Package 'CatDyn'

May 15, 2015

Type Package

Title Fishery Stock Assessment by Generalized Depletion Models

LazyLoad yes

LazyData yes

Version 1.1-0

Date 2015-05-15

Author Ruben H. Roa-Ureta

Maintainer Ruben H. Roa-Ureta <ruben.roa.ureta@mail.com>

Depends R (>= 3.0.0)

Imports optimx (>= 2013.8.6), BB

Description Based on fishery Catch Dynamics instead of fish Population Dynamics (hence CatDyn) and using high-frequency or medium-frequency catch in biomass or numbers, fishing nominal effort, and mean fish body weight by time step, from one or two fishing fleets, estimate stock abundance, natural mortality rate, and fishing operational parameters. It includes methods for data organization, plotting standard exploratory and analytical plots, predictions, for 77 types of models of increasing complexity, and 56 likelihood models for the data.

NeedsCompilation no

Repository CRAN

Date/Publication 2015-05-15 12:57:30

R topics documented:

`CatDyn-package` *Fisheries Stock Assessment by Generalized Depletion (Catch Dynamics) Models*

Description

Using high-frequency (daily, weekly) or medium frequency (monthly) catch and effort data CatDyn implements a type of stock assessment model oriented to the operational fishing data. The estimated parameters are in two groups, stock abundance and fishing operation. CatDyn includes 77 versions of the models depending on the number of fleets, the number of perturbations to depletion, whether the stock is resident or in transit, and 56 likelihood models.

Details

Package:	CatDyn
Type:	Package
Version:	1.1-0
Date:	2015-05-15
License:	GPL (>= 2)

Create a data object using raw data and the as.CatDynData() function. Examine the data for regularities and perturbations using the generic plot() function on an object of class CatDynData. Examine the goodness of initial parameter values before statistical inference by using the catdynexp() exploratory prediction function and the plot generic function on an object of class CatDynExp. Fit the model to the data by using the wrapper function CatDynFit(), which in turn will call the optimx() optimizer wrapper of package optimx, with several numerical methods available to be used. Examine the quality of the fit with the plot() function on an object of class CatDynMod created by the CatDynPred() function. Compare different models fit to the data with CatDynSum() and CatDynCor, based on information theoretic, statistical, and numerical criteria.

The process equations in the Catch Dynamics Models in this package are of the form

$$C_t = ke^{-M/2}E_t^a N_t^b$$

$$N_t = N_0 e^{-Mt} - e^{M/2} \sum_{i<t} C_{t-1} e^{-M(t-i-1)} + \sum_j P_j e^{-M(t-j)}$$

where C is catch in numbers, t, i are time step indicators, j is perturbation index (j=1,2,...,100), k is a scaling constant, E is nominal fishing effort, an observed predictor of catch, a is a parameter of effort synergy or saturability, N is abundance, a latent predictor of catch, b is a parameter of hyperstability or hyperdepletion, and M is natural mortality rate per time step. The second summand of the expanded latent predictor is a discount applied to the earlier catches in order to avoid an M-biased estimate of initial abundance. Perturbations to depletion represent fish migrations into the fishing grounds or expansions of the fishing grounds by the fleet(s) resulting in point pulses of abundance. In transit models (limited to one fleet) there are also emmigration events happening at specific time steps for each perturbation. In 2 fleet cases the fleets contribute complementary information about stock abundance, and thus operate additively; any interaction between the fleets is latent and affects the estimated values of fleet dependent parameters, such as k, a, and b.

The observation model can take any of the following forms: a Poisson counts process or a negative binomial counts process for catch recorded in numbers, an additive random normal term added to the continuous catch (in weight) predicted by the process (normal and adjusted profile normal), a multiplicative exponential term acting on the process-predicted catch such as the logarithm of this multiplier distributes normally (lognormal and adjusted profile lognornmal), and Gamma (shape and scale parameterization).

Author(s)

Ruben H. Roa-Ureta <ruben.roa.ureta@mail.com>

References

Roa-Ureta, R. H. 2012. ICES Journal of Marine Science 69(8):1403-1415. Roa-Ureta, R. H. 2015. Fisheries Research (In Press), doi:10.1016/j.fishres.2014.08.014 Roa-Ureta, R. H. et al. 2015. Fisheries Research (In Press), doi:10.1016/j.fishres.2014.12.006

Examples

```
#See examples for CatDynFit()
```

`as.CatDynData` *Data Object for the Estimation of Catch Dynamic Models*

Description

It takes the vectors of catch, effort, and mean body weight from a dataframe and creates an object of class CatDynData. Objects of this class are lists with two components, one for properties of the data such as units and another for the data: catch, effort, mean body weight by fleet, and the catch spike statistic.

Usage

```
as.CatDynData(x, step, fleet.name, coleff, colcat, colmbw,
              unitseff, unitscat, unitsmbw, nmult, season.dates)
```

Arguments

`x`	A dataframe where to find the columns of catch, effort, and mean body weight
`step`	Character. The time step of the dynamics, either "day", "week", or "month".
`fleet.name`	Character. The name of the fleet(s).
`coleff`	Integer. The column(s) in "x" where to find the effort data.
`colcat`	Integer. The column(s) in "x" where to find the catch data.
`colmbw`	Integer. The column(s) in "x" where to find the mean body weight data.
`unitseff`	Character. The unit(s) of effort.
`unitscat`	Character. The unit of catch. Either "ton" (metric tonnes), "kg", or "ind" (individuals).
`unitsmbw`	Character. The unit of body weight. Either "kg", "g", or "ind" (individuals).
`nmult`	Character. The multiplier that scales the catch in numbers. Either "bill" (billions), "mill" (millions), "thou" (thousands), or "ind" (individuals).
`season.dates`	Character vector. A two component character vector with the initial and final dates of the season in the ISO 8601 standard.

Details

The time step determines the rows of 'x'. Make it sure that the number of rows, i.e. the length of the season in time steps, is large enough to estimate all parameters in the model. The simplest model has five parameters, the most complex model has 50 parameters. A rule of thumb would be that the number of time steps be at least three times the number of parameters.

If it is a two fleet system, combine the fleet names, such as c("industrial", "artisanal"), and likewise with coleff, colcat, and colmbw, such as c(5,9) to indicate the columns of catch for the industrial and artisanal fleets respectively. The same applies to units of effort. In a two fleet system, the time step, and the units of catch, mean body weight, and the multiplier must be the same for both fleets.

When the unit of catch and of body weight is "ind", it means that the catch was counted in numbers, not in biomass. In that case the mandatory column of mean body weight should be a column of 1s. The multiplier is the quantity by which the catch in the model shall be raised to be scaled to the actual catch. The idea here is that in many fisheries the daily, weekly, or monthly catch (for example, anchovies, squids) is very large so by setting the multiplier to "bill", "mill", or "thou", the model is working with catches in the orders of tens at most. If the multiplier is set to "ind" then the catch is modeled at the level of the actual catch by time step. This option is useful for sport fisheries, in combination with the 'poisson' or 'negbin' option for distribution.

The 'season.dates' parameter will allow counting the number of steps in integer sequential fashion. If the "time.step" parameter is "day" or "week" the dates may jump one year at most, whereas if the 'time.step' parameter is "month" then 'season.dates' may jump over many years. When the time step is week or months, this parameter needs not be precisely specified; any day within the right week or month will suffice. If you get an error message saying that the number of time steps is not right and that you should consider changing 'season.dates', then just change the dates a few days until you no longer get the error message.

The catch spike statistic is a fleet-specific statistic that is useful to identify the timing of perturbations to depletion; it is defined as

$$S_{f,t} = 10 \times (X_{f,t}/max(X_{f,t}) - E_{f,t}/max(E_{f,t}))$$

where *X* is the observed catch at time step *t* by fleet *f*, and *E* is the observed effort. When this statistic is positive and high then the time step at which this happened is a candidate for a perturbation. In transit fisheries the complementary reasoning is valid: when the statistic is negative and low then the time step at which this happened is a candidate for the timing of one emmigration event.

Value

A list of length 2.

`Properties`	The units for time step, catch, body weight, and the catch numbers multiplier; the names of the fleets and the units of effort, and the start date and end date of the fishing season
`Data`	One dataframe for each fleet with the time step, effort, catch in biomass, mean body weight, catch in the numbers multiplier, and the catch spike statistic

Note

The objects created with as.CatDynData will pass the raw data to plotting and estimating functions.

Author(s)

Ruben H. Roa-Ureta

References

Roa-Ureta, R. H. 2012. ICES J. Mar. Sci. 69:1403-1415. Roa-Ureta, R. H. 2015. Fisheries Research (In Press), doi:10.1016/j.fishres.2014.08.014 Roa-Ureta, R. H. et al. 2015. Fisheries Research (In Press), doi:10.1016/j.fishres.2014.12.006

Examples

```
lgahi <- as.CatDynData(x=lolgahi,
                       step="day",
                       fleet.name="Fk",
                       coleff=2,
                       colcat=1,
                       colmbw=3,
                       unitseff="nboats",
                       unitscat="kg",
                       unitsmbw="kg",
                       nmult="bill",
                       season.dates=c(as.Date("1990-01-31"),
                                      as.Date("1990-05-30")))
```

catdyn — *Class Attribute of Numerically Fit CatDyn Model Objects*

Description

To be used by CatDynPred() to create model results.

Usage

```
catdyn(x, ...)
```

Arguments

`x`	A list object coming from the optimization wrapper CatDynFit().
`...`	Not used.

Value

A class attribute.

Author(s)

Ruben H. Roa-Ureta

Examples

```
#See examples for CatDynFit().
```

CatDynCor — *Correlation Plot Among Parameter Estimates*

Description

It extracts the correlation matrices output by CatDynFit() from several model fits and plots histograms of pairwise correlation coefficients.

Usage

```
CatDynCor(x, ttl, meths, arr)
```

Arguments

`x`	A list of model objects of class 'catdyn' output by CatDynFit().
`ttl`	A character vector to use as title of each histogram.
`meths`	A character vector of numerical methods to extract correlation matrices.
`arr`	A numerical vector of length 2 used to organize the panels containing each histogram. Passed as is to par().

Details

The arguments 'x', 'ttl', and 'meths' must be of the same length.

It might be useful to examine the results of different numerical methods applied to the same model with this function. To do this just repeat the name of the object in 'x' and specify the different numerical methods in 'meths'.

Value

A multiple panel correlation plot.

Note

Those histograms that show more correlations concentrated around zero indicate a better fit.

Author(s)

Ruben H. Roa-Ureta

Examples

```
#See examples for CatDynFit().
```

CatDynData — *Class Attribute of CatDyn Data Object*

Description

To be used by plot.CatDynData() to examine raw data and CatDynFit() to fit models.

Usage

```
CatDynData(x, ...)
```

Arguments

`x`	A list object coming from as.CatDynData().
`...`	Not used.

Value

A class attribute.

Author(s)

Ruben H. Roa-Ureta

Examples

```
#See example for as.CatDynData().
```

CatDynExp *Class Attribute of Exploratory Analysis Object*

Description

To be used by plot.CatDynExp() to explore initial parameter values before fitting models.

Usage

```
CatDynExp(x, ...)
```

Arguments

x	A list object coming from internal functions.
...	Not used.

Value

A class attribute.

Author(s)

Ruben H. Roa-Ureta

Examples

```
#See examples for CatDynFit().
```

catdynexp *Exploratory Evaluation of Initial Values of CatDyn Models*

Description

Using a CatDynData object and initial parameter values, create predictions of model results to examine the goodness of initial parameter values before passing them to the numerical optimizer.

Usage

```
catdynexp(x, p, par, dates, distr)
```

Arguments

x
: An object of class CatDynData.

p
: Integer. The process model type, which quantifies the number of perturbations to depletion. In one-fleet cases 'p' is a scalar integer that can take any value between -20 and 20. In two-fleet cases 'p' is a two-components integer vector that quantifies the number of perturbation events of each fleet. It can take values c(0,0), c(0,1), ..., c(0,5), c(1,1), ..., c(1,5), ..., c(4,5), c(5,5), c(6,6), ..., c(20,20). In transit fisheries, where the stock only passes through the area where it is being fished and leave the fishing grounds at some time step within the season, 'p' should be negative and will take any integer value between -1 and -20.

par
: Numeric. Vector of initial parameter values in the log scale.

dates
: Integer. Vector with the time steps of start of season, perturbations (if any), and end of season. In transit fisheries in addition to the timing of entry of perturbations, the timing of exit for each perturbation shall also be provided, right after the time of entry. For example, p=c(1,4,50,10,60,61) would specify a two-perturbation model which starts at time step 1, has the first perturbation at time step 4, with exit at time step 50, second perturbation at time 10, with exit at time step 60, and season finishing at time step 61.

distr
: Character, either "poisson", "negbin", "normal","apnormal","lognormal","aplnormal", or "gamma", or any pair of these seven (two-fleets systems), corresponding to the likelihood model.

Details

The "negbin" value for the 'distr' parameter corresponds to the negative binomial distribution for counts, as an alternative to the 'poisson' for cases where the assumption of the mean equal to the variance is untenable.

The difference between "normal" and "apnormal", "lognormal" and "aplnormal" is that in the former the dispersion parameters is included in the likelihood function and it is a free parameter to be estimated along with the parameters of the generalized depletion model (and therefore an initial value for the dispersion must be provided) whereas in the latter the dispersion is eliminated by using the adjusted profile likelihood approximation. For the "negbin" and "gamma" distributions the dispersion parameter is always estimated along with the model parameters. In two-fleets models any pair of the seven available likelihood models can be specified although since the units of the catch need to be the same for both fleets then it does not make sense to specify a combination of a distribution for counts and for a continuous variable.

Value

A list of length 2.

Properties
: A list of length 3. 'Units' is a dataframe with the units of time step, catch, body weight, and the numbers multiplier. 'Fleets' is a dataframe with the fleets names and the units of nominal effort for each fleet. 'Dates' is a dataframe with start and end dates of the fishing season in the ISO 8601 format

Model
: A list of length 5. 'Type' is the perturbation type of model. 'Dates' is the timing of perturbations, 'Distr' is the chosen likelihood model, 'Parameters' is

the parameter values being explored, and 'Results', is a dataframe with the time step, and for each fleet, the observed effort, observed catch, predicted catch, and the residuals, plus two more columns, one with the predicted population abundance, and a final one with the predicted population biomass.

Note

The plot.CatDynData() function, acting on objects of class CatDynData, provides a plot of a statistic called the catch spike statistic, that can be useful to determine the 'p' and 'dates' arguments.

Author(s)

Ruben H. Roa-Ureta

Examples

```
#See examples for CatDynFit().
```

CatDynFit *Fit CatDyn Models by Maximum Likelihood*

Description

A wrapper and post-processing tool that checks that the data are passed with proper characteristics, calls optimx() (from package optimx) on any of dozens of possible versions of the generalized depletion models (as internal functions), and then it post-processes optimx() results and join all results in a list of lists.

Usage

```
CatDynFit(x, p, par, dates, distr, method, control = list(),
          hessian = TRUE, itnmax)
```

Arguments

x	A data object of class CatDynData. See as.CatDynData().
p	Integer. The process model type, which quantifies the number of perturbations to depletion. In one-fleet cases 'p' is a scalar integer that can take any value between -20 and 20. In two-fleet cases 'p' is a two-components integer vector that quantifies the number of perturbation events of each fleet. It can take values c(0,0), c(0,1), ..., c(0,5), c(1,1), ..., c(1,5), ..., c(4,5), c(5,5), c(6,6), ..., c(20,20). In transit fisheries, where the stock only passes through the area where it is being fished and leave the fishing grounds at some time step within the season, 'p' should be negative and will take any integer value between -1 and -20.
par	Numeric. Vector of initial parameter values in the log scale.

`dates`	Integer. Vector with the time steps of start of season, perturbations (if any), and end of season. In transit fisheries in addition to the timing of entry of perturbations, the timing of exit for each perturbation shall also be provided, right after the time of entry. For example, p=c(1,4,50,10,60,61) would specify a two-perturbation model which starts at time step 1, has the first perturbation at time step 4, with exit at time step 50, second perturbation at time 10, with exit at time step 60, and season finishing at time step 61.
`distr`	Character, either "poisson", "negbin", "normal","apnormal","lognormal","aplnormal", or "gamma" (one fleet), or any pair of these seven (two fleets) corresponding to the likelihood model.
`method`	Character. Any method accepted by optimx() can be used, but some may return warnings or errors.
`control`	A list of control arguments to be passed to optimx().
`hessian`	Logical. Defaults to TRUE. If set to FALSE all numerical methods tried will fail.
`itnmax`	Numeric. Maximum number of iterations, to pass to optimx().

Details

Much care should be taken in selecting good initial values to pass in the par argument. To accomplish this CatDyn includes the CatDynExp class, and the catdynexp() and the plot.CatDynExp() functions to graphically fine tune the initial values for all model parameters. In multi-annual applications and monthly time step this might be time consuming but it should be carried out to increase the chance that the optimizers will converge to reasonable parameter space.

Initial parameter values must be passed log-transformed by the user. CatDynFit() will backtransform the maximum likelihood estimates and its numerical Hessian matrix without user intervention using the delta method.

Generally, when 'p' is 5 or lower (one fleet) or c(5,5) (two fleets) or lower, the model is applied to one annual season of data and the time step is "day" or "week". Conversely, when 'p' is 6 (one fleet) or c(6,6) (two fleets) or higher the model is applied to multiannual series and the time step is the month, although it is conceivable that for a higly perturbed fishing system higher 'p' values would be applied to single season cases.

The models set up for transit fisheries are single fleet only, so when 'p' is negative, taking any value in the admissible range, its length must be 1.

The discrete Poisson distribution option is recommended for fisheries where the catch is counted in number of fish instead of weight.

The "negbin" value for the 'distr' parameter corresponds to the negative binomial distribution for counts, as an alternative to "poisson" for cases where the assumption of the mean equal to the variance is untenable.

The difference between "normal" and "apnormal", "lognormal" and "aplnormal" is that in the former the dispersion parameters is included in the likelihood function and it is a free parameter to be estimated along with the parameters of the generalized depletion model (and therefore an initial value for the dispersion has to be provided) whereas in the latter the dispersion is eliminated by using the adjusted profile likelihood approximation. For the "negbin" and "gamma" distributions the dispersion parameter is always estimated along with the model parameters.

In two-fleets models any pair of the seven available likelihood models can be specified although since the units of the catch need to be the same for both fleets then it does not make sense to specify a combination of a distribution for the catch in counts ("poisson" and "negbin") and a continuous distribution for the catch in biomass.

Value

A list of length 3.

`Data` A list of length 2. 'Properties' is a list of length 3. 'Units' is a dataframe with the units of time step, catch, body weight, and the numbers multiplier. 'Fleets' is a dataframe with the fleets names and the units of nominal effort for each fleet. 'Dates' is a dataframe with the start and end dates of the season in the ISO 8601 format. 'Data' is a list of length equal to the number of fleets (1 or 2). Each component is a dataframe with the raw data, time step, observed effort, observed catch, observed mean body weight, observed catch in numbers, and the catch spike statistic.

`Initial` A dataframe with named initial values of all free parameters in the model.

`Model` A list with length equal to the number of numerical methods. Each component has the perturbation type model, the dates of events, the chosen distribution for the observation of catch, the integer code describing the success or not of covergence returned by the method, the Karush Kuhn Tucker conditions, hopefully TRUE and TRUE, the value of the Akaike Information Criterion, not comparable between different distributions, the back-transformed (from log) maximum likelihood estimates, the numerical gradients at each maximum likelihood estimate, the standard errors of backtransformed (from log) maximum likelihood estimates, and the correlation matrix of the back-transformed (from log) maximum likelihood estimates.

Note

Complex models may take several hours to converge on a PC. As an example, a two fleet model with 18 perturbations each fleet, p=c(18,18), and the 'aplnormal' likelihood model, totalling 44 parameters to estimate from 216 monthly time steps, coverged successfully in 16 hours on a Windows 7, 64 bit, 3 GHz processor, 8 GB RAM.

Some effort has been made to avoid being kicked out of numerical optimization by just one numerical method that fails, so that optimization continues with other methods, but there may remain some cases when the whole optimization process is aborted by failure in just one method. Try taking out some suspicious methods and optimize again. Experience shows that methods 'spg' and 'CG' are robust for this kind of model so both should be considered as the baseline case for numerical optimization. When using the option of modeling transit fisheries with the Poisson distribution it has been observed that methods 'bobyqa' and 'newuoa' also perform well, so keep an open mind and take advantage of optimx by trying several numerical methods.

Author(s)

Ruben H. Roa-Ureta

References

Roa-Ureta, R. H. 2012. ICES Journal of Marine Science 69(8):1403-1415. Roa-Ureta, R. H. 2015. Fisheries Research (In Press), doi:10.1016/j.fishres.2014.08.014 Roa-Ureta, R. H. et al. 2015. Fisheries Research (In Press), doi:10.1016/j.fishres.2014.12.006

Examples

```
#Falkland Islands one-fleet squid fishery in 1990.
#Create the data object
lgahi <- as.CatDynData(x=lolgahi,
                       step="day",
                       fleet.name="Fk",
                       coleff=2,
                       colcat=1,
                       colmbw=3,
                       unitseff="nboats",
                       unitscat="kg",
                       unitsmbw="kg",
                       nmult="bill",
                       season.dates=c(as.Date("1990-01-31"),as.Date("1990-05-30")))
plot(lgahi,mark=TRUE,offset=c(NA,NA,.75),hem="S")
#
#1) Fit a 1-fleet 1P model with lognormal observation error and the adjusted
#profile approximation to the likelihood to eliminate the dispersion parameter
M         <- 0.011 #1/Time step
N0.ini    <- 3.8 #billions
P1.ini    <- 1.3 #billions
k.ini     <- 5.0e-05 #1/n of boats
alpha.ini <- 1.7 #adimensional
beta.ini  <- 0.6 #adimensional
pars.ini  <- log(c(M,
                   N0.ini,
                   P1.ini,
                   k.ini,
                   alpha.ini,
                   beta.ini))
#Dates
P1    <- 70 #Selected by visual inspection of standard plot
dates <- c(head(lgahi$Data$Fk$time.step,1),
           P1,
           tail(lgahi$Data$Fk$time.step,1))
lgahi.apln.1P.ini <- catdynexp(x=lgahi,
                               p=1,
                               par=pars.ini,
                               dates=dates,
                               distr="aplnormal")
plot(x=lgahi.apln.1P.ini,
     leg.pos="topright",
     Biom.tstep="mid",
     Biom.xpos=0.4,
     Biom.ypos=0)
#fit
```

```
(lgahi.apln.1P.fit <- CatDynFit(x=lgahi,
                                p=1,
                                par=pars.ini,
                                dates=dates,
                                distr="aplnormal",
                                method="spg",
                                itnmax=10))
#examine results
lgahi.apln.1P.pred.spg <- CatDynPred(lgahi.apln.1P.fit,"spg")
plot(x=lgahi.apln.1P.pred.spg,
     leg.pos="topright",
     Biom.tstep=7,
     Biom.xpos=0.18,
     Biom.ypos=0.1,
     AIC.xpos=0.18,
     AIC.ypos=0.2)
#
#2) Fit a 1-fleet 2P model with lognormal observation error and full exact
#likelihood including the dispersion parameter
M          <- 0.011 #1/Time step
N0.ini     <- 3.8 #billions
P1.ini     <- 1.3 #billions
P2.ini     <- 0.5 #billions
k.ini      <- 4.0e-05 #1/n of boats
alpha.ini <- 1.7 #adimensional
beta.ini  <- 0.6 #adimensional
#Note how to get reasonable initial value for dispersion parameter
psi.ini   <- 0.33*sd(log(lgahi$Data$Fk$obscat.bill))^2
pars.ini  <- log(c(M,
                   N0.ini,
                   P1.ini,
                   P2.ini,
                   k.ini,
                   alpha.ini,
                   beta.ini,
                   psi.ini))
#Dates
P1    <- 70  #Selected by visual inspection of standard plot
P2    <- 135 #Selected by visual inspection of standard plot
dates <- c(head(lgahi$Data$Fk$time.step,1),
           P1,
           P2,
           tail(lgahi$Data$Fk$time.step,1))
lgahi.ln.2P.ini <- catdynexp(x=lgahi,
                             p=2,
                             par=pars.ini,
                             dates=dates,
                             distr="lognormal")
plot(x=lgahi.ln.2P.ini,
     leg.pos="topright",
     Biom.tstep="ini",
     Biom.xpos=0.4,
     Biom.ypos=0)
```

```
#fit lognormal
(lgahi.ln.2P.fit <- CatDynFit(x=lgahi,
                              p=2,
                              par=pars.ini,
                              dates=dates,
                              distr="lognormal",
                              method="spg",
                              itnmax=10))
#examine results
lgahi.ln.2P.pred.spg <- CatDynPred(lgahi.ln.2P.fit,"spg")
plot(x=lgahi.ln.2P.pred.spg,
     leg.pos="topright",
     Biom.tstep=7,
     Biom.xpos=0.18,
     Biom.ypos=0.1,
     AIC.xpos=0.18,
     AIC.ypos=0.2)
#
#3) Fit a 1-fleet 2P model with gamma observation error and full, exact
#likelihod including the dispersion parameter
M         <- 0.011 #1/Time step
N0.ini    <- 3.8 #billions
P1.ini    <- 1.3 #billions
P2.ini    <- 0.5 #billions
k.ini     <- 4.0e-05 #1/n of boats
alpha.ini <- 1.7 #adimensional
psi.ini   <- 0.08 #From psi max. lik. estimate in 2P lognormal model
pars.ini  <- log(c(M,
                   N0.ini,
                   P1.ini,
                   P2.ini,
                   k.ini,
                   alpha.ini,
                   beta.ini,
                   psi.ini))
#Dates
P1    <- 70  #Selected by visual inspection of standard plot
P2    <- 135 #Selected by visual inspection of standard plot
dates <- c(head(lgahi$Data$Fk$time.step,1),
           P1,
           P2,
           tail(lgahi$Data$Fk$time.step,1))
lgahi.g.2P.ini <- catdynexp(x=lgahi,
                            p=2,
                            par=pars.ini,
                            dates=dates,
                            distr="gamma")
plot(x=lgahi.g.2P.ini,
     leg.pos="topright",
     Biom.tstep="ini",
     Biom.xpos=0.4,
     Biom.ypos=0)
#fit gamma
```

```
(lgahi.g.2P.fit <- CatDynFit(x=lgahi,
                             p=2,
                             par=pars.ini,
                             dates=dates,
                             distr="gamma",
                             method="spg",
                             itnmax=10))
#examine results
lgahi.g.2P.pred.spg <- CatDynPred(lgahi.g.2P.fit,"spg")
plot(x=lgahi.g.2P.pred.spg,
     leg.pos="topright",
     Biom.tstep=7,
     Biom.xpos=0.18,
     Biom.ypos=0.1,
     AIC.xpos=0.18,
     AIC.ypos=0.2)
#
#Summary table for model selection
(lgahi.sum <- CatDynSum(x=list(lgahi.apln.1P.fit,
                               lgahi.ln.2P.fit,
                               lgahi.g.2P.fit),
                         season=1990,
                         meths=c("spg","spg","spg")))
#Plot for correlations among parameter estimates
#
CatDynCor(x=list(lgahi.apln.1P.fit,
                 lgahi.ln.2P.fit,
                 lgahi.g.2P.fit),
          ttl=c("Adjusted Profile Lognormal 1P","Lognormal 2P","Gamma 2P"),
          meths=c("spg","spg","spg"),
          arr=c(2,2))
```

CatDynMod *Class Attribute of Model Output Object*

Description

To be used by plot.CatDynMod() to graphically examine model results.

Usage

```
CatDynMod(x, ...)
```

Arguments

x	A list object coming from CatDyn().
...	Not used.

Value

A class attribute

Author(s)

Ruben H. Roa-Ureta

Examples

```
#See examples for CatDynFit().
```

`CatDynPred` *Predictions from a Generalized Depletion (Catch Dynamics) Model*

Description

Calculates the predicted catch, residuals, by fleet, and population biomass of a fish stock using parameters estimated for a Catch Dynamic Model.

Usage

```
CatDynPred(x, method)
```

Arguments

`x`	An object of class catdyn.
`method`	Character. The particular numerical method from which estimates have to be drawn.

Details

This function is very similar to catdynexp() but instead of using arbitrary parameter values given by the user, it takes the maximum likelihood estimates produced by the optimizer.

Value

A list of length 2.

`Properties`	A list of length 3. 'Units' is a dataframe with the units of time step, catch, body weight, and the numbers multiplier. 'Fleets' is a dataframe with the fleets names and the units of nominal effort for each fleet. 'Dates' is a dataframe with start and end dates of the fishing season in the ISO 8601 format.
`Model`	A list of length 5. 'Type' is the perturbation type of model. 'Dates' is the timing of perturbations, 'Distr' is the chosen likelihood model, 'Parameters' are the maximum likelihood estimates from CatDynFit() plus the observation variance (except for adjusted profile likelihood models), and 'Results' is a dataframe with the time step, and for each fleet, the observed effort, observed catch, predicted catch, and the residuals, plus two more columns, one with the predicted population abundance, and a final one with the predicted population biomass.

Author(s)

Ruben H. Roa-Ureta

References

Roa-Ureta, R. H. 2012. ICES Journal of Marine Science 69(8):1403-1415. Roa-Ureta, R. H. 2015. Fisheries Research (In Press), doi:10.1016/j.fishres.2014.08.014 Roa-Ureta, R. H. et al. 2015. Fisheries Research (In Press), doi:10.1016/j.fishres.2014.12.006

Examples

```
#See examples for CatDynFit().
```

CatDynSum — *Summary Table for Generalized Depletion (Catch Dynamics) Model Comparison*

Description

Summarize information theoretic, numerical and statistical results of several models fit to the data to select a best working model.

Usage

```
CatDynSum(x, season, meths)
```

Arguments

`x`	A list of model objects of class 'catdyn' output by CatDynFit().
`season`	Character, an identifier for the set of models.
`meths`	A character vector of numerical methods to extract from each element of the 'x' list.

Details

The arguments 'x' and 'meths' must be of the same length.

It might be useful to examine the results of different numerical methods applied to the same model with this function. To do this just repeat the name of the object in 'x' and specify the different numerical methods in 'meths'.

Value

A data.frame with an extraction of comparative information from several models fitted to the data.

Author(s)

Ruben H. Roa-Ureta

Examples

```
#See examples for CatDynFit().
```

`deltamethod` *First-order Taylor Series Expansion of Functions of Random Variables*

Description

The delta method for approximating the standard error of a transformation $g(X)$ of a random variable $X = (x1, x2, ...)$, given estimates of the mean and covariance matrix of X.

Usage

```
deltamethod(g, mean, cov, ses = TRUE)
```

Arguments

`g`	A formula representing the transformation. The variables must be labelled $x1$, $x2$, ... For example, ~ 1 / ($x1$ + $x2$). If the transformation returns a vector, then a list of formulae representing ($g1$, $g2$, ...) can be provided, for example list(~ $x1$ + $x2$, ~ $x1$ / ($x1$ + $x2$)).
`mean`	The estimated mean of X.
`cov`	The estimated covariance matrix of X.
`ses`	If TRUE, then the standard errors of $g1(X)$, $g2(X)$, ... are returned. Otherwise the covariance matrix of $g(X)$ is returned.

Details

This function was copied from package msm. It is used in CatDyn to backtransform from the logarithm because CatDyn parameters are all estimated in the log scale to improve numerical performance. For more details see the help pages for function deltamethod of package msm.

Value

A vector containing the standard errors of $g1(X)$, $g2(X)$, ... or a matrix containing the covariance of $g(X)$.

Author(s)

C. H. Jackson <chris.jackson@mrc-bsu.cam.ac.uk>

Examples

```
#See the examples in package msm.
```

`gayhake` *Industrial and Artisanal Catch and Effort Data from the Chilean Hake Fishery*

Description

Exhaustive recorded operational activity of the *Merluccius gayi* two fleet fishery off Central Chile in 2006.

Usage

```
data("gayhake")
```

Format

A data frame with 53 observations on the following 7 variables.

`TimeStep` a numeric vector

`Catch.Ind.kg` a numeric vector

`NShips.Ind` a numeric vector

`Body.Ind.g` a numeric vector

`Catch.Art.kg` a numeric vector

`NShips.Art` a numeric vector

`Body.Art.g` a numeric vector

Source

Under-Secretariat of Fishing, Chilean government.

Examples

```
data(gayhake)
```

`lolgahi` *Industrial Trawling Data from the Squid Fishery of the Falkland Islands*

Description

Exhaustive recorded operational activity of the *Loligo gahi* fleet off the Falkland Islands during the summer season of 1990.

Usage

```
data(lolgahi)
```

Format

A data frame with 120 observations on the following 6 variables.

`obscat` a numeric vector

`obseff` a numeric vector

`obsmbm` a numeric vector

Source

http://fis.com/falklandfish/

References

Roa-Ureta, R. H. 2012. ICES Journal of Marine Science 69(8):1403-1415.

Examples

```
data(lolgahi)
```

---

plot.CatDynData *Exploratory Analysis of Catch and Effort Fisheries Data*

Description

Allows examining the relation between catch and effort, the marginal distributions of catch and effort, and the time series of catch, effort, the catch spike statistic, and mean body mass in the catch.

Usage

```
## S3 method for class 'CatDynData'
plot(x, mark, offset, hem, ...)
```

Arguments

`x`	An object of class CatDynData.
`mark`	Logical. If TRUE then the time step is posted on top of each point of the time series of catch, effort, and the catch spike statistic.
`offset`	Numeric. A vector of length 3 that positions the mark above a given distance over the point.
`hem`	Character. Either "N" (northern hemisphere) or "S" (southern hemisphere).
`...`	Further arguments to be passed to plot(), hist().

Details

Use NA to cancel the mark over the points of any of the three time series that can be marked. In the case of two-fleet models, the plot will display the data for the first fleet, then the user needs to hit Enter to display the data for the second fleet.

Value

A seven panel plot.

Author(s)

Ruben H. Roa-Ureta

Examples

```
#See examples for CatDynFit().
```

`plot.CatDynExp` *Exploratory Modeling of Fisheries with Catch Dynamic Models*

Description

Evaluate and refine the goodness of initial parameter values before fitting catch dynamic models to data.

Usage

```
## S3 method for class 'CatDynExp'
plot(x, leg.pos, Biom.tstep, Biom.xpos, Biom.ypos, ...)
```

Arguments

`x`	An object of class CatDynExp
`leg.pos`	The position of the legend in the first panel. Passed to legend().
`Biom.tstep`	Integer. The number of time steps over which to average the population biomass counting from the end of the season backwards.
`Biom.xpos`	Numeric. The position of the biomass value on the x-axis of the first panel of the plot in relative units.
`Biom.ypos`	Numeric. The position of the biomass value on the y-axis of the first panel of the plot in relative units.
`...`	Further arguments to pass to plot(), hist().

Details

If the average population biomass over the whole season is to be posted then an integer equal to the number of time steps in the season shall be entered for the 'Biom.tstep' argument.

Value

A four panel plot of data, model predictions, and residual analysis.

Note

The target symbols on the bottom of the tope left panel are the timings of any perturbations set by the user. In transit fisheries, entry target symbols are in red and exit target symbols are in blue.

Author(s)

Ruben H. Roa-Ureta

Examples

```
#See examples for CatDynFit().
```

---

plot.CatDynMod *Examination of Results from Fitting Catch Dynamic Models*

Description

After model fit and prediction, examine model results on a graphical display.

Usage

```
## S3 method for class 'CatDynMod'
plot(x, leg.pos, Biom.tstep, Biom.xpos, Biom.ypos, AIC.xpos,
                          AIC.ypos, ...)
```

Arguments

`x`	An object of class CatDynMod.
`leg.pos`	The position of the legend in the top left panel. Passed to legend().
`Biom.tstep`	Integer. The number of time steps over which to average the population biomass counting from the end of the season backwards.
`Biom.xpos`	Numeric. The position of the biomass estimate on the x-axis of the first panel of the plot in relative units.
`Biom.ypos`	Numeric. The position of the biomass estimate on the y-axis of the first panel of the plot in relative units.
`AIC.xpos`	Numeric. The position of the Akaike Information Criterion on the x-axis of the first panel of the plot in relative units.
`AIC.ypos`	Numeric. The position of the Akaike Information Criterion on the y-axis of the first panel of the plot in relative units.
`...`	Further arguments to pass to plot(), hist().

Details

If the average population biomass over the whole season is to be posted then an integer equal to the number of time steps in the season shall be entered for the 'Biom.tstep' argument.

Value

A four panel plot of data, model predictions, and residual analysis.

Note

The target symbols on the bottom of the tope left panel are the timings of any perturbations set by the user.

In transit fisheries, entry target symbols are in red and exit target symbols are in blue.

Author(s)

Ruben H. Roa-Ureta

References

Roa-Ureta, R. H. 2012. ICES Journal of Marine Science 69(8):1403-1415.

Examples

```
#See examples for CatDynFit().
```

---

twelver *Sport Fishing for Eel Juveniles in Taiwan*

Description

Exhaustive recorded operational activity of the *Anguilla japonica* sport fishery in Taiwan during the 1983-1984 season.

Usage

```
data("twelver")
```

Format

A data frame with 110 observations on the following 4 variables.

`Date` a Date

`Effort.Hour.Soak` a numeric vector

`Catch.Ind` a numeric vector

`Mbw` a numeric vector

Source

Dpt. of Zoology, National Taiwan University

Examples

```
data(twelver)
```

Index