Package ‘SAFR’

November 18, 2014

Type Package

Title Survival Analysis for Fisheries Research demonstrates applications of survival analysis (Cox, 1984) to analyze catch at age data in fisheries research

Version 0.1

Date 2014-10-27

Author Marco Kienzle

Maintainer Marco Kienzle <Marco.Kienzle@gmail.com>

Description This package implement several likelihood function that allow the user to estimate mortality rates (both natural and fishing) as well as selectivity and recruitment from catch at age data from fisheries.

R topics documented:

`Caaa2Coaa`	*Convert catch-at-age to cohort-at-age*

Description

Convert a matrix of year x age into a matrix of cohort x age

Usage

```
Caaa2Coaa(mat)
```

Arguments

`mat`	a matrix of catch at age

References

NONE

`EstimateRecruitment`	*Estimate recruitment using total mortality (Z) and catch at age from a single cohort*

Description

This function provide an estimate of recruitment using total mortality (Z) and catch at age. The idea is that each catch at age figure provides an estimate of recruitment which first and second moment are calculated

Usage

```
EstimateRecruitment(Z)
```

Arguments

`Z`	a numerica scalar >= 0

References

no reference

Examples

```
# Suppose age varies between 0 and 10
age = seq(0,10)

# Suppose you have a M=0.105, F and N0 are arbitrary (randomly generated)
M <- 0.105
F <- runif(1, min = 0.1, max = 3)
print(paste("Simulated Z is", round(M+F,3)))

# Generate a random recruitment
N0 <- runif(1, min = 1e3, max = 1e4)
print(paste("Simulated recruitment", round(N0)))

# Calculate number at age using a simple exponential model ( see Quinn and Deriso, 1999)
nb.at.age <- cbind(age, N0 * exp(-(M+ F)) ^ age)

# Calculate the total number of individual dying at age
total.death <- N0 * (exp(-(M+F) * age) - exp(-(M+F) * (age+1)))

# And the fraction dying from fishing
catch <- F/(M+F) * total.death

# Estimate Z
#best.Z.estimate <- EstimateZ(catch)
#best.Rec.estimate <- EstimateRecruitment(Z=best.Z.estimate$par)
```

GenerateData2 *Convert catch-at-age to cohort-at-age*

Description

Convert a matrix of year x age into a matrix of cohort x age

Usage

```
GenerateData2(max.age = 10, nb.of.cohort = 20, ...)
```

Arguments

max.age	max age
nb.of.cohort	number of cohorts
...	other parameters

References

NONE

`llfunc1` *log-likelihood function of catch at age and total mortality (Z) written to Z*

Description

This function calculate the log-likelihood of the survival model assuming constant mortality rate (Z) given catch at age.

NOTE that the optimization does not constrains Z > 0 because we noticed problems of convergence of the example below on 32-bit systems when using method = "L-BFGS-B".

Usage

```
llfunc1(Z, catch)
```

Arguments

`Z`	a positive or null scalar giving the constant mortality rate (units 1/year)
`catch`	a vector of number of individual caught at age. The first value represent a number of animal between age 0 and 1, the second between 1 and 2, etc...

References

Cox (1984) - Analysis of survival data

Examples

```
# Suppose age varies between 0 and 10
age <- seq(0,10)

# Suppose you have a M=0.105, F and N0 are arbitrary (randomly generated)
M <- 0.105
F <- runif(1, min = 0.1, max = 3)
print(paste("Simulated Z is", round(M+F,3)))

# Generate a random recruitment
N0 <- runif(1, min = 1e3, max = 1e4)
print(paste("Simulated recruitment", round(N0)))

# Calculate number at age using a simple exponential model ( see Quinn and Deriso, 1999)
nb.at.age <- cbind(age, N0 * exp(-(M+ F)) ^ age)

# Calculate the total number of individual dying at age
total.death <- N0 * (exp(-(M+F) * age) - exp(-(M+F) * (age+1)))

# And the fraction dying from fishing
catch <- F/(M+F) * total.death

# Estimate Z
```

```
result <- optim(par = c(0.1), fn = llfunc1, catch = catch, method = c("L-BFGS-B"),
      lower = c(1e-2), upper = c(3), hessian = TRUE)
print(paste("Estimated Z is", round(result$par,3), "+-", round(sqrt(diag(solve(result$hessian))),3)))
```

`llfunc2` *log-likelihood function of catch at age to estimate catchability and natural mortality*

Description

This function calculate the log-likelihood of the survival model assuming fishing mortality is a linear function of effort and natural mortality is constant.

NOTE that the optimization does not constrains both q and M to be > 0 because we noticed problems of convergence of the example below on 32-bit systems when using method = "L-BFGS-B".

Usage

```
llfunc2(par, catch, effort, catchability.scaling.factor)
```

Arguments

`par`	a vector of two parameters: catchability and natural mortality
`catch`	a vector of catch
`effort`	a vector of effort
`catchability.scaling.factor`	a factor to scale the parameters

References

Cox (1984) - Analysis of survival data

Examples

```
# Suppose age varies between 0 and 10
age <- seq(0,10)

# Generate a random natural mortality
M <- runif(1, min = 1e-2, max = 0.3)

effort <- runif(length(age)-1, min = 1e3, max = 2e3)
catchability <- runif(1, min = 1/3e3, max = 1/2e3)

# Catchability scaling factor
csf <- 1e-4
F <-  catchability * effort
```

```
print(paste("Simulated q is", round(catchability / csf,3), as.character(csf)))
print(paste("Simulated M is ", round(M,3)))

N0 <- runif(1, min = 4e3, max = 1e4)
print(paste("Simulated recruitment is", round(N0)))

# Calculate number at age using a simple exponential model ( see Quinn and Deriso, 1999)
nb.at.age <- cbind(age, N0 * exp(-c(0, cumsum(M + F))))

# Calculate the total number of individual dying at age
total.death <- N0 * (exp(-c(0,cumsum(M+F)[-length(effort)])) - exp(-cumsum(M+F)))

# Number of fish dying from fishing is a fraction of total mortality
catch <- F/(M+F) * total.death

# Estimate q and M
best.qM.est <- optim(par = c(10,1), fn = llfunc2, catch = catch, effort
= effort, catchability.scaling.factor = csf, hessian = TRUE)

errors <- sqrt(diag(solve(best.qM.est$hessian)))

print(" ##### ")
print(paste("Estimated catchability is", round(best.qM.est$par[1],3), "+-", round(errors[1],3), as.character(cs
print(paste("Estimated M is", round(best.qM.est$par[2],3), "+-", round(errors[2],3)))
```

`llfunc3` *log-likelihood function of catch at age matrix to estimate catchability, selectivity and natural mortality*

Description

This function calculate the log-likelihood of the survival model assuming fishing mortality the outer product of catchability times effort and selectivity and natural mortality is constant.

Usage

```
llfunc3(par, catch, effort, selectivity.at.age, catchability.scaling.factor)
```

Arguments

`par`	a vector of two parameters: catchability and natural mortality
`catch`	a vector of catch
`effort`	a vector of effort
`selectivity.at.age`	a vector of selectivity at age bound between 0 and 1
`catchability.scaling.factor`	a factor to scale the parameters

References

NONE

Examples

```
# Simulate data
set.seed(3)
max.age <- 9
sim <- GenerateData2(max.age = max.age, nb.of.cohort = 30) # Generate catch using gear selectivity

# Estimate assuming you know selectivity
result <- optim(par = c(0.2,1), fn = llfunc3, catch = sim$catch, effort = sim$effort, catchability.scaling.factor
 errors <- sqrt(diag(solve(result$hessian)))

print(paste("Estimated catchability is", round(result$par[1],3), "+-", round(errors[1],3), " x 10^-4"))
print(paste("Estimated M is", round(result$par[2],3), "+-", round(errors[2],3)))
```

`llfunc4` *log-likelihood function of catch at age matrix to estimate catchability, selectivity and natural mortality*

Description

This function calculate the log-likelihood of the survival model assuming fishing mortality the outer product of catchability times effort and selectivity and natural mortality is constant.

Usage

```
llfunc4(par, catch, effort, catchability.scaling.factor)
```

Arguments

`par`	a vector of two parameters: catchability and natural mortality
`catch`	a vector of catch
`effort`	a vector of effort
`catchability.scaling.factor`	a factor to scale the parameters

References

NONE

Examples

```
# Simulate data
set.seed(3)
max.age <- 9
sim <- GenerateData2(max.age = max.age, nb.of.cohort = 30) # Generate catch using gear selectivity

# Estimate parameters
result2 <- optim(par = c(0.2,1, c(rep(1e-12,5), rep(1, max.age-5))), fn = llfunc4, catch = sim$catch, effort = sin
      lower = c(5e-2,5e-2, rep(1e-12, max.age)), upper = c(10,0.5, rep(1, max.age)), hessian = TRUE, control = lis
errors2 <- sqrt(abs(diag(solve(result2$hessian))))

print(cbind("Estimate" = result2$par, "Error" = errors2))
```

---

llfunc5 *log-likelihood function of catch at age matrix to estimate catchability, selectivity and natural mortality*

Description

This function calculate the log-likelihood of the survival model assuming fishing mortality the outer product of catchability times effort and selectivity and natural mortality is constant.

Usage

```
llfunc5(par, catch, effort, catchability.scaling.factor)
```

Arguments

`par` a vector of two parameters: catchability and natural mortality

`catch` a vector of catch

`effort` a vector of effort

`catchability.scaling.factor`
a factor to scale the parameters

References

NONE

Examples

```
# Simulate data
set.seed(3)
max.age <- 9
sim <- GenerateData2(max.age = max.age, nb.of.cohort = 30) # Generate catch using gear selectivity

# Estimate parameters, fixing selectivity for the last 2 age-groups to 1
result3 <- optim(par = c(0.2,1, rep(1e-12,max.age-2)), fn = llfunc5, catch = sim$catch, effort = sim$effort, catc
      lower = c(5e-2,5e-2, rep(1e-12, max.age-2)), upper = c(10,0.5, rep(1, max.age-2)), hessian = TRUE, control =
```

```
errors3 <- sqrt(abs(diag(solve(result3$hessian))))

print(cbind("Estimate" = result3$par, "Error" = errors3))
```

`my.cumsum` *a cumsum that omit NA*

Description

A function to sum cumulatively rows of a matrix, replacing NA by 0.

Usage

```
my.cumsum(mat)
```

Arguments

`mat`	a matrix

References

NONE

`total.over.lines` *calculate cumulative sum over the lines of a matrix, ignoring NAs*

Description

Calculate the cumulative sum

Usage

```
total.over.lines(mat)
```

Arguments

`mat`	a matrix

References

NONE

`which.cohort` *A function that counts and numbers cohorts given a catch at age matrix*

Description

Number the cohort in a matrix

Usage

```
which.cohort(mat)
```

Arguments

`mat`	a matrix of data

Examples

```
mat <- matrix(NA, nrow = 5, ncol = 7)
which.cohort(mat)
```

Index