)
plot(spatialCollEx@lineobj, col = "red", add = TRUE)
plot(spatialCollEx@polyobj, col = "blue", add = TRUE)
```

---

to_fit *Identify fully ice-covered and ice-free regions*

Description

Determine which regions are completely ice-filled (full) or ice-free (empty) in all years in the training period. Also, make the polygon corresponding to regions that are fully ice-covered.

Usage

```
to_fit(y_obs, reg_info)
```

Arguments

y_obs	list of y values outputted from y_obs function
reg_info	a reg_info list (see documentation for reg_info)

---

train_ind *Find indices on which to train contour model*

Description

Identify the years on which to train accounting for years with missing data

Usage

```
train_ind(maps)
```

Arguments

maps	output of a create_mapping object

`ts_adj_mu` *Trend Adjustment For* `mu`

Description

Trend Adjustment For `mu`

Usage

```
ts_adj_mu(obs_list, forecast_year, train_start_year, train_end_year)
```

Arguments

`obs_list` partial output of `get_map function`, `maps$obs_list[[r]]`, where r is the region of interest

`forecast_year` year to be forecast

`train_start_year` first year in training period

`train_end_year` last year in training period

Value

vector of the length of the number of lines in the mapping that represent by what factor each estimated `mu` should be adjusted

`untwist` *Remove self-intersections*

Description

Function to remove all self-intersections from a contour.

Usage

```
untwist(my_poly, plotting = FALSE, poly_name = "unspecified",
  min_area = 12.5)
```

Arguments

`my_poly` `SpatialPolygons` object from which self-intersections need to be removed

`plotting` boolean indicating if results should be plotted

`poly_name` name for `SpatialPolygons` object to return (defaults to "unspecified")

`min_area` minimum area for any individual polygon

Value

SpatialPolygons object with self-intersections removed

Examples

```
## Not run:
par(mfrow = c(1, 2))
plot(interEx, main = "Original Contour")
noInter <- untwist(interEx, poly_name = "interEx")
plot(noInter, main = "Final Contour")

## End(Not run)
```

untwist_sec *Remove self-intersections from one section of a contour*

Description

Function to correct self-intersections in a section of a line.

Usage

```
untwist_sec(line, tol = 0, eps = 0.25)
```

Arguments

line	N x 2 matrix of coordinates
tol	how much of a difference between the original line and the simplified line is allowed
eps	how much to increase tol by on each iteration

Value

n x 2 matrix of the new coordinates with self-intersections removed

Examples

```
par(mfrow = c(1, 2))
plot(currSecEx, type = "l", main = "Original Line Section", xlab = "", ylab = "")
new_sec <- untwist_sec(currSecEx)
plot(new_sec, type = "l", main = "New Line Section", xlab = "", ylab =  "")
```

wght_mod *Function to weight two models*

Description

Function to weight two models

Usage

```
wght_mod(w, mod1, mod2)
```

Arguments

w	weight on model 1
mod1	array with estimated sea ice probability from model 1. Dimensions are nuumber of training years x lon x lat.
mod2	array with estimated sea ice probability from model 1. Dimensions are nuumber of training years x lon x lat.

Examples

```
## Not run:
weight <- fit_weights(mod1 = clim_9_2005_2007, mod2 = ppe_9_2005_2007,
obs = obs_9_2005_2007, prop_area = prop_area)
wght_mod(w = weight, mod1 = clim_9_2008, mod2 = ppe_9_2008)

## End(Not run)
```

y_obs *Compute y*

Description

Compute y values from the output of the `create_mapping` object

Usage

```
y_obs(maps, reg_info)
```

Arguments

maps	output of the `create_mapping` function
reg_info	a `reg_info` list (see documentation for `reg_info`)

Value

List of matrices, one per region, giving the observed y values. Each row corresponds to the lines in L and each column corresponds to a training year

Examples

```
y_obs <- y_obs(maps = obs_maps, reg_info)
```

Index