Fitting stock-recruitment relationships with FLSR

October 2014

The stock and recruitment problem may be considered as the search for the relationship between parental stock size

A fundamental problem in the quantitative assessment of fisheries is the relationship between the reproductive potential of the stock, commonly approximated by the Spawning Stock Biomass (SSB) or Total Egg production (TEP), and the subsequent recruitment in numbers, i.e. the year class strengt. The stock-recruitment relationship is an essential element for assessing the resilience of a fish population, and forecasting the likely effect and time of responses to management measures.

Although modern statistical catch-at-age assessment methods generally integrate the estimation of the parameters of this relationship together with all others, VPA-based models require fitting the stock-recruitment relationship based on the model estimates of abundance. Also, the relationship is employed in most forecasting algorithms, such as those offered by the `FLash` package.

Stock-recruitment relationships in FLR are handled by the `FLSR` class, which contains slots for both inputs (`rec` and `ssb`), outputs (`fitted` and `residuals`), model specification (`model`, `lkhd`, `initial`), and parameter outputs (`params`, `vcov`, `logLik`).

We will now look at how objects of this class can be created, how to fit stock-recruitment models, what models are already available in `FLCore`, and the methods for model fitting, inspection of results and diagnostics.

The FLSR class

```
library(FLCore)
```

```
## Loading required package: grid
## Loading required package: lattice
```

```
## Loading required package: MASS
## FLCore (Version 2.5.20141011, packaged: 2014-10-24 11:32:38 UTC)
##
## Attaching package: 'FLCore'
##
## The following objects are masked from 'package:base':
##
##     cbind, rbind
```

```
library(ggplotFL)
```

```
## Loading required package: ggplot2
##
## Attaching package: 'ggplot2'
##
## The following object is masked from 'package:FLCore':
##
##     %+%
##
## Loading required package: gridExtra
## Loading required package: reshape2
```

Class structure

- name (`character`)
- desc (`character`)
- range (`numeric`)
- rec (`FLQuant`)
- ssb (`FLQuant`)
- covar (`FLQuants`)
- fitted (`FLQuant`)
- residuals (`FLQuant`)
- model (`formula`)
- lkhd (`function`)
- gr (`function`)
- initial (`function`)
- logerror (`logical`)

- distribution (`factor`)
- params (`FLPar`)
- logLik (`logLik`)
- vcov (`array`)
- hessian (`array`)
- details (`list`)

```
data(nsher)
```

Creating `FLSR` objects

The class has a creator method of the same name. The `FLSR()` method will take as inputs the objects to be assigned into the individual slots. For example, to create a new `FLSR` object with existing `FLQuant` object for the two time series, we assign them to their respective slots in the method call:

```
ssb <- FLQuant(100 + rnorm(31, 0, 12), quant="age",
    dimnames=list(year=1900:1930), units="t")
rec <- FLQuant(20 + c(0, cumsum(sample(rnorm(30, 0, 2)))),
    dimnames=list(year=1901:1931, age="1"), units="1000")

exs <- FLSR(rec=rec, ssb=ssb, name="randFLSR")
```

Please note how the lag between spawning and recruitment is reflected in the `dimnames` of the two `FLQuant` slots: year dimnames for `ssb` go from 1900 to 1930 while the `rec` series starts in 1901 until 1931. Age at recruitment is 1 in this case, as reflected by the `rec` dimnames for the first dimension:

```
# CHECK recruitment lag as age dimension name
dimnames(rec(exs))$age
```

```
## [1] "1"
```

Although many methods won't use this information, and will simply assume you have placed both time series in the object with the correct lag, it is useful to label the object dimensions correctly to display results, and to later recall the precise structure of the data.

Specifying the stock-recruitment model

The call to the `FLSR()` creator allows specifying the stock-recruitment model to be used. As explained above, two slots contain the information necessary for fitting through maximum likelihood, `model` and `lkhd`, while a third, `initial`, simplifies the call to the optimizer by generating initial values based on the input data, but they can also be provided in the call to the minimization routine.

`FLCore` already contains a number of commonly-used stock-recruitment relationships in various formulations (see a full list below at Stock-recruitment models). For each of them, a function has been defined that returns a list containing those three elements, for example

```
# INSPECT bevholt function
bevholt()
```

```
## $logl
## function (a, b, rec, ssb)
## loglAR1(log(rec), log(a * ssb/(b + ssb)))
## <environment: 0x4eaebc8>
##
## $model
## rec ~ a * ssb/(b + ssb)
## <environment: 0x4eaebc8>
##
## $initial
## function (rec, ssb)
## {
##     a <- max(quantile(c(rec), 0.75, na.rm = TRUE))
##     b <- max(quantile(c(rec)/c(ssb), 0.9, na.rm = TRUE))
##     return(FLPar(a = a, b = a/b))
## }
## <environment: 0x4eaebc8>
## attr(,"lower")
## [1] -Inf -Inf
## attr(,"upper")
## [1] Inf Inf
```

and the `model<-` assigment method, which usually would only modify the content of the `model` slot, a formula, can also place those three elements in the right slots, giving us the necessary elements for fitting the chosen model.

```
# REPLACE model (plus logl and initial)
model(nsher) <- bevholt()
```

Modifying the model in an object will also modify other slots. Those that keep results (`fitted`, `residuals`, `params`, `vcov`, `hessian`, ...) will have their structure adapted to the parameter set defined by the new model and their contents turn into `NA`s.

```
# INSPECT object after model change
summary(nsher)
```

```
## An object of class "FLSR"
##
## Name: Autumn spawning herring in IV, V  3/4/2005 14:46
## Description: 'rec' and 'ssb' slots obtained from a 'FLStock' object
## Range:
##
## Quant: age
##
## rec           : [ 1 45 1 1 1 1 ], units =  10^3
## ssb           : [ 1 45 1 1 1 1 ], units =  t*10^3
## residuals     : [ 1 45 1 1 1 1 ], units =  NA
## fitted        : [ 1 45 1 1 1 1 ], units =  10^3
##
## Model:   rec ~ a * ssb/(b + ssb)
## <environment: 0x5331db8>
## Parameters:
##     params
## iter  a  b
##    1 NA NA
##
## Log-likelihood:  NA(NA)
## Variance-covariance:       params
## params  a  b
##      a NA NA
##      b NA NA
```

```
params(nsher)
```

```
## An object of class "FLPar"
## params
##  a  b
## NA NA
## units:  NA NA
```

Converting from other classes

FLStock

A common source of recruitment and stock biomass estimates for stock-recruitment analysis is the result of an stock assessment run, stored in an object of class `FLStock`. To simplify creating an `FLSR` object from those estimates, a conversion method exists, `as.FLSR`, to be called as follows:

```
# LOAD FLStock object
data(ple4)

# CONVERT to FLSR
psr4 <- as.FLSR(ple4)

summary(psr4)
```

```
## An object of class "FLSR"
##
## Name: Plaice in IV
## Description: 'rec' and 'ssb' slots obtained from a 'FLStock' object
## Range:    min    minyear max maxyear
##  1   1958    1   2008
## Quant: age
##
## rec           : [ 1 51 1 1 1 1 ], units =  10^3
## ssb           : [ 1 51 1 1 1 1 ], units =  kg
## residuals     : [ 1 51 1 1 1 1 ], units =  NA
## fitted        : [ 1 51 1 1 1 1 ], units =  10^3
##
## Model:   list()
## <environment: 0x3f951e8>
## Parameters:
##
##    1
##
## Log-likelihood:  NA(NA)
## Variance-covariance: <0 x 0 matrix>
```

which will use the `FLQuant` objects extracted by calling the `rec` and `ssb` methods on the input object to fill those slots. Other `FLQuant` slots are then sized as required. Note that `rec(FLStock)` will return by default the abundances at the first available age.

The selected model can also be specified at the convertion stage, or later added and modified using the `model<-` method

```
# ASSIGN model
psr4 <- as.FLSR(ple4, model=ricker)

model(psr4) <- ricker
```

Stock-recruitment models

- Beverton & Holt (`bevholt()`)
- Ricker (`ricker()`)
- Segmented regression (`segres()`)
- Cushing (`cushing()`)
- Shepherd (`shepherd()`)

Model fitting

Maximum likelihood estimation with `fmle`

Fixing parameters

Using covariates

Assessing the fit

```
logLik(nsher)

## 'log Lik.' NA (df=2)

AIC(nsher)

## [1] NA

BIC(nsher)

## [1] NA
```

Likelihood profiling

```
profile(nsher)

## Warning in .local(fitted, ...): model has not been fitted: initial values
## are used for profile range
```

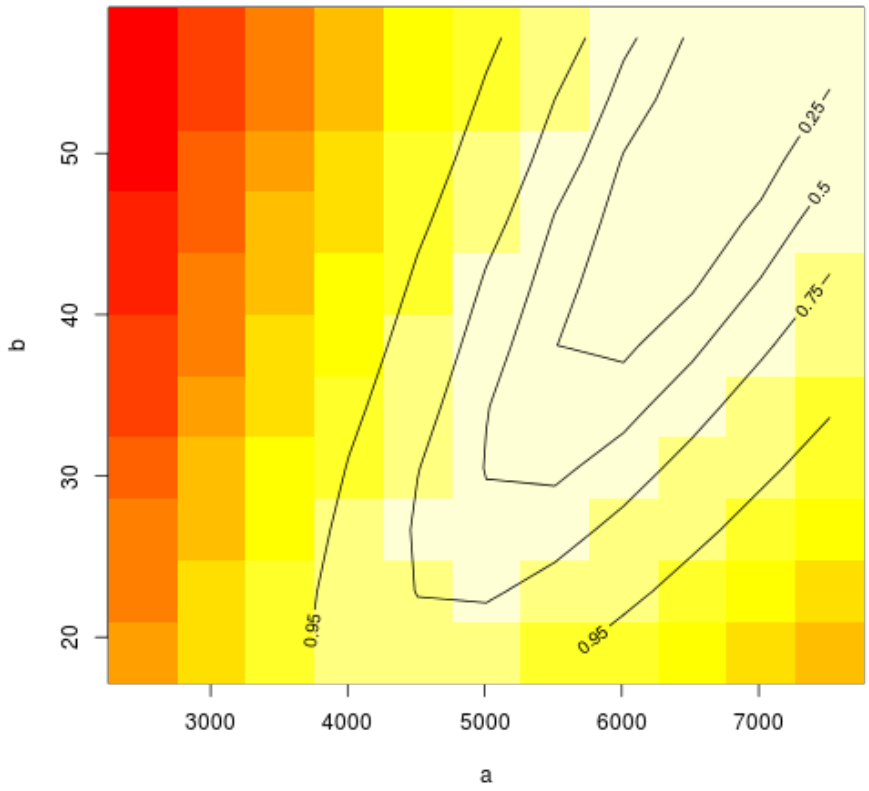

Figure 1: plot of chunk profplot

Adding other SR models

Further Reading

References

Beverton, R.J.H. and Holt, S.J. (1957) On the dynamics of exploited fish populations. MAFF Fish. Invest., Ser: II 19, 533.

Needle, C.L. Recruitment models: diagnosis and prognosis. Reviews in Fish Biology and Fisheries 11: 95-111, 2002.

Ricker, W.E. (1954) Stock and recruitment. J. Fish. Res. Bd Can. 11, 559-623.

Shepherd, J.G. (1982) A versatile new stock-recruitment relationship for fisheries and the construction of sustainable yield curves. J. Cons. Int. Explor. Mer 40, 67-75.