FLsp - a Surplus Production model in FLR

Finlay Scott <finlay.scott@cefas.co.uk
Cefas, Lowestoft, UK

June 2011

1 Introduction

This package implements the surplus production described in Polacheck REF. At the moment only the model including observation error is implemented. The Pella Tomlinson shape is used (which defaults to Schaeffer) Tested against the three data sets in the paper. Accurate gradients and hessian are returned using automatic differentiation (implemented using ADOLC REF)

2 The Model

Following the Polacheck paper Assumptions: Observation error Only r and k are estimated sigma and q are approximated as described in the paper The inital biomass is k

The general equation for the biomass through time is:

$$B_{y+1} = B_y + g(B_y) - C_y \tag{1}$$

where B is the stock biomass at the start of year y, C is the catch during the year and g is surplus production as a function of biomass.

Here we implement the Pella-Tomlinson form of surplus production:

$$g(B) = \frac{r}{p} B(1 - (B/K)^p) \tag{2}$$

where r is the intrinsic growth rate parameter and K is the average biomass level prior to exploitation. By default, p is set to 1 making the surplus prodution formulation the same as a Schaefer model.

The biomass is related to an index of abundance:

$$I_y = qB_y \tag{3}$$

Where I is an index of relative abundance in year y and q is the catchability coefficient.

Here we assume that errors are introduced through observation. The population dynamics are assumed to be deterministic and all of the error occurs in the relationship between stock biomass and the index of abundance. It is assumed that the error is multiplicative and log-normal with a constant coefficient of variation. The estimates of the model parameters are (B_0, r, K and q) are obtained by maximising the likeilhood function:

$$L = \prod exp(\hat{v}_y^2/(2\hat{\sigma}_v^2))/(\sqrt{2\pi}\hat{\sigma}_v) \tag{4}$$

where the product is over all years for which CPUE data are available and:

$$\hat{v}_y = log(C/E)_y - log(\widehat{C/E})_y \tag{5}$$

$$\hat{\sigma}_v^2 = \sum \hat{v}_y^2/n \tag{6}$$

where n is the number of data points.

The value of q which maximises the likelihood is given by:

$$\hat{q} = exp\left(\frac{1}{n}\sum_y log(I_y/\hat{B}_y)\right) \tag{7}$$

Following Polacheck et al B_0 is set to K. This means that only two parameters need to be estimated: r and K. In *FLsp* the estimation is performed using the *DEoptim* package REF.

3 The FLsp class

The *FLsp* class extends the *FLModel* class by including slots to store the catch and index time series. Catch is represented as an *FLQuant* and index is represented as an *FLQuants* object. This allows multiple indices to be used (not yet implemented).

To estimate the parameters r and K, an *FLsp* object must be created with catch and index data. The method *fitsp*() is then called. Once the object has been fitted, the biomass trajectory and other variables of interest (e.g. $sigma^2$ and $\hat{q}$ can be calculated).

4 Creating and fitting FLsp objects

Here we show how to create and fit a surplus production model using *FLsp*. The data set is New Zealand Rock Lobster, taken from Polcheck REF.

```
> # Load the library
> library(FLsp)
> # Load the New Zealand Rock Lobster data set
> data(nzrl)
> # This is a dataframe with year, catch and cpue
> # Take a look at the top of it
> head(nzrl)
> # Make FLQuant objects of the catch and cpue series
> catch <- FLQuant(nzrl$catch, dimnames=list(year=nzrl$year))
> index <- FLQuant(nzrl$cpue, dimnames=list(year=nzrl$year))
> # Create the FLsp object
> nzrl <- FLsp(catch=catch,index=index)
```

After creating our object we are ready to fit the parameters.

```
> nzrl <- fitsp(nzrl)
```

The published values for this data set are: $r = 0.0659$, $K = 129000$, $\hat{q} = 2.461\text{e-}5$, $\sigma = 0.207$, $B_{current} = 21150$. These can be compared to our results by interrogating the *FLsp* object.

```
> # Look at the fitted parameters
> params(nzrl)

An object of class "FLPar"
params
         r          k
4.9406e-02 1.4236e+05
units:  NA

> # qhat
> qhat(nzrl)

An object of class "FLQuant"
, , unit = unique, season = all, area = unique


     year
quant 1
  all 2.2126e-05

units:  NA

> # sigma2
> sqrt(sigma2(nzrl))

An object of class "FLQuant"
, , unit = unique, season = all, area = unique


     year
quant 1
  all 0.20695

units:  NA

> # returns the full biomass timeseries
> biomass(nzrl)

An object of class "FLQuant"
, , unit = unique, season = all, area = unique


     year
quant 1945   1946   1947   1948   1949   1950   1951   1952   1953   1954
  all 142356 141547 140733 139893 138653 136959 134544 132074 129222 125651
     year
quant 1955   1956   1957   1958   1959   1960   1961   1962   1963   1964
  all 120838 116732 111223 107375 104232 101593  99268  96711  93660  90689
     year
quant 1965   1966   1967   1968   1969   1970   1971   1972   1973   1974
  all  87718  84397  80800  77744  74513  71481  68540  65818  64071  62028
     year
quant 1975   1976   1977   1978   1979   1980   1981   1982   1983   1984
  all  60114  58843  57238  55692  53949  51554  48989  46518  43734  40846
     year
quant 1985   1986   1987   1988   1989   1990
  all  37374  33880  30498  27182  25141  22846
```

	New Zealand Rock Lobster		South Atlantic Albacore		Northern Namibian Hake	
Measure	FLsp	Published	FLsp	Published	FLsp	Published
r	0.0494	0.0659	0.315	0.328	0.37	0.379
K ('000 t)	142	129.0	246	239.6	2820	2772.6
σ	0.207	0.207	0.11	0.111	0.125	0.124
$\hat{q}$	2.21e-05	2.461e-05	0.261	0.2671	0.000427	4.360e-04
$B_{current}$	22.8	21.15	76.2	75.51	1660	1646.3
MSY	0	2133.74	0	19.65	0	263.2

Table 1: Comparing the published results with those from *FLsp* for three data sets.

It can be seen that there is good agreement between the published results and those generated with *FLsp*. The differences are likely caused by the precision of the printed data set in the Polcheck paper (REF) and the fitting method used.

5 Testing FLsp against the other data sets

The results fitted with *FLsp* are in good agreement with the published results.

6 Plotting results

There is no generic plot for FLsp at the moment. However, it is possible to look at the fitted index and residuals using relatively simple code. For example, to plot the indices with the fitted indices you can use (see Figure 1):

```
> fitted <- cbind(as.data.frame(nzrl@fitted_index),type="fitted")
> index <- cbind(as.data.frame(nzrl@index),type="index")
> index <- rbind(index,fitted)
> print(xyplot(data ~ year | qname, group=type, data=index, type="b",auto.key=TRUE))
```

To look at the residuals and put a loess function through them use (see Figure 2):

```
> residuals <- as.data.frame(nzrl@residuals_index)
> print(xyplot(data ~ year | qname, data = residuals, panel = function(x,
+     y) {
+     panel.xyplot(x, y)
+     panel.loess(x, y, span = 1)
+ }))
```

7 Profiling the fit

You can explore how good the fit is by looking at the likelihood profile. This is easily done by using the *profile*() method (see Figure 3).

Notice that the profile plot has a banana shaped flat section which contains the optimum solution. This is because the parameters r and K are correlated, making them difficult to estimate unless there is sufficient information in the data. The profile plot also includes the gradient of the log likelihood as r and K change

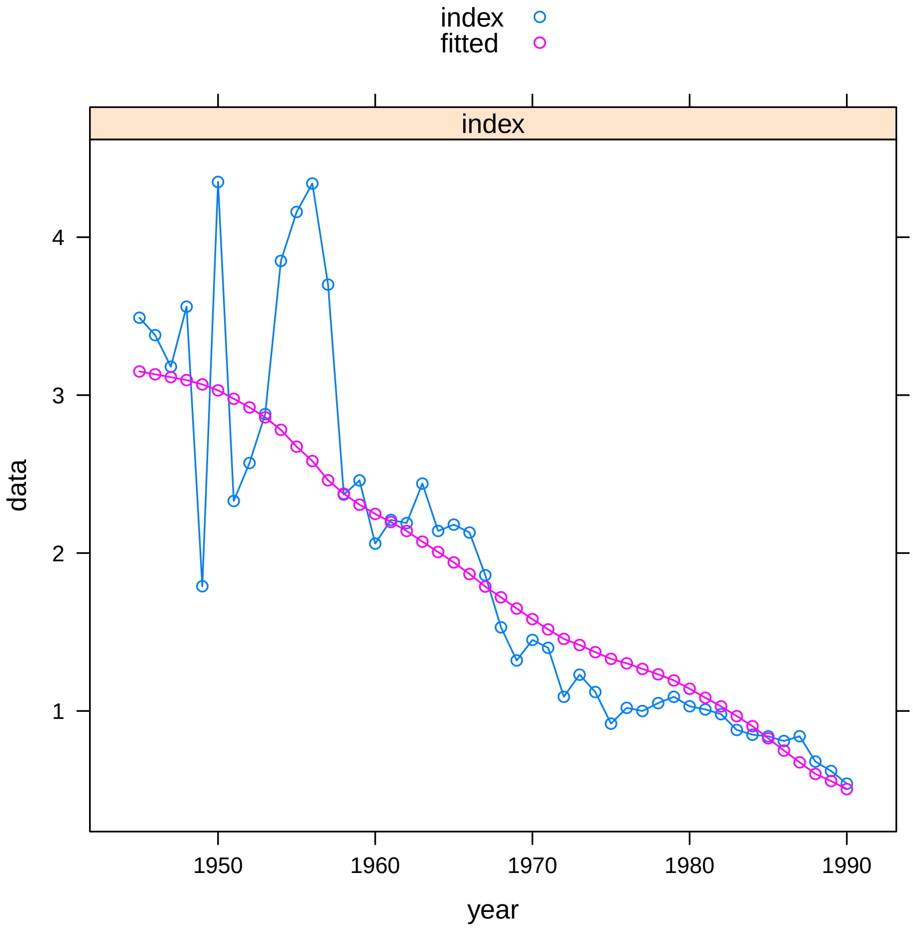

Figure 1: Indices and fitted indices for New Zealand rock lobster

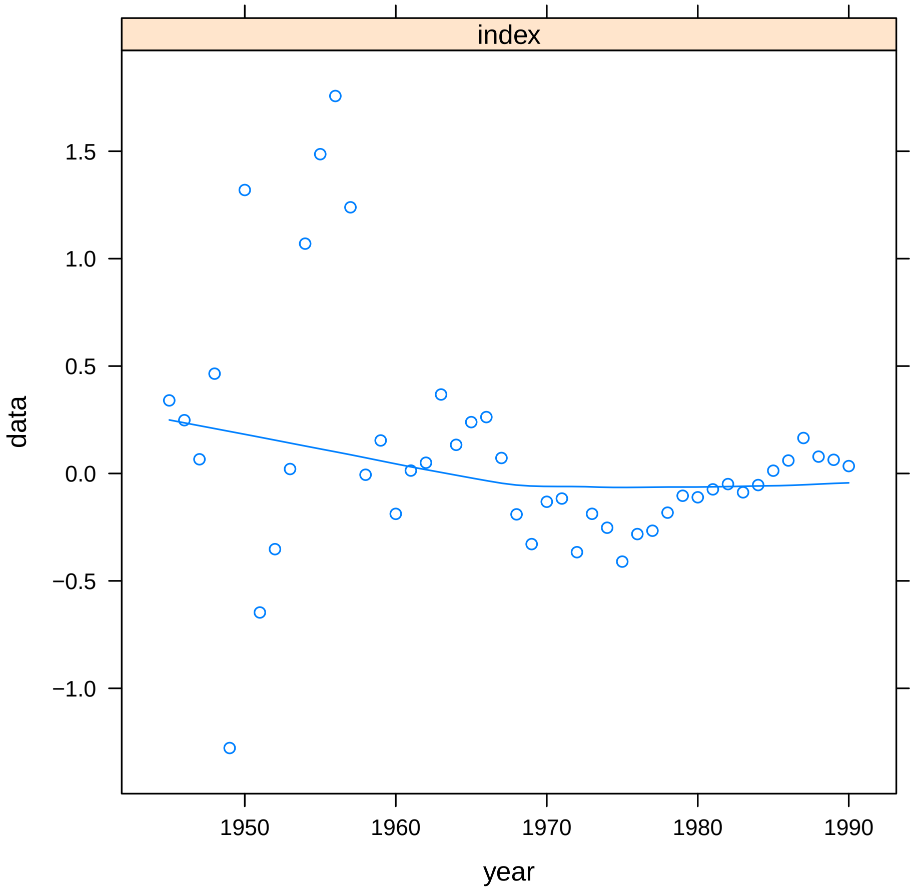

Figure 2: Residuals for New Zealand rock lobster

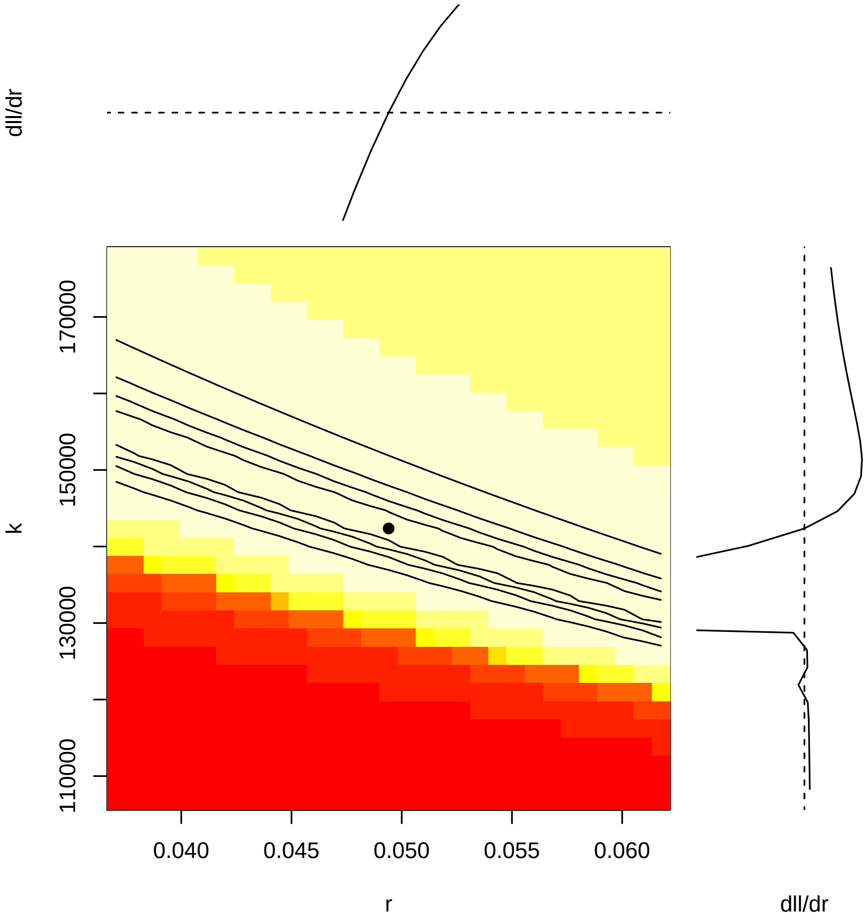

Figure 3: Profile plot for New Zealand rock lobster

(keeping K and r fixed at the estimated value found by the optimiser respectively). The dashed line is at a gradient of 0. If the fitting has worked, the gradient should be 0 at the estimated parameter values. It is just another simple way to check that the results of from the fitting process are sensible.

```
> profile(nzrl, maxsteps = 31, range = 0.25)
```

8 Uncertainty

We are trying to generate a range of simulated values of r and K that have the same statistical properties as the esimated values. i.e. the mean and the variance-covariance matrix of the simulated values should be the same as for the estimated values.

This will be done here by following the method described in Gentle (Elements of Computationsl Statistics REF). The method involves generating a vector of values that are independent and identically distributed (i.i.d.) and transforming them to simulated values of r and K. The vector of i.i.d. values is called Y and has a variance equal to 1 and a mean equal to 0. We want to generate a transformed variable X (here, pairs of values of r and K) so we need to generate A, a non-singular matrix, that gives the transformation:

$$X = AY \tag{8}$$

The variance-covariance matrix of the transformed random variable X is Σ. As Y is i.i.d. with a variance of 1 and mean of 0, Σ is:

$$\Sigma = AA^T \tag{9}$$

The matrix A can be found using the Cholesky decomposition of Σ. As Y has a mean equal to 0 (to ensure generality of the approach) the mean of the simulated values is also 0. This is straightforward to correct.

This procedure may sound complicated but it is easy to carry out in R! The approach is demonstrated below.

First we want to the get the variance-covariance matrix of the estimated parameters. It is possible to estimate the variance-covariance matrix from the matrix of second-order (partial) derivatives, called the Hessian matrix, which is returned in the fitted *FLsp* object. If we assume the parameter distribution has (multivariate) normality, it is a standard statistical result that $\mathbf{V} = (-\mathbf{A})^{-1}$ (REF) i.e. the variance-covariance matrix is the inverse of the negative hessian matrix.

This relationship makes sense if you think about the change of slope of the likelihood function at the estimated parameter values. If the change is very sharp (i.e. if you look at the likelihood profile there is a clearly defined minimum), then the second-order derivative will be relatively large. This means that there will be lot of confidence in the parameter estimate because it is clearly identifiable and hence the standard error will be small. On the other hand, if the second-order derivative is low, then the change in the slope around the function minimum is low (the likelihood profile will look flat). This means that the parameter value can vary in any direction without greatly affecting the value of the likelihood function. This implies that the standard error of the parameter will be large.

```
> vcov.matrix <- solve(-1 * nzrl@hessian[,,1])
```

We are ready to get the Cholesky factor of the variance-covariance matrix

```
> A <- t(chol(vcov.matrix))
```

Now we create a vector of i.i.d. values. We use values pulled from a normal distributution with a mean of 0 and a standard deviation of 1. This means that we are making the assumption that r and K are multivariate normal random variables. IS THIS SAFE?

```
> nsam <- 1000
> Y <- matrix(nrow=nsam,ncol=2)
> Y[,1] <- rnorm(nsam)
> Y[,2] <- rnorm(nsam)
```

Finally we multiply the decomposed Cholesky matrix by the pairs of uncorrelated i.i.d. values. We also have to correct the mean.

```
> X <- t(apply(Y,1,function(x,A){x <- A%*%x},A))
> colnames(X) <- c("r","k")
> X <- sweep(X,2,params(nzrl),"+")
```

We can then check the statistical properties of the simulated parameter values.

```
> cov(X)

              r             k
r  2.009701e-03 -1.938838e+03
k -1.938838e+03  1.874203e+09

> vcov.matrix

              r             k
r  1.874934e-03 -1.804984e+03
k -1.804984e+03  1.741359e+09

> apply(X,2,mean)

           r            k
4.715716e-02 1.445400e+05

> c(params(nzrl))

[1] 4.940571e-02 1.423562e+05
```

We can see that the variance-covariance matrix and the mean of the simulated values of r and K are the same as for the estimated values.