Package ‘crawl’

April 22, 2017

Type Package

Title Fit Continuous-Time Correlated Random Walk Models to Animal Movement Data

Version 2.1.1

Date 2017-04-21

Author Devin S. Johnson

Maintainer Devin S. Johnson <devin.johnson@noaa.gov>

Depends R (>= 2.10)

Imports mvtnorm, Rcpp (>= 0.11.1), methods, shiny, dplyr, raster, gdistance, sp

LinkingTo Rcpp, RcppArmadillo

Suggests ggplot2, rgdal, argosfilter, tidyr, lubridate, xts, broom, sf, doParallel

Description Fit continuous-time correlated random walk models with time indexed covariates to animal telemetry data. The model is fit using the Kalman-filter on a state space version of the continuous-time stochastic movement process.

License CC0

LazyLoad yes

ByteCompile TRUE

NeedsCompilation yes

RoxygenNote 6.0.1

Repository CRAN

Date/Publication 2017-04-21 22:48:36 UTC

R topics documented:

crawl-package .......................................................... 2
aic.crw ................................................................. 3
The Correlated RAndom Walk Library (I know it is not an R library, but, "crawp" did not sound as good) of R functions was designed for fitting continuous-time correlated random walk (CTCRW) models with time indexed covariates. The model is fit using the Kalman-Filter on a state space version of the continuous-time stochastic movement process.
recommendations and conclusions presented here are those of the authors and this software should not be construed as official communication by NMFS, NOAA, or the U.S. Dept. of Commerce. In addition, reference to trade names does not imply endorsement by the National Marine Fisheries Service, NOAA. While the best efforts have been made to insure the highest quality, tools such as this are under constant development and are subject to change.

Author(s)

Devin S. Johnson
Maintainer: Devin S. Johnson <devin.johnson@noaa.gov>

References


---

**aic.crw**

*Calculates AIC for all objects of class crwFit listed as arguments*

Description

AIC, delta AIC, and Akaike weights for all models listed as arguments.

Usage

```r
aic.crw(...)
```

Arguments

```
...  a series of crwFit objects
```

Details

The function can either be executed with a series of `crwFit` objects (see `crwMLE`) without the `.crwFit` suffix or the function can be called without any arguments and it will search out all `crwFit` objects in the current workspace and produce the model selection table for all `crwFit` objects in the workspace. Caution should be used when executing the function in this way. ALL `crwFit` objects will be included whether or not the same locations are used! For all of the models listed as arguments (or in the workspace), AIC, delta AIC, and Akaike weights will be calculated.

Value

A table, sorted from lowest AIC value to highest.

Author(s)

Devin S. Johnson
### argosDiag2Cov

*Transform Argos diagnostic data to covariance matrix form*

#### Description

Using this function the user can transform the Argos diagnostic data for location error into a form usable as a covariance matrix to approximate the location error with a bivariate Gaussian distribution. The resulting data.frame should be attached back to the data with `cbind` to use with the `crwMLE` function.

#### Usage

```r
argosDiag2Cov(Major, Minor, Orientation)
```

#### Arguments

- **Major**: A vector containing the major axis information for each observation (na values are ok)
- **Minor**: A vector containing the minor axis information for each observation (na values are ok)
- **Orientation**: A vector containing the angle orientation of the Major axis from North (na values are ok)

#### Value

A data.frame with the following columns:

- **ln.sd.x**: The log standard deviation of the location error in the x coordinate
- **ln.sd.y**: The log standard deviation of the location error in the x coordinate
- **rho**: The correlation of the bivariate location error ellipse

#### Author(s)

Devin S. Johnson

---

### as.flat

*‘Flattening’ a list-form crwPredict object into a data.frame*

#### Description

“Flattens” a list form `crwPredict` object into a flat data.frame.

#### Usage

```r
as.flat(predObj)
```
Arguments

predObj  A crwPredict object

Value

a \texttt{data.frame} version of a crwPredict list with columns for the state standard errors

Author(s)

Devin S. Johnson

See Also

\texttt{northernFurSeal} for use example

\begin{tabular}{ll}
beardedSeals & \textit{Bearded Seal Location Data} \\
\end{tabular}

Description

Bearded Seal Location Data

Format

A data frame with 27,548 observations on 3 bearded seals in Alaska:

\begin{verbatim}
  deployid  Unique animal ID
  ptt       Hardware ID
  instr     Hardware type
  date_time Time of location
  type      Location type
  quality   Argos location quality
  latitude  Observed latitude
  longitude Observed longitude
  error_radius  Argos error radius
  error_seminajor_axis Argos error ellipse major axis length
  error_seminor_axis  Argos error ellipse minor axis length
  error_ellipse_orientation Argos error ellipse degree orientation
\end{verbatim}

Source

Marine Mammal Laboratory, Alaska Fisheries Science Center, National Marine Fisheries Service, NOAA 7600 Sand Point Way NE Seattle, WA 98115
check_csv

Start a shiny app to check data stored in a .csv file for model fitting with \texttt{crwMLE} function.

\textbf{Description}

Users can start a beta version of Shiny app that allows for data checking and basic location projection.

\textbf{Usage}

\begin{verbatim}
check_csv()
\end{verbatim}

crwMLE

Fit Continuous-Time Correlated Random Walk Models to Animal Telemetry Data

\textbf{Description}

The function uses the Kalman filter to estimate movement parameters in a state-space version of the continuous-time movement model. Separate models are specified for movement portion and the location error portion. Each model can depend on time indexed covariates. A “haul out” model where movement is allowed to completely stop, as well as, a random drift model can be fit with this function.

\textbf{Usage}

\begin{verbatim}
crwMLE(mov.model = \sim 1, err.model = NULL, activity = NULL, drift = FALSE, 
data, coord = c("x", "y"), Time.name, initial.state, theta, fixPar, 
method = "Nelder-Mead", control = NULL, constr = list(lower = -Inf, 
upper = Inf), prior = NULL, need.hess = TRUE, initialSANN = list(maxit = 
200), attempts = 1)
\end{verbatim}

\textbf{Arguments}

\begin{itemize}
  \item \texttt{mov.model} formula object specifying the time indexed covariates for movement parameters.
  \item \texttt{err.model} A 2-element list of formula objects specifying the time indexed covariates for location error parameters.
  \item \texttt{activity} formula object giving the covariate for the activity (i.e., stopped or fully moving) portion of the model.
  \item \texttt{drift} logical indicating whether or not to include a random drift component. For most data this is usually not necessary. See \texttt{northernFurSeal} for an example using a drift model.
\end{itemize}
data frame object containing telemetry and covariate data. A `SpatialPointsDataFrame` object from the package `sp` or an `sf` object from the `sf` package with a geometry column of type `sfc_POINT`. `spacetime` will also be accepted. Values for coords will be taken from the spatial data set and ignored in the arguments.

coord A 2-vector of character values giving the names of the "X" and "Y" coordinates in data.

Time.name character indicating name of the location time column

initial.state list object containing the initial state of the Kalman filter.

theta starting values for parameter optimization.

fixPar Values of parameters which are held fixed to the given value.

method Optimization method that is passed to `optim`.

control Control list which is passed to `optim`.

constr Named list with elements `lower` and `upper` that are vectors the same length as theta giving the box constraints for the parameters.

prior A function returning the log-density function of the parameter prior distribution. THIS MUST BE A FUNCTION OF ONLY THE FREE PARAMETERS. Any fixed parameters should not be included.

need.hess A logical value which decides whether or not to evaluate the Hessian for parameter standard errors

initialSANN Control list for `optim` when simulated annealing is used for obtaining start values. See details

attempts The number of times likelihood optimization will be attempted

Details

A full model specification involves 4 components: a movement model, a stopping model, 2 location error models, and a drift indication. The movement model (`mov.model`) specifies how the movement parameters should vary over time. This is a function of specified, time-indexed, covariates. The movement parameters (sigma for velocity variation and beta for velocity autocorrelation) are both modeled with a log link as \( \text{par} = \exp(\eta) \), where \( \eta \) is the linear predictor based on the covariates. The `err.model` specification is a list of 2 such models, one for "longitude" and one for "latitude" (in that order) location error. If only one location error model is given, it is used for both coordinates (parameter values as well). If `drift.model` is set to TRUE, then, 2 additional parameters are estimated for the drift process, a drift variance and a beta multiplier. If `polar.coord=TRUE` then the ad-hoc longitude correction factor described by Johnson et al. (2008) (Ecology 89:1208-1215) is used to adjust the variance scale for the longitude model.

The `initial.state` is a list with the following elements (with the exact names):

- `a` A vector with initial state values. It has 4 elements (x location at time 1, x velocity at time 1, y location at time 1, y velocity at time 1) for non-drift models and 6 elements for drift models (x location at time 1, x velocity at time 1, x drift velocity at time 1, etc...).

- `theta` and `fixPar` are vectors with the appropriate number of parameters. `theta` contains only those parameters which are to be estimated, while `fixPar` contains all parameter values with `NA` for parameters which are to be estimated.
The data set specified by `data` must contain a numeric or POSIXct column which is used as the time index for analysis. The column name is specified by the `time.name` argument. If a POSIXct column is used it is internally converted to a numeric vector with units of hours. The spacetime package supports an STIDF object that contains slots for both spatial and time series data types. If `data` is of class STIDF then the spatial and temporal information are automatically extracted and `polar.coord`, `time.name` and `coord` are not required. If your data are not compatible with these data structures, it is better to convert it yourself prior to analysis with `crawl`. Also, for stopping models, the stopping covariate must be between 0 and 1 inclusive, with 1 representing complete stop of the animal (no true movement, however, location error can still occur) and 0 represent unhindered movement. The coordinate location should have `NA` where no location is recorded, but there is a change in the movement covariates.

The CTCRW models can be difficult to provide good initial values for optimization. If `initial.SANN` is specified then simulated annealing is used first to obtain starting values for the specified optimization method. If simulated annealing is used first, then the returned `init` list of the `crwFit` object will be a list with the results of the simulated annealing optimization.

**Value**

A list with the following elements:

- `par` : Parameter maximum likelihood estimates (including fixed parameters)
- `estPar` : MLE without fixed parameters
- `se` : Standard error of MLE
- `ci` : 95% confidence intervals for parameters
- `Cmat` : Parameter covariance matrix
- `loglik` : Maximized log-likelihood value
- `aic` : Model AIC value
- `initial.state` : Initial state provided to `crwMLE` for model fitting
- `coord` : Coordinate names provided for fitting
- `fixPar` : Fixed parameter values provided
- `convergence` : Indicator of convergence (0 = converged)
- `message` : Messages given by `optim` during parameter optimization
- `activity` : Model provided for stopping variable
- `drift` : Logical value indicating random drift model
- `mov.model` : Model description for movement component
- `err.model` : Model description for location error component
- `n.par` : number of parameters
- `nms` : parameter names
- `n.mov` : number of movement parameters
- `n.errX` : number or location error parameters for "longitude" error model
- `n.errY` : number or location error parameters for "latitude" error model
- `stop.mf` : covariate for stop indication in stopping models
crwN2ll

polar.coord Logical indicating coordinates are polar latitude and longitude
init Initial values for parameter optimization
data Original data.frame used to fit the model
lower The lower parameter bounds
upper The upper parameter bounds
need.hess Logical value
runTime Time used to fit model

Author(s)
Devin S. Johnson, Josh M. London

crwN2ll -2 * log-likelihood for CTCRW models

Description
This function is designed for primary use within the crwMLE model fitting function. But, it can be accessed for advanced R and crawl users. Uses the state-space parameterization and Kalman filter method presented in Johnson et al. (2008).

Usage
```r
crwN2ll(theta, fixPar, y, noObs, delta, a, P, mov.mf, err.mfX, err.mfY,
    rho = NULL, activity = NULL, n.errX, n.errY, n.mov, driftMod, prior,
    need.hess, constr = list(lower = -Inf, upper = Inf))
```

Arguments
theta parameter values.
fixPar values of parameters held fixed (contains NA for theta values).
y N by 2 matrix of coordinates with the longitude coordinate in the first column.
noObs vector with 1 for unobserved locations, and 0 for observed locations.
delta time difference to next location.
a initial state mean.
P initial state covariance matrix
mov.mf Movement covariate data.
err.mfX longitude error covariate data.
err.mfY latitude error covariate data.
rho A vector of known correlation coefficients for the error model, typically used for modern ARGOS data.
activity Stopping covariate (= 0 if animal is not moving).
n.errX  number or longitude error parameters.
n.errY  number of latitude error parameters.
n.mov  number or movement parameters.
driftMod  Logical. indicates whether a drift model is specified.
prior  Function of theta that returns the log-density of the prior
need.hess  Whether or not the Hessian will need to be calculated from this call
constr  Named list giving the parameter constraints

Details
This function calls compiled C++ code which can be viewed in the src directory of the crawl source package.

Value
-2 * log-likelihood value for specified CTCRW model.

Author(s)
Devin S. Johnson

References

See Also
crwMLE

crwPostIS  Simulate a value from the posterior distribution of a CTCRW model

Description
The crwPostIS draws a set of states from the posterior distribution of a fitted CTCRW model. The draw is either conditioned on the fitted parameter values or "full" posterior draw with approximated parameter posterior.

Usage
crwPostIS(object.sim, fullPost = TRUE, df = Inf, scale = 1, thetaSamp = NULL)
crwPostIS

Arguments

- `object.sim`: A `crwSimulator` object from `crwSimulator`.
- `fullPost`: logical. Draw parameter values as well to simulate full posterior.
- `df`: degrees of freedom for multivariate t distribution approximation to parameter posterior.
- `scale`: Extra scaling factor for t distribution approximation.
- `thetaSamp`: If multiple parameter samples are available in `object.sim`, setting `thetaSamp=n` will use the nth sample. Defaults to the last.

Details

The `crwPostIS` draws a posterior sample of the track state matrices. If `fullPost` was set to `TRUE` when the `object.sim` was build in `crwSimulator` then a pseudo-posterior draw will be made by first sampling a parameter value from a multivariate t distribution which approximates the marginal posterior distribution of the parameters. The covariance matrix from the fitted model object is used to scale the MVt approximation. In addition, the factor "scale" can be used to further adjust the approximation. Further, the parameter simulations are centered on the fitted values.

To correct for the MVt approximation, the importance sampling weight is also supplied. When calculating averages of track functions for Bayes estimates one should use the importance sampling weights to calculate a weighted average (normalizing first, so the weights sum to 1).

Value

List with the following elements:

- `alpha.sim.y`: A matrix a simulated latitude state values.
- `alpha.sim.x`: Matrix of simulated longitude state values.
- `locType`: Indicates prediction types with a "p" or observation times with an "o".
- `time`: Initial state covariance for latitude.
- `loglik`: log likelihood of simulated parameter.
- `par`: Simulated parameter value.
- `log.isw`: non normalized log importance sampling weight.

Author(s)

Devin S. Johnson

See Also

See `demo(northernFurSealDemo)` for example.
crwPredict

Predict animal locations and velocities using a fitted CTCRW model and calculate measurement error fit statistics

Description

The crwMElfilter function uses a fitted model object from crwMLE to predict animal locations (with estimated uncertainty) at times in the original data set and supplemented by times in predTime. If speedEst is set to TRUE, then animal log-speed is also estimated. In addition, the measurement error shock detection filter of de Jong and Penzer (1998) is also calculated to provide a measure for outlier detection.

Usage

crwpredict(object, predTime = NULL, flat = TRUE, ...)

Arguments

object
A model object from crwMLE.
predTime
vector of additional prediction times (numeric or POSIXct). Alternatively, a character vector specifying a time interval (see Details).
flat
logical. Should the result be returned as a flat data.frame.
...
Additional arguments for testing new features

Details

The requirements for data are the same as those for fitting the model in crwMLE.

- ("predTime") predTime can be either passed as a separate vector of POSIXct or numeric values for additional prediction times beyond the observed times. If the original data were provided as a POSIXct type, then crwpredict can derive a sequence of regularly spaced prediction times from the original data. This is specified by providing a character string that corresponds to the by argument of the seq.POSIXct function (e.g. '1 hour', '30 mins'). crwpredict will round the first observed time up to the nearest unit (e.g. '1 hour' will round up to the nearest minute) and start the sequence from there. The last observation time is truncated down to the nearest unit to specify the end time.

Value

List with the following elements:

originalData
A data.frame with is data merged with predTime.
alpha.hat
Predicted state
Var.hat
array where Var.hat[,i] is the prediction covariance matrix for alpha.hat[,i].
fit.test
A data.frame of chi-square fit (df=2) statistics and naive (pointwise) p-values.

If flat is set to TRUE then a data set is returned with the columns of the original data plus the state estimates, standard errors (se), speed estimates, and the fit statistics and naive p-values.
**Author(s)**
Devin S. Johnson

**References**

---

**crwPredictPlot**  
*Plot CRW predicted object*

**Description**
Creates 2 types of plots of a crwPredict object: a plot of both coordinate axes with prediction intervals and a plot of just observed locations and predicted locations.

**Usage**
crwPredictPlot(object, plotType = "ll", ...)

**Arguments**
- **object**: crwPredict object.
- **plotType**: type of plot has to be one of the following: “map” or “ll” (default).
- **...**: Further arguments passed to plotting commands.

**Value**
A plot.

**Author(s)**
Devin S. Johnson and Sebastian Luque

**See Also**
See demo(northernFurSealDemo) for additional examples.
crwSamplePar

Create a weighted importance sample for posterior predictive track simulation.

Description

The crwSamplePar function uses a fitted model object from crwMLE and a set of prediction times to construct a list from which crwPostIS will draw a sample from either the posterior distribution of the state vectors conditional on fitted parameters or a full posterior draw from an importance sample of the parameters.

Usage

crwSamplePar(object.sim, method = "IS", size = 1000, df = Inf, grid.pts = 1, crit = 2.5, scale = 1, force.quad)

Arguments

- object.sim: A simulation object from crwSimulator.
- method: Method for obtaining weights for movement parameter samples
- size: Size of the parameter importance sample
- df: Degrees of freedom for the t approximation to the parameter posterior
- grid.pts: Grid size for method="quadrature"
- crit: Criterion for deciding "significance" of quadrature points (difference in log-likelihood)
- scale: Scale multiplier for the covariance matrix of the t approximation
- force.quad: A logical indicating whether or not to force the execution of the quadrature method for large parameter vectors.

Details

The crwSamplePar function uses the information in a crwSimulator object to create a set of weights for importance sample-resampling of parameters in a full posterior sample of parameters and locations using crwPostIS. This function is usually called from crwPostIS. The average user should have no need to call this function directly.

Value

List with the following elements:

- x: Longitude coordinate with NA at prediction times
- y: Similar to above for latitude
- locType: Indicates prediction types with a "p" or observation times with an "o"
- P1.y: Initial state covariance for latitude
**crwSamplePar**

- `P1.x`: Initial state covariance for longitude
- `a1.y`: Initial latitude state
- `a1.x`: Initial longitude state
- `n.errX`: Number of longitude error model parameters
- `n.errY`: Number of latitude error model parameters
- `delta`: Vector of time differences
- `driftMod`: Logical. Indicates random drift model
- `stopMod`: Logical. Indicates stop model fitted
- `stop.mf`: Stop model design matrix
- `err.mfX`: Longitude error model design matrix
- `err.mfY`: Latitude error model design matrix
- `mov.mf`: Movement model design matrix
- `fixPar`: Fixed values for parameters in model fitting
- `Cmat`: Covariance matrix for parameter sampling distribution
- `Lmat`: Cholesky decomposition of Cmat
- `par`: Fitted parameter values
- `N`: Total number of locations
- `loglik`: Log likelihood of the fitted model
- `Time`: Vector of observation times
- `coord`: Names of coordinate vectors in original data
- `Time.name`: Name of the observation times vector in the original data
- `thetaSampList`: A list containing a data frame of parameter vectors and their associated probabilities for a resample

**Author(s)**

Devin S. Johnson

**See Also**

See `demo(northernFurSealDemo)` for example.
crwSimulator

Construct a posterior simulation object for the CTCRW state vectors

Description

The crwSimulator function uses a fitted model object from crwMLE and a set of prediction times to construct a list from which crwPostIS will draw a sample from either the posterior distribution of the state vectors conditional on fitted parameters or a full posterior draw from an importance sample of the parameters.

Usage

```r
crwSimulator(object.crwFit, predTime = NULL, method = "IS", parIS = 1000,
              df = Inf, grid.eps = 1, crit = 2.5, scale = 1, force.quad)
```

Arguments

- **object.crwFit**: A model object from crwMLE.
- **predTime**: vector of additional prediction times.
- **method**: Method for obtaining weights for movement parameter samples
- **parIS**: Size of the parameter importance sample
- **df**: Degrees of freedom for the t approximation to the parameter posterior
- **grid.eps**: Grid size for method="quadrature"
- **crit**: Criterion for deciding "significance" of quadrature points (difference in log-likelihood)
- **scale**: Scale multiplier for the covariance matrix of the t approximation
- **force.quad**: A logical indicating whether or not to force the execution of the quadrature method for large parameter vectors.

Details

The crwSimulator function produces a list and preprocesses the necessary components for repeated track simulation from a fitted CTCRW model from crwMLE. The method argument can be one of "IS" or "quadrature". If method="IS" is chosen standard importance sampling will be used to calculate the appropriate weights via t proposal with df degrees of freedom. If df=Inf (default) then a multivariate normal distribution is used to approximate the parameter posterior. If method="quadrature", then a regular grid over the posterior is used to calculate the weights. The argument grid.eps controls the quadrature grid. The arguments are approximately the upper and lower limit in terms of standard deviations of the posterior. The default is grid.eps, in units of 1sd. If object.crwFit was fitted with crwArgoFilter, then the returned list will also include p.out, which is the approximate probability that the observation is an outlier.
Value

List with the following elements:

- **x**: Longitude coordinate with NA at prediction times
- **y**: Similar to above for latitude
- **locType**: Indicates prediction types with a "p" or observation times with an "o"
- **P1.y**: Initial state covariance for latitude
- **P1.x**: Initial state covariance for longitude
- **a1.y**: Initial latitude state
- **a1.x**: Initial longitude state
- **n.errX**: number of longitude error model parameters
- **n.errY**: number of latitude error model parameters
- **delta**: vector of time differences
- **driftMod**: Logical. indicates random drift model
- **stopMod**: Logical. Indicated stop model fitted
- **stop.mf**: stop model design matrix
- **err.mfX**: Longitude error model design matrix
- **err.mfY**: Latitude error model design matrix
- **mov.mf**: Movement model design matrix
- **fixPar**: Fixed values for parameters in model fitting
- **Cmat**: Covariance matrix for parameter sampling distribution
- **Lmat**: Cholesky decomposition of Cmat
- **par**: fitted parameter values
- **N**: Total number of locations
- **loglik**: log likelihood of the fitted model
- **Time**: vector of observation times
- **coord**: names of coordinate vectors in original data
- **Time.name**: Name of the observation times vector in the original data
- **thetaSamplList**: A list containing a data frame of parameter vectors and their associated probabilities for a resample

Author(s)

Devin S. Johnson

See Also

See demo(northernFurSealDemo) for example.
displayPar | Display the order of parameters along with fixed values and starting values

**Description**

This function takes the model specification arguments to the `crwMLE` function and displays a table with the parameter names in the order that `crwMLE` will use during model fitting. This is useful for specifying values for the `fixPar` or `theta` (starting values for free parameters) arguments.

**Usage**

```r
displayPar(mov.model = ~1, err.model = NULL, activity = NULL, 
           drift = FALSE, data, Time.name, theta, fixPar, ...)
```

**Arguments**

- `mov.model` formula object specifying the time indexed covariates for movement parameters.
- `err.model` A 2-element list of formula objects specifying the time indexed covariates for location error parameters.
- `activity` formula object giving the covariate for the stopping portion of the model.
- `drift` logical indicating whether or not to include a random drift component.
- `data` data.frame object containing telemetry and covariate data. A `SpatialPointsDataFrame` object from the package 'sp' will also be accepted.
- `Time.name` character indicating name of the location time column
- `theta` starting values for parameter optimization.
- `fixPar` Values of parameters which are held fixed to the given value.
- `...` Additional arguments (probably for testing new features.)

**Value**

A data frame with the following columns

- `ParNames` The names of the parameters specified by the arguments.
- `fixPar` The values specified by the `fixPar` argument for fixed values of the parameters. In model fitting, these values will remain fixed and will not be estimated.
- `thetaIndex` This column provides the index of each element of the theta argument and to which parameter it corresponds.
- `thetaStart` If a value is given for the `theta` argument it will be placed in this column and its elements will correspond to the `thetaIndex` column.

**Author(s)**

Devin S. Johnson
**expandPred**

See Also

demo(northernFurSealDemo) for example.

---

**expandPred**

Expand a time indexed data set with additional prediction times

---

**Description**

Expands a covariate data frame (or vector) that has a separate time index by inserting prediction times and duplicating the covariate values for all prediction time between subsequent data times.

**Usage**

`expandPred(x, Time = "Time", predTime, time.col = FALSE)`

**Arguments**

- **x**: Data to be expanded.
- **Time**: Either a character naming the column which contains original time values, or a numeric vector of original times.
- **predTime**: Prediction times to expand data.
- **time.col**: Logical value indicating whether to attach the new times to the expanded data.

**Value**

data.frame expanded by `predTime`

**Author(s)**

Devin S. Johnson

**Examples**

```r
#library(crawl)
origTime <- c(1:10)
x <- cbind(rnorm(10), c(21:30))
predTime <- seq(1,10, by=0.25)
expandPred(x, Time=origTime, predTime, time.col=TRUE)
```
fillCols  \hspace{1cm} \textit{Fill missing values in data set (or matrix) columns for which there is a single unique value}

Description

Looks for columns in a data set that have a single unique non-missing value and fills in all NA with that value

Usage

\texttt{fillCols(data)}

Arguments

data \hspace{1cm} \texttt{data.frame}

Value

\texttt{data.frame}

Author(s)

Devin S. Johnson

Examples

\begin{verbatim}
#library(crawl)
data1 <- data.frame(constVals=rep(c(1,NA),5), vals=1:10)
data1[5,2] <- NA
data1
data2 <- fillCols(data1)
data2

mat1 <- matrix(c(rep(c(1,NA),5), 1:10), ncol=2)
mat1[5,2] <- NA
mat1
mat2 <- fillCols(mat1)
mat2
\end{verbatim}
**fix_path**

*Project path away from restricted areas*

**Description**

Corrects a path so that it does not travel through a restricted area.

**Usage**

`fix_path(xy, time, res_raster, trans)`

**Arguments**

- `xy` Coordinate locations for the path. Can be one of the following classes: (1) a two column matrix, (2) 'SpatialPoints' or 'SpatialPointsDataFrame' object from the `sp` package, (3) 'crwPredict' object from the `crwPredict` function (4) 'crwIS' object from the `crwPostIS` function
- `time` A vector of times associated with `xy` locations
- `res_raster` An indicator raster object with cells = 1 if it is 'off-limits' and 0 elsewire.
- `trans` A transition matrix object from the `gdistance` package.

**Value**

Either matrix or 'SpatialPoints' object with path projected around restricted areas

---

**flatten**

*'Flattening' a list-form crwPredict object into a data.frame*

**Description**

“Flattens” a list form `crwPredict` object into a flat data.frame.

**Usage**

`flatten(predObj)`

**Arguments**

- `predObj` A `crwPredict` object

**Value**

A `data.frame` version of a `crwPredict` list with columns for the state standard errors
get_restricted_segments

Find the sections of a path that pass through a restricted area

Description

This function is used to identify sections of a path that pass through restricted areas. The CTCRW model in crawl cannot actively steer paths away from restricted areas as it knows nothing of spatial information. So, this function will identify areas that for which the unrestrained path passes through these areas. If the path/points end within the restricted area, those records will be removed. The user can then use this information to adjust the path as desired.

Usage

get_restricted_segments(xy, res_raster)

Arguments

xy

A SpatialPoints object from the sp package or a 2-column matrix of x and y locations

res_raster

A raster object from the raster package that indicates restricted areas with a 1, else 0 for unrestricted areas.

Value

A data.frame with each row associated with each section of the path that crosses a restricted area. The columns provide the start and end row indices of xy where the section occurs and the previous and post locations that are in unrestricted space.

Author(s)

Josh M. London (josh.london@noaa.gov)
harborSeal

Harbor seal relocation data set used in Johnson et al. (2008)

Description

Harbor seal relocation data set used in Johnson et al. (2008)

Format

A data frame with 7059 observations on the following 5 variables.

- **Time** a numeric vector.
- **latitude** a numeric vector.
- **longitude** a numeric vector.
- **DryTime** a numeric vector.
- **Argos_loc_class** a factor with levels 0 1 2 3 A B.

Author(s)

Devin S. Johnson

Source

Marine Mammal Laboratory, Alaska Fisheries Science Center, National Marine Fisheries Service, NOAA 7600 Sand Point Way NE Seattle, WA 98115

References


intTo POSIX

Reverse as.numeric command that is performed on a vector of type POSIXct

Description

Takes integer value produced by as.numeric(x), where x is a POSIXct vector and returns it to a POSIXct vector

Usage

intTo POSIX(timeVector, tz = "GMT")
mergeTrackStop

Arguments

- **timeVector**: A vector of integers produced by `as.numeric` applied to a POSIXct vector.
- **tz**: Time zone of the vector (see `as.POSIXct`).

Value

- POSIXct vector

Note

There is no check that `as.numeric` applied to a POSIX vector produced `timeVector`. So, caution is required in using this function. It was included simply because I have found it useful.

Author(s)

Devin S. Johnson

Examples

```r
#library(crawl)
timeVector <- as.numeric(Sys.time())
timeVector
intTo POSIX(timeVector, tz="")
```

Description

The function merges a location data set with a stopping variable data set.

Usage

```r
mergeTrackStop(data, stopData, Time.name = "Time", interp = c("zeros", "ma0"), win = 2, constCol)
```

Arguments

- **data**: Location data.
- **stopData**: stopping variable data set.
- **Time.name**: character naming time index variable in both data sets.
- **interp**: method of interpolation.
- **win**: window for "ma0" interpolation method.
- **constCol**: columns in data for which the user would like to be constant, such as id or sex.
northernFurSeal

Details

Simply merges the data frames and interpolates based on the chosen method. Both data frames have to use the same name for the time variable. Also contains stopType which = "o" if observed or "p" for interpolated.

The merged data is truncated to the first and last time in the location data set. Missing values in the stopping variable data set can be interpolated by replacing them with zeros (full movement) or first replacing with zeros then using a moving average to smooth the data. Only the missing values are then replace with this smoothed data. This allows a smooth transition to full movement.

Value

Merged data.frame with new column from stopData. Missing values in the stopping variable will be interpolated

Author(s)

Devin S. Johnson

Examples

```r
track <- data.frame(TimeVar=sort(runif(20,0,20)), x=1:20, y=20:1)
track
stopData <- data.frame(TimeVar=0:29, stopVar=round(runif(30)))
stopData
mergeTrackStop(track, stopData, Time.name="TimeVar")
```

northernFurSeal

Northern fur seal pup relocation data set used in Johnson et al. (2008)

Description

Northern fur seal pup relocation data set used in Johnson et al. (2008)

Format

A data frame with 795 observations on the following 4 variables:

- **Time** a numeric vector.
- **Argos_loc_class** a factor with levels 0 1 2 3 A.
- **latitude** a numeric vector.
- **longitude** a numeric vector.
tidy_crwFit

tidy-like method for crwFit object

Description

this function mimics the approach taken by broom::tidy to present model output parameters in a tidy, data frame structure.

Usage

```r
 tidy_crwFit(fit)
```

Arguments

- `fit` crwFit object from crawl::crwMLE

Source

Marine Mammal Laboratory, Alaska Fisheries Science Center, National Marine Fisheries Service, NOAA 7600 Sand Point Way NE Seattle, WA 98115

References

Index

*Topic datasets
  beardedSeals, 5
  harborSeal, 23
  northernFurSeal, 25

aic.crw, 3
argosDiag2Cov, 4
as.flat, 4
as.POSIXct, 24
beardedSeals, 5
check_csv, 6
crawl(crawl-package), 2
crawl-package, 2
crWMLE, 3, 6, 9, 10, 12, 16, 18
crwN211, 9
crwPostIS, 10, 14, 16
crwPredict, 4, 12, 21
crwPredictPlot, 13
crwSamplePar, 14
crwSimulator, 11, 14, 16
data.frame, 5, 21
displayPar, 18
expandPred, 19
fillCols, 20
fix_path, 21
flatten, 21
get_restricted_segments, 22
harborSeal, 23
intToPOSIX, 23
mergeTrackStop, 24
northernFurSeal, 5, 6, 22, 25
optim, 7
tidy_crwFit, 26