R goes fishing: Analyzing fisheries data using AD Model Builder and R

Arni Magnusson

22 August 2005*

Abstract

Fisheries stock assessment is mainly about two questions: what is the current stock status, and what are the consequences of alternative harvest levels. The data provide indirect information in the form of annual landed catch, age distribution, index of relative abundance, and biological features like length, weight, and maturity at age. Other possible data include tag recoveries, juvenile surveys, and acoustic estimates of absolute abundance. Due to these diverse data types, as well as economic importance, fisheries stock assessment has pushed the limits of applied statistics over the last fifty years. The dynamic models are highly nonlinear, include dozens of parameters, and the objective function consists of several likelihood components. The most efficient algorithm to fit these models is automatic differentiation, as implemented with AD Model Builder, a dialect of C++. To diagnose the model fit, trellis plots in R have proven very useful, and the coda package is used to diagnose MCMC convergence. Here, I demonstrate the functions I have implemented to make these tasks easier, both for newcomers to R and old dogs.

Introduction

There are two goals I hope to achieve with my presentation and these accompanying notes:

- Briefly describe fisheries stock assessment methodology to scientists who also work with applied statistics, in hope to learn from one another.
- Introduce and demonstrate a forthcoming R package called **scape** (statistical catch-at-age plotting environment).

When selecting the references, I strive to include only those that are available online, free or by subscription, to make it easier for the casual reader to look something up. Let me begin with the only exception to this rule, by recommending three excellent textbooks on fisheries stock assessment that are available at major bookstores.

Hilborn and Walters (1992) introduce the main concepts, popular models and common mistakes, Quinn and Deriso (1999) explain the models in more mathematical detail, and Walters and Martell (2004) cover modelling and management topics that have received the most attention during the last decade or so. The books complement each other quite well.

*Presented at the *Directions in Statistical Computing* conference in Seattle, 12–14 August 2005.

1 Fisheries stock assessment

1.1 Questions

Fisheries management is in the realm of politics, synthesizing social, economic and ecological factors to choose an appropriate management action. It relies on stock assessment to get an idea about the current stock status, and the consequences of alternative harvest levels. We focus on stock assessment here, which is perhaps better described as a technical task than as empirical science. Results from the "experiments" are typically not reproducible, and even if they were, two scientists might reach very different conclusions, due to subjective decisions in the data analysis.

The interface between stock assessment output and fisheries management input can use Bayesian decision analysis (Punt and Hilborn 1997) as a coherent statistical foundation. The basic idea is to evaluate the likely outcome of each policy option, given each possible state of nature. After assigning probabilities to the possible states of nature and defining a quantitative objective, the best policy option can be inferred. The procedure is revisited each year or every few years, depending on the commercial importance of the fishery and the level of depletion.

1.2 Data

The observed data come mainly from three sources: the fleet at sea (skipper logbooks and/or observers), ports (landed weight and biological measurements from samples), and research surveys. Most commercially important fish populations have an annual cycle in terms of migration and reproduction, and the fishery operations also have an annual pattern, so it makes sense to aggregate the data within each year. After processing the data, a combination of three data types are particularly useful in stock assessment: annual landings, the age distribution of the catch, and an index of relative abundance over time. The age composition can be informative, both about the relative size of each cohort and about the mortality rate in each year.

Methods to estimate age distributions from length distributions are discussed by Salthaug (2003). Methods to estimate the abundance index from catch rates in the commercial fishery are reviewed by Maunder and Punt (2004), and Helser et al. (2004) demonstrate how a generalized linear mixed model can be used to analyze similar data from research surveys. Alternatively, the catch rate analysis could be integrated into the stock assessment model (Maunder and Langley 2004).

1.3 Models

The objective of a stock assessment model is to reliably estimate the stock status and predict the outcomes of alternative policy options. Ideally, it should use all the available data and capture the underlying dynamics with an appropriate level of realism. The key biological processes include recruitment, natural mortalities, fishing mortalities, growth, and spawning.

In the simplest models all fish are identical, but when age specific data are available, more complex models can be used that keep track of age groups with different body size, maturity, probability of capture, and so on. It is common to assume a closed population, with no immigration or emigration, and a constant rate of natural mortality across ages and years. This last assumption is often criticized, but in practice, fisheries data contain very little information to partition what proportion of a cohort died from fishing and what proportion from natural (other) causes.

Smith and Addison (2003) review a variety of stock assessment models, and Hilborn (2003) describes recent development in the field. Modern textbooks generally recommend using a family of models called statistical catch-at-age (SCA). They are likelihood-based and generalized so they can be fitted to many kinds of data simultaneously. Furthermore, the modeller can choose which assumptions to make, as opposed to older techniques where implicit assumptions are not easily relaxed. SCA models can be applied when large amounts of diverse data are at hand, or when data are scant and include years of missing data.

There are mainly three software implementations of SCA models that have been distributed and are currently used in routine assessments around the world: Stock Synthesis (Methot 2000, available by request), Coleraine (Hilborn et al. 2003, available on the web), and CASAL (Bull et al. 2004, available by request). Many stock assessment scientists, however, have implemented their own software, some of which may be distributed in the near future.

1.4 Parameter estimation

When using a SCA model, the user first decides which quantities are treated as known, and which are subject to observation error. Then, the distribution and magnitude of the error is assumed, based on ecological theory and empirical data. Finally, a likelihood function is evaluated for each data component. For example, the age distribution of the catch could be assumed to have multinomial observation error, and the abundance index to have lognormal observation error. The maximum likelihood parameter estimates are found at the best fit, where the objective function (sum of all negative log likelihoods) is minimized.

Automatic differentiation (e.g. Skaug 2002) is by far the most efficient algorithm that fisheries scientists have come across for minimizing the objective function of these dynamic nonlinear models. The most popular software implementation among fisheries scientists is AD Model Builder (Otter Research 2004). In his software review, Maunder (2000) gives a concise description: "AD Model Builder (ADMB) is a computer programming template and a set of C++ libraries that aid in the development and parameter estimation of non-linear models, particularly models that include a large number of parameters to be estimated." It is a commercial product available for Windows or Linux from `http://otter-rsch.com/`.

The ADMB template syntax, resembling both R and C, is translated to C++ and then compiled with any standard compiler. The `autodif` library is at the heart of ADMB, but other convenient features include linear algebra, random effects, and many commonly used statistical functions. The user specifies the bounds and estimation phase for each parameter, giving the user the opportunity to begin estimating some parameters while keeping others constant. In the final estimation phase, all parameters are estimated simultaneously.

The default ADMB output is text files with point estimates and the variance and correlation matrix of estimated parameters, and of derived quantities if any are defined. This matrix is estimated using the delta method, via the inverse of the Hessian of the objective function. Optionally, the user can run Markov chain Monte Carlo (MCMC) simulations, again saving the output in text files. In recent years, ADMB has become a common platform to discuss and implement fisheries stock assessment models, making it easier for two or more scientists to pinpoint the differences between their modelling methods and results.

1.5 Uncertainty

A variety of methods can be used to evaluate the uncertainty about estimated parameters and derived quantities. Two have already been mentioned here; the delta method and MCMC. Other popular methods include likelihood profiles, sampling importance resampling (SIR), and many variations of the bootstrap.

Patterson et al. (2001) and Gavaris and Ianelli (2002) discuss the differences between those methods, both practical and theoretical. Like in other fields of applied statistics, the MCMC method has been adopted by many fisheries scientists. With software like ADMB or BUGS (Meyer and Millar 1999) it's easy for the user to run MCMC, and for most models the computations will be finished when we come to work in the morning—perhaps after running variations of the same model on different computers, to address model uncertainty.

Comparative simulation studies are still needed to determine which uncertainty methods perform reliably for stock assessment models.

1.6 Discussion

In this brief overview of stock assessment methodology, many details have been left out. The references should be useful to study the various methods and trace their historical development. Some recent methods that should at least be mentioned are meta analysis and the use of Bayesian priors for estimated parameters (Hilborn and Liermann 1998, Myers et al. 1999), as well as state space models (Millar and Meyer 2000) that admit both observation and process error in the data. Traditional alternatives to SCA models are described by Shepherd (1999) and Gavaris and Ianelli (2002).

Today's fisheries managers are not only asking about the current stock status and consequences of alternative harvest levels. They're also interested in the ecological and evolutionary effects each fishery has on every element of the marine ecosystem. Moreover, there are more policy options to choose from than just different harvest levels, such as gear restrictions and temporary or permanent area closures.

The winter 2003 issue of Natural Resource Modeling, Vol. 16(4), was devoted to eight insightful essays on current trends in fisheries stock assessment models. Fisheries scientists are tackling more complex questions today than in the past, and both accurate single-species models and holistic ecosystem models are now forming the basis of scientific management advice.

2 The scape package

scape, n. *A view of scenery of any kind, whether consisting of land, water, cloud, or anything else.*
—Oxford English Dictionary

2.1 Approach

The importance of visualizing the data and model fit increases when complex models are fitted to several data types simultaneously. Richards et al. (1997) gave guidelines for doing this efficiently in fisheries stock assessment, emphasizing that visual inspection of the data can reveal important features that must be accommodated in the model development. Their study also demonstrates how the final report should include plots with routine information, as well as plots showing specific patterns that emerged in that assessment.

The goal of **scape** is to assist the statistical catch-at-age (SCA) modeller by providing an environment to plot the observed data and fits, diagnose residuals, plot posterior distributions, and assess MCMC convergence. It should:

- Import SCA model results, but assume as little as possible about the model implementation and output file formats.
- Provide multipanel plotting functions whose default output is of adequate quality for on-the-fly analysis, presentation slides, and technical reports.

Given the exploratory and graphical nature of the problem, R is a highly suitable platform. Trellis plots (Becker et al. 1996, Sarkar 2005) are used for graphical display and the **coda** package (Best et al. 1995, Plummer et al. 2005) is used for MCMC convergence diagnostics.

The basic idea in **scape** is that a fitted model is imported and stored as a list containing:

`N`	N@A matrix (predicted numbers at age by year)
`B`	year-things (biomass, landings, recruitment)
`Sel`	age-things (selectivity, maturity)

The list may also contain, depending on the assessment, any or all of the following:

`Dev`	recruitment deviates
`CPUE, Survey`	abundance indices and fit
`CAc, CAs`	commercial and survey C@A (catch at age) and fit
`CLc, CLs`	commercial and survey C@L (catch at length) and fit
`LA`	L@A (length at age) and fit

The only thing **scape** assumes about the assessment model is that the results can be imported into R and stored as a list with these elements. Results from Coleraine, Stock Synthesis, CASAL, or any other model can be imported, but so far only one import function has been implemented, for Coleraine.

Like other R packages, **scape** also assumes that the user is familiar with basic R concepts, in this case reading data from text files and manipulating lists. Some knowledge of trellis plots would also be helpful.

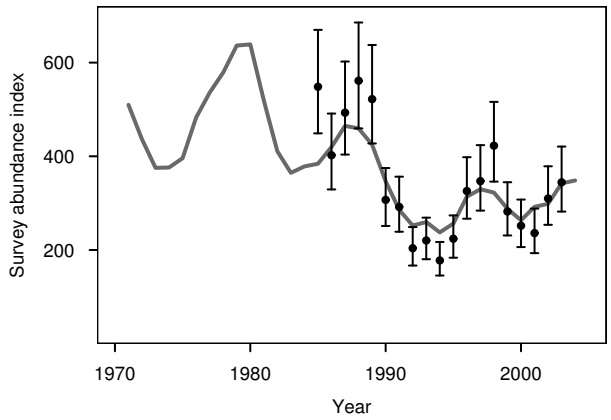
An illustrated tour follows. Besides **scape**, four packages are required: **coda**, **gdata**, **Hmisc**, and **lattice**.

2.2 Functions and example data

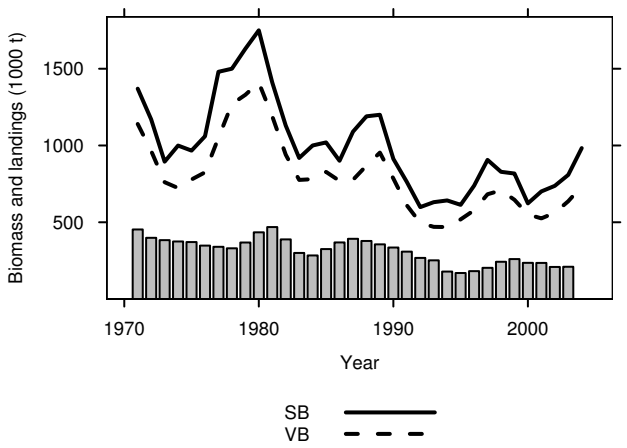
```
> ls()
 [1] "importRes" "plotB"     "plotCA"    "plotCL"    "plotIndex" "plotLA"
 [7] "plotN"     "plotSel"   "x.cod"     "x.ling"    "x.oreo"    "x.sbw"

> ll()
             Class  KB
importRes function 207
plotB     function  54
plotCA    function  74
plotCL    function  74
plotIndex function  58
plotLA    function  57
plotN     function  72
plotSel   function  51
x.cod        scape  76
x.ling       scape 262
x.oreo       scape 344
x.sbw        scape  49

> args(importRes)
function (res.file, info = "", Dev = FALSE, CPUE = FALSE, Survey = FALSE,
    CAc = FALSE, CAs = FALSE, CLc = FALSE, CLs = FALSE, LA = FALSE,
    verbose = FALSE)
NULL

> Args(importRes)
         value
res.file
info     ""
Dev      FALSE
CPUE     FALSE
Survey   FALSE
CAc      FALSE
CAs      FALSE
CLc      FALSE
CLs      FALSE
LA       FALSE
verbose  FALSE

> x.cod  <- importRes("c:/scape/example/cod.res", Dev=T, Survey=T, CAc=T, CAs=T)
> x.ling <- importRes("c:/scape/example/ling.res", Dev=T, CPUE=T, Survey=T, CAs=T, CLc=T)
> x.oreo <- importRes("c:/scape/example/oreo.res", CPUE=T, Survey=T, CLc=T, CLs=T, LA=T)
> x.sbw  <- importRes("c:/scape/example/sbw.res", Dev=T, Survey=T, CAc=T, verbose=T)

Parsing text file c:/scape/example/sbw.res:

Preamble  vector...vector...vector...vector...vector...vector...vector...vector...OK
N         matrix...OK
B         matrix...vector...vector...OK
Sel       matrix...vector...vector...OK
Dev       vector...vector...OK
Survey    vector...vector...matrix...matrix...OK
CAc       vector...matrix...matrix...OK
```

2.3 Data and model fit

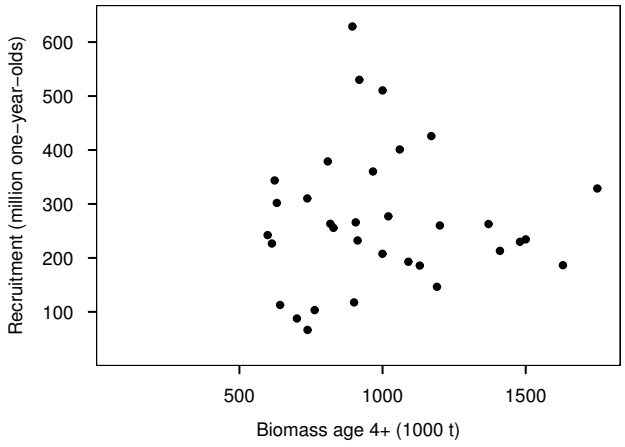
```
> plotIndex(x.cod, "s", strip=F, xlab="Year", ylab="Survey abundance index")
```



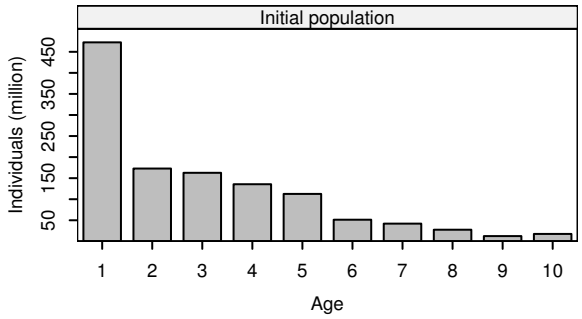

```
> plotB(x.cod, div=1000, xlab="Year", ylab="Biomass and landings (1000 t)",
        lty.lines=c(1,2))
```

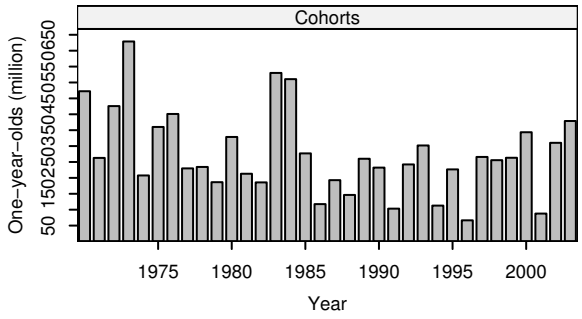



```
> plotB(x.cod, "s", div=1000, xlab="Biomass age 4+ (1000 t)",
        ylab="Recruitment (million one-year-olds)")
```

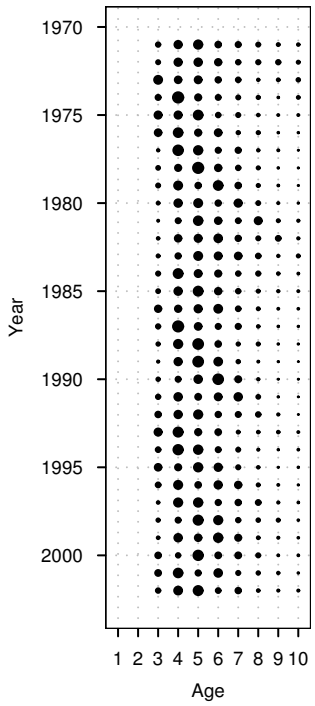


```
> plotN(x.cod, div=1000, xlab=c("Age","Year"),
        ylab=c("Individuals (million)","One-year-olds (million)"))
```

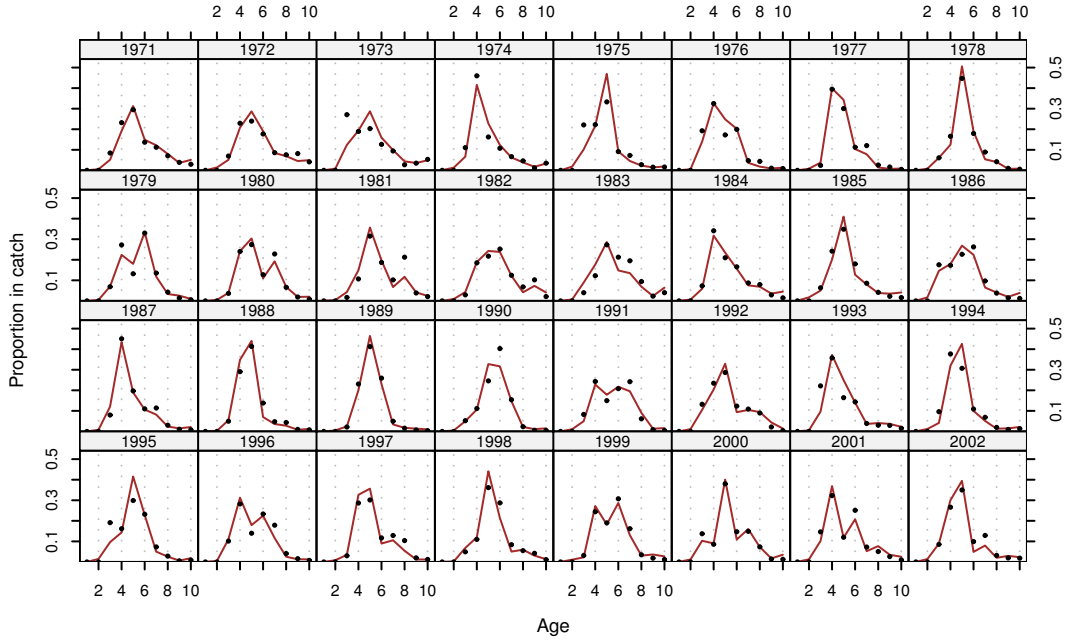




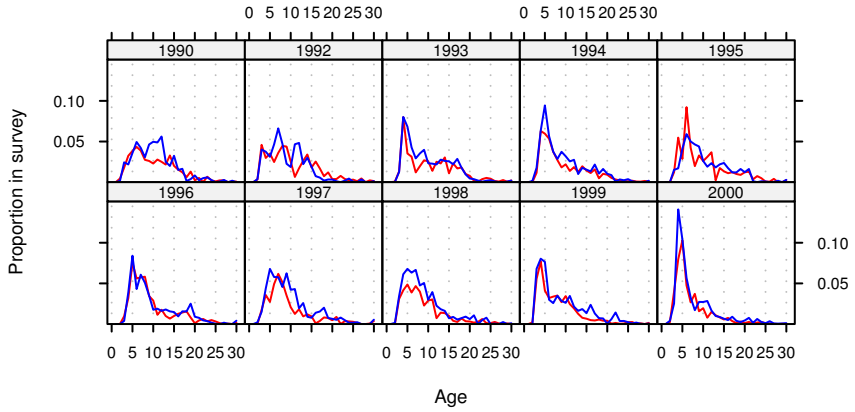

```
> plotCA(x.cod, fit=F, strip=F, xlab="Age", ylab="Year", tick.number=10)
```



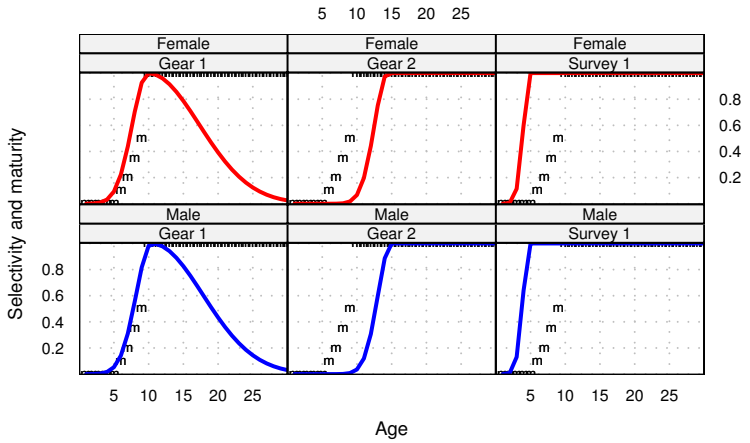
```
> plotCA(x.cod, xlab="Age", ylab="Proportion in catch", col.lines="brown",
        layout=c(8,4))
```



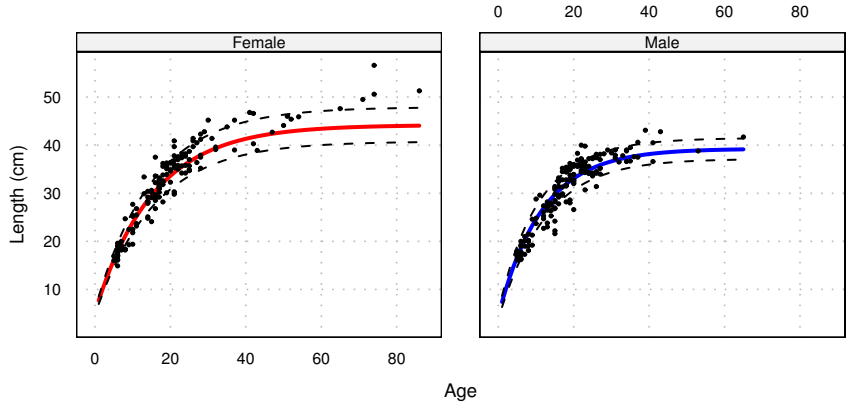

```
> plotCA(x.ling, "s", fit=F, xlab="Age", ylab="Proportion in survey", tck=0.5,
        layout=c(5,2))
```



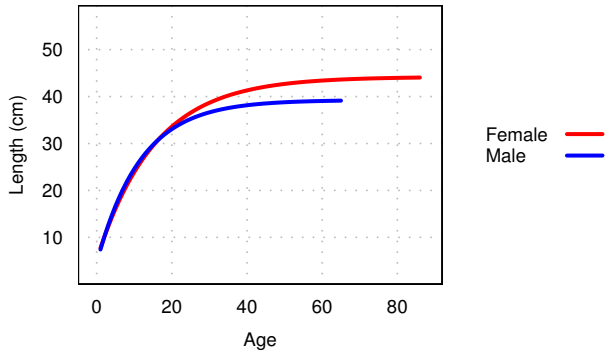

```
> plotSel(x.ling, xlab="Age", ylab="Selectivity and maturity")
```



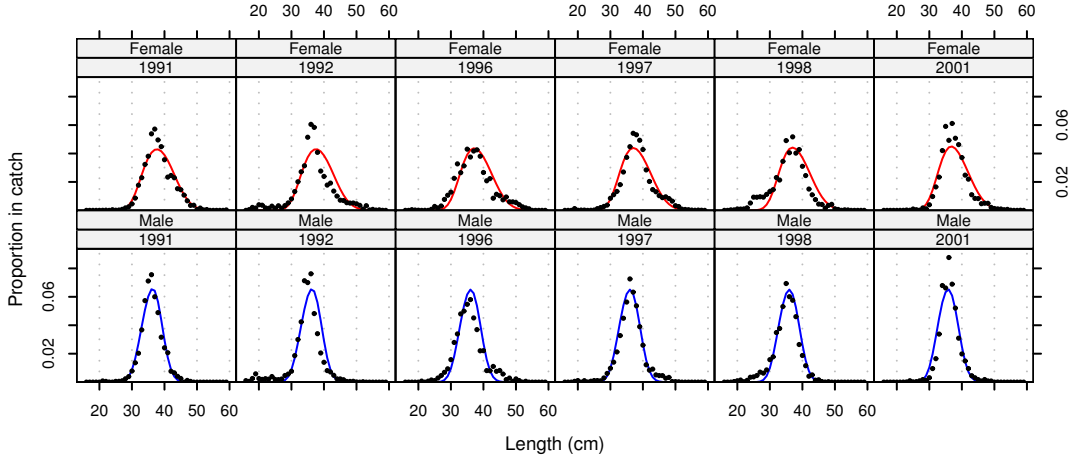
```
> plotLA(x.oreo, xlab="Age", ylab="Length (cm)")
```



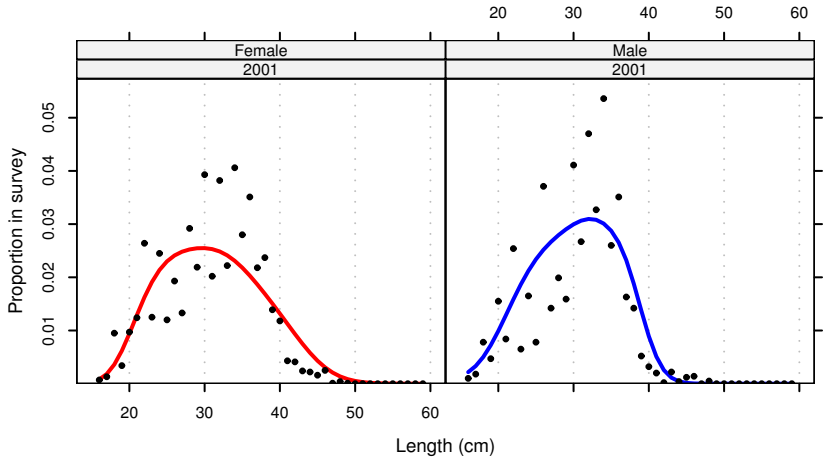

```
> mykey <- list(text=list(lab=c("Female","Male")), space="right",
                lines=list(col=c("red","blue")))
> plotLA(x.oreo, together=T, xlab="Age", ylab="Length (cm)", pch=NA, key=mykey)
```




```
> plotCL(x.oreo, xlab="Length (cm)", ylab="Proportion in catch")
```



```
> plotCL(x.oreo, "s", xlab="Length (cm)", ylab="Proportion in survey", layout=c(2,1))
```



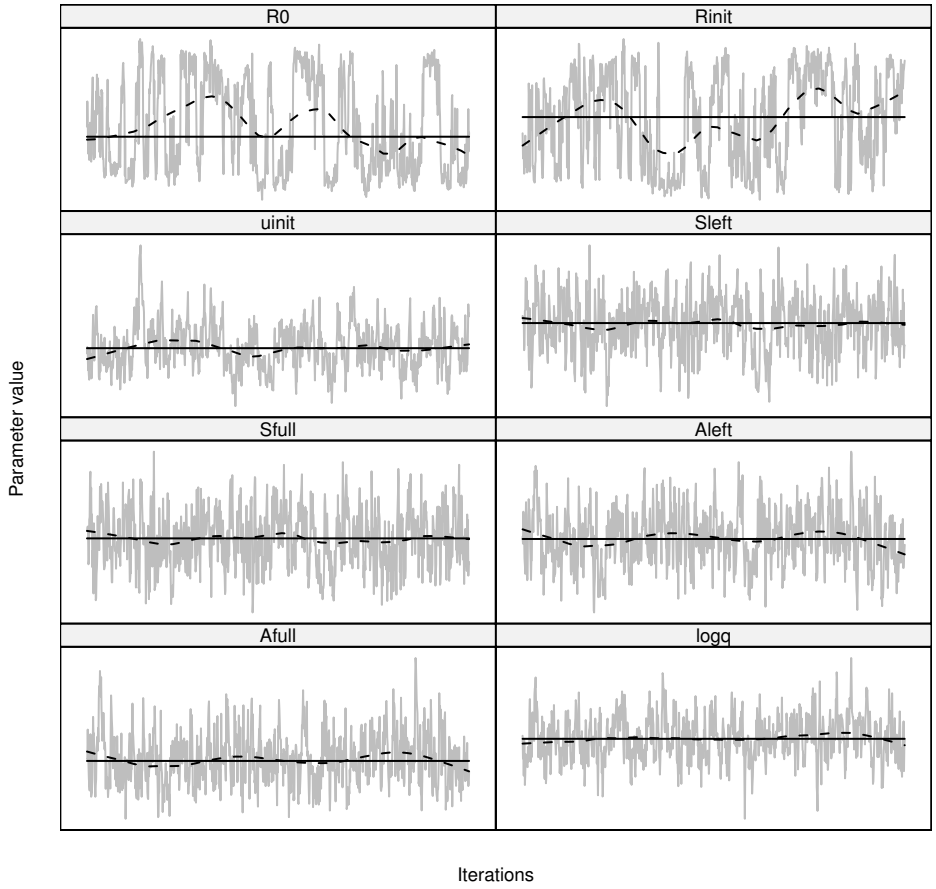
2.4 MCMC results

```
> ll()
              Class   KB
importMCMC  function   57
importPROJ  function   35
plotAuto    function    4
plotCumu    function    4
plotDens    function   30
plotQuant   function   31
plotSplom   function    1
plotTrace   function   30
xmcmc           mcmc  644
xproj      mcmc.list 1560
```

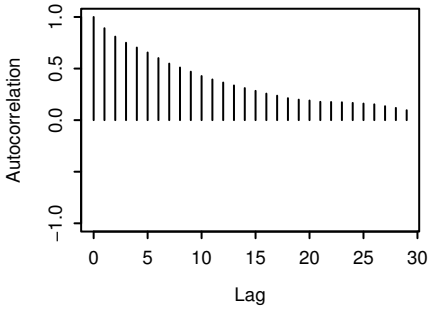
```
> xmcmc <- importMCMC("c:/scape/example/mcmc", verbose=T)

Parsing files in directory c:/scape/example/mcmc:

Likelihoods  file...labels...choose...OK
Parameters   file...labels...choose...OK
Biomass      file...labels...OK
Recruitment  file...labels...OK
```

```
> ll(xmcmc, dim=T)
       Class  KB       Dim
L data.frame  79  1000 x 5
P data.frame  99  1000 x 8
B data.frame 268 1000 x 34
R data.frame 198 1000 x 33
```

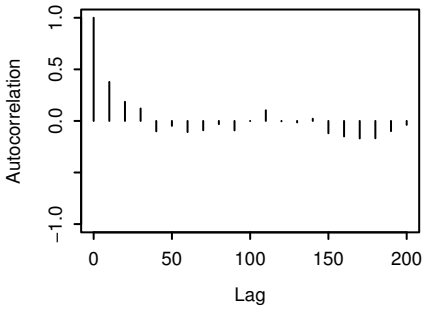
```
> plotTrace(xmcmc$P, xlab="Iterations", ylab="Parameter value", layout=c(2,4))
```



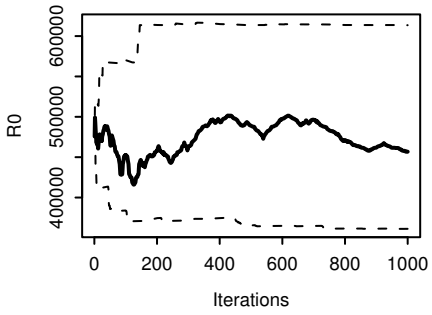

```
> plotAuto(xmcmc$P$R0)
```



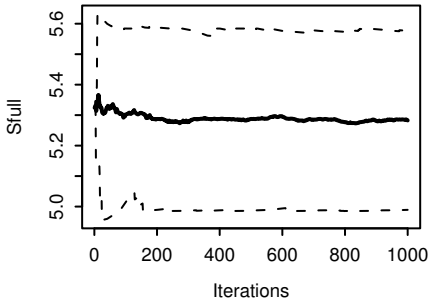

```
> plotAuto(xmcmc$P$R0, thin=10)
```



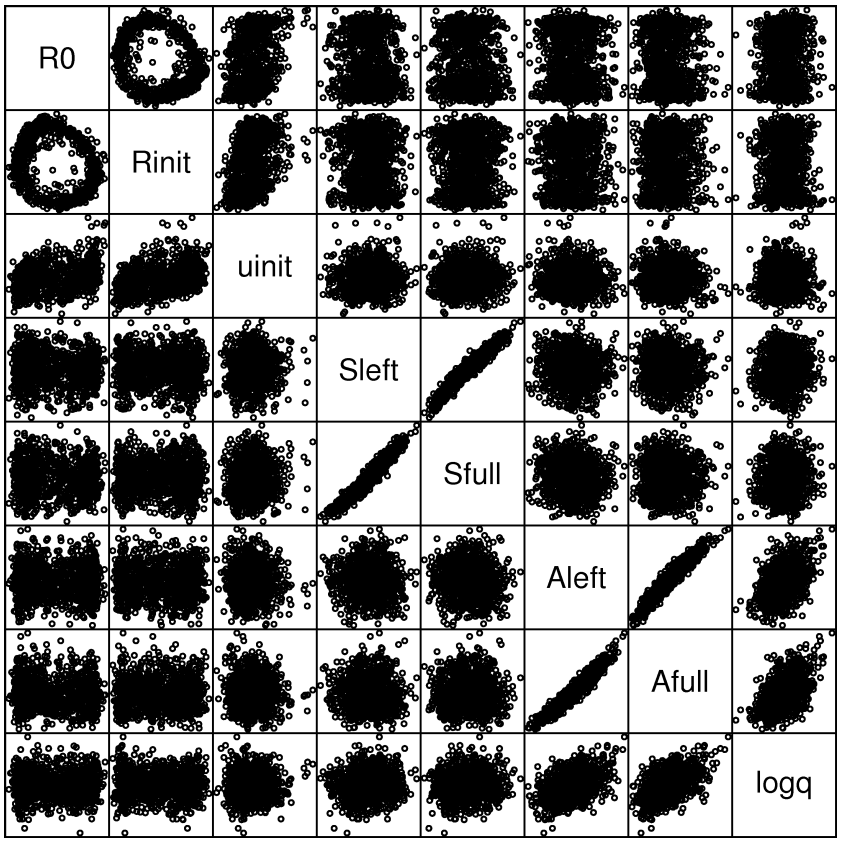
```
> plotCumu(xmcmc$P$R0, ylab="R0")
```



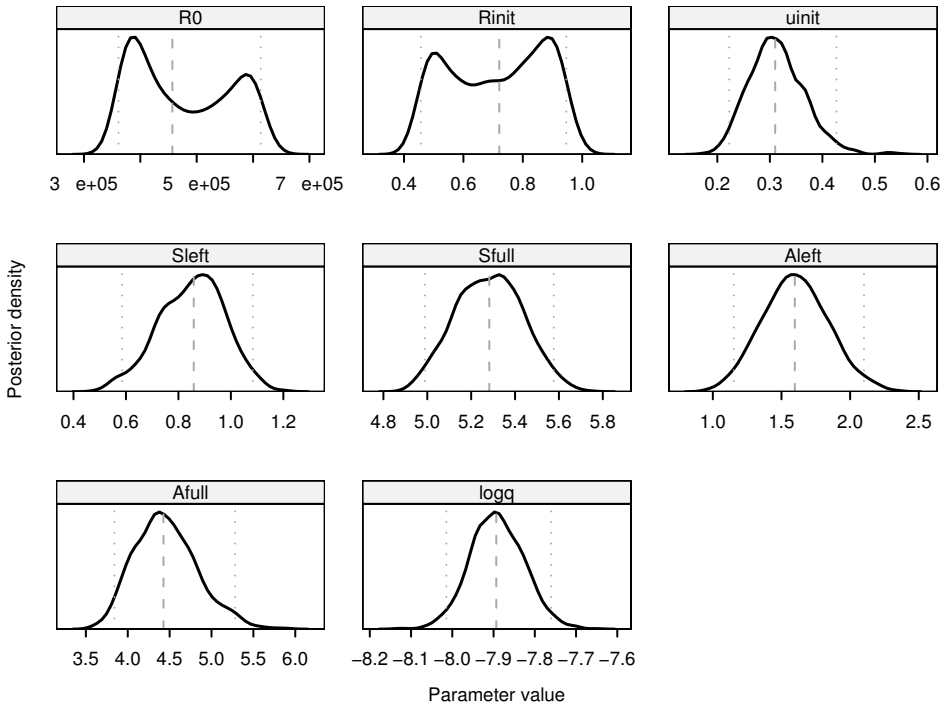

```
> plotCumu(xmcmc$P$Sfull, ylab="Sfull")
```



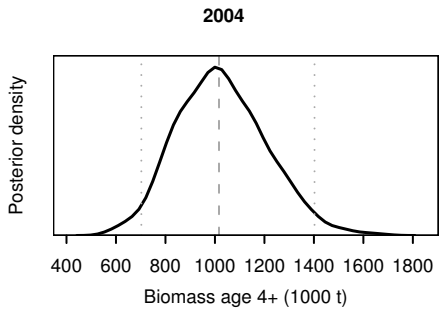

```
> plotSplom(xmcmc$P)
```



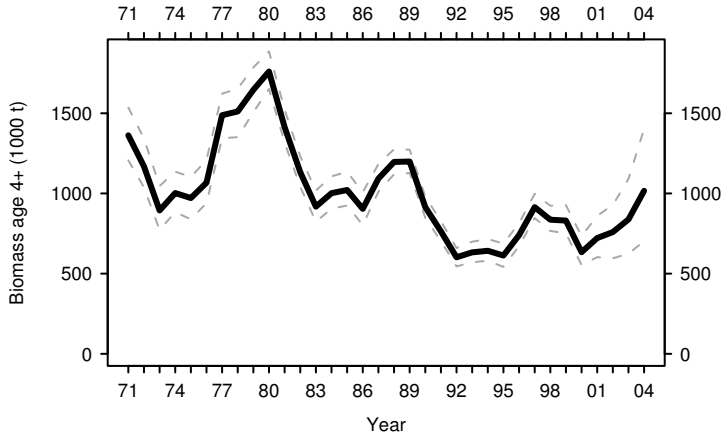
```
> plotDens(xmcmc$P, xlab="Parameter value", ylab="Posterior density", axes=T)
```



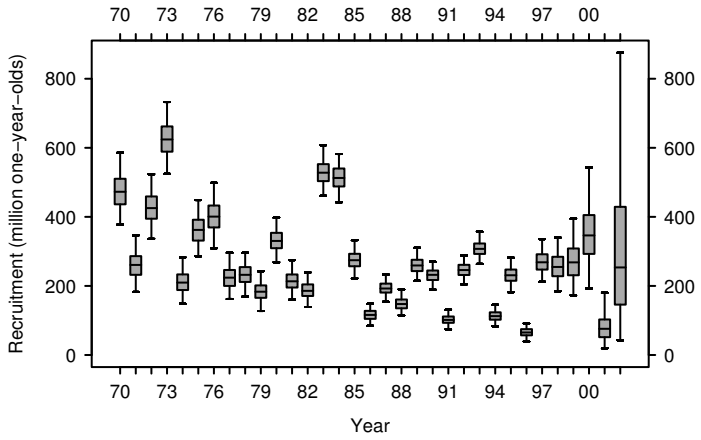

```
> plotDens(xmcmc$B$"2004"/1e3, main="2004\n", xlab="Biomass age 4+ (1000 t)",
          axes=T, strip=F)
```




```
> plotQuant(xmcmc$B/1e3, style="lines", xlab="Year", ylab="Biomass age 4+ (1000 t)")
```

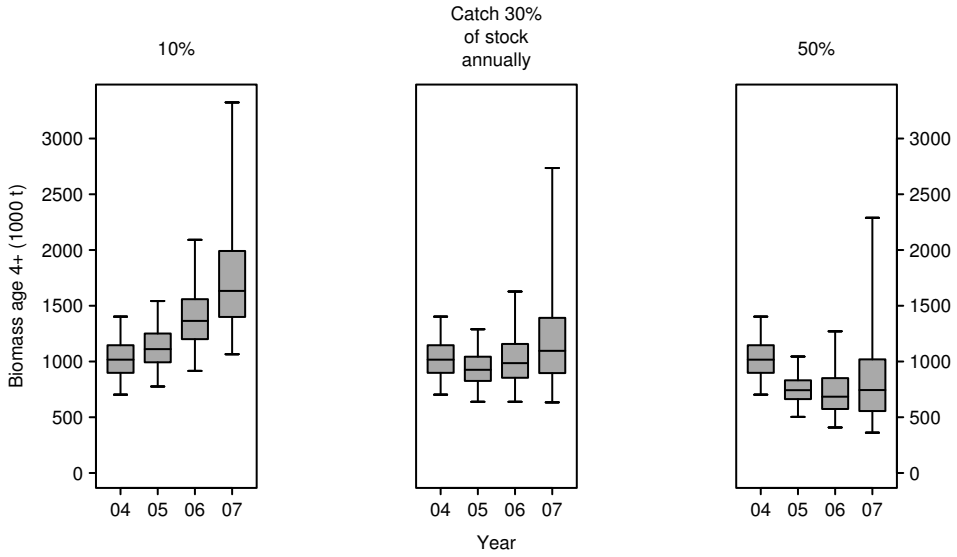


```
> plotQuant(xmcmc$R/1e3, style="boxes", xlab="Year",
            ylab="Recruitment (million one-year-olds)")
```




```
> ll()
               Class   KB
importMCMC  function   57
importPROJ  function   35
plotAuto    function    4
plotCumu    function    4
plotDens    function   30
plotQuant   function   31
plotSplom   function    1
plotTrace   function   30
xmcmc           mcmc  644
xproj      mcmc.list 1560

> xproj <- importPROJ("c:/scape/example/proj", verbose=T)

Parsing files in directory c:/scape/example/proj:

Policies  file...unique...OK
Years     file...labels...unique...OK
Biomass   file...list...OK
Landings  file...list...OK

> ll(xproj)
  Class  KB
B  mcmc 852
Y  mcmc 707

> ll(xproj$B, dim=T)
          Class KB      Dim
0    data.frame 71 1000 x 4
0.05 data.frame 71 1000 x 4
0.1  data.frame 71 1000 x 4
0.15 data.frame 71 1000 x 4
0.2  data.frame 71 1000 x 4
0.25 data.frame 71 1000 x 4
0.3  data.frame 71 1000 x 4
0.35 data.frame 71 1000 x 4
0.4  data.frame 71 1000 x 4
0.45 data.frame 71 1000 x 4
0.5  data.frame 71 1000 x 4
0.55 data.frame 71 1000 x 4
```

```
> head(xproj$B$"0")
     2004    2005    2006    2007
1  982954 1177150 1578160 2132800
2 1030410 1216270 1890110 2523060
3  945345 1117860 1919950 2288310
4  883340 1062460 1365830 1647320
5  815420 1031500 1296090 2090670
6  950912 1221400 1631770 1854260


> par(mfrow=c(1,3))
> plotQuant(xproj$B$"0.1"/1e3, ylim=c(0,3350), sides=1:2,
            main="\n10%", ylab="Biomass age 4+ (1000 t)")
> plotQuant(xproj$B$"0.3"/1e3, ylim=c(0,3350), sides=1,
            main="Catch\n30% of stock\nannually", xlab="Year")
> axis(4, labels=F)
> plotQuant(xproj$B$"0.5"/1e3, ylim=c(0,3350), sides=c(1,4), main="\n50%")
```



2.5 Manipulating trellis plots

```
> plotLA(x.oreo)
> savePlot("c:/oreo", "png")

> one <- plotIndex(x.cod, plot=F)
> two <- plotCA(x.cod, plot=F)
> three <- plotB(x.cod, plot=F)

> trellis.device("pdf", file="c:/booklet.pdf", color=F)
> one
> two
> three
> dev.off()

> trellis.device("postscript", file="c:/combo.eps", width=6, height=8,
>                horizontal=F, onefile=F, paper="special")
> print(one, position=c(0.0,0.4, 0.4,0.9), more=T)
> print(two, position=c(0.4,0.4, 1.0,1.0), more=T)
> print(three, position=c(0,0, 1,0.4))
> dev.off()
> ?print.trellis
```

```
> trellis.device("win.metafile", width=6, height=8)
> print(one, position=c(0.0,0.4, 0.4,0.9), more=T)
> print(two, position=c(0.4,0.4, 1.0,1.0), more=T)
> print(three, position=c(0,0, 1,0.4))
> dev.off()

> trellis.device("win.metafile", width=8, height=4, color=F)
> plotCL(x.oreo)
> dev.off()  # on Windows clipboard, ready to be pasted into any program

> ll(one, sort=T)
> one$strip <- F
> one                    # strip label is gone
> update(one, strip=T)  # and back
```

3 Technical discussion

3.1 Using scape

The **scape** package can save the stock assessment modeller substantial time by creating plots that are needed on a routine basis, employing trellis plots which are particularly effective for compact layout of multipanel plots. The main downside of trellis plots is probably that changing the default format can be a daunting and time-consuming task, involving nested lists and customized panel functions. The **scape** package addresses this problem by providing plot functions whose defaults are tailored for the stock assessment end user. Commonly needed graphical parameters are accessible as top-level function arguments.

By using AD Model Builder and R to analyze fisheries data, the modeller is holding two powerful tools, one for parameter estimation and the other for plotting and diagnosing the results. Given that the estimation model has already been implemented, the procedure could be:

Fit model

- Edit ADMB input file from editor/spreadsheet
- Run ADMB input file from editor/spreadsheet/command line
- Read ADMB output file into R (e.g. with **scape**)
- Plot and diagnose model fit (e.g. with **scape**)
- Repeat until model has been selected

MCMC analysis

- Run ADMB input file from editor/spreadsheet/command line
- Read ADMB output files into R (e.g. with **scape**)
- Plot MCMC results (e.g. with **scape**)
- Reassess whether another model needs to be fitted and analyzed

3.2 Interface between scape and external software

In the above procedure, ADMB results are stored and imported as text files, but the niche that **scape** occupies is quite generic. The estimation model and the MCMC simulations do not have to be implemented using AD Model Builder, and the interface between **scape** and the external software does not have to be text files. Two projects developed by fisheries scientists involve a tighter interface between R and the estimation model:

Beaufort `http://shrimp.ccfhrb.noaa.gov/~mprager/Rinter.html`
: Prager and Williams (2005) have implemented C++ and Fortran libraries to help implementing models that output R code instead of plain text files. Instead of writing an import function in R, it is sourced directly using the `dget` function.

FLR `http://flr-project.org`
: At last year's useR conference, Grosjean et al. (2004) introduced the FLR project, an ambitious bundle of R packages based on S4 classes. Its main goal is the evaluation of management strategies using simulations, but stock assessment is also within the scope of the project. All model input and output is managed within R, and computations can be performed either within R or by calling C++ or Fortran via dynamic link libraries (DLL).

Both the Beaufort and FLR projects are still in development and are not yet available on CRAN, but the main components have been tested and distributed on the project websites. Estimation models that output R code or can be called directly within R via DLL, would make the use of **scape** that much smoother. The ADMB manual describes how models can be compiled as DLL (Otter Research 2004, ch. 10).

The **scape** package is released and distributed at an early stage, so that its development can be shaped by user requests. I'm thankful to Allan Hicks, Jim Ianelli, Isaac Kaplan, and particularly Ian Stewart for fruitful discussions. I'd also like to thank Martyn Plummer, Deepayan Sarkar, and Greg Warnes for responding positively to my suggestions and code, in the development of their R packages.

References

Becker, R.A., W.S. Cleveland, and M.J. Shyu. 1996. The visual design and control of trellis display. Journal of Computational and Graphical Statistics 5:123–155. http://cm.bell-labs.com/stat/doc/trellis.jcgs.col.ps

Best, N.G., M.K. Cowles, and S.K. Vines. 1995. CODA: Convergence diagnosis and output analysis software for Gibbs sampling output, version 0.3. Cambridge: MRC Biostatistics Unit. http://www.mrc-bsu.cam.ac.uk/bugs/documentation/coda03/cdaman03.html.

Bull, B., R.I.C.C. Francis, A. Dunn, A. McKenzie, D.J. Gilbert, and M.H. Smith. 2004. CASAL (C++ algorithmic stock assessment laboratory) user manual. Version 2.06. NIWA Technical Report 126. http://www.niwa.co.nz/ncfa/tools/casal/

Gavaris, S. and J.N. Ianelli. 2002. Statistical issues in fisheries' stock assessments. Scandinavian Journal of Statistics 29:245–271.

Grosjean, P., L. Kell, D. Die, R. Scott, and J. De Oliveira. 2004. FLR, a framework for fisheries management in R using S4 classes. Presented at UseR symposium in Vienna. http://www.ci.tuwien.ac.at/Conferences/useR-2004/abstracts/Kell+Grosjean+Die+Scott.pdf

Helser, T.E., A.E. Punt, and R.D. Methot. 2004. A generalized linear mixed model analysis of a multi-vessel fishery resource survey. Fisheries Research 70:251–264.

Hilborn, R. 2003. The state of the art in stock assessment: Where we are and where we are going. Scientia Marina 67(S1):15–20.

Hilborn, R. and M. Liermann. 1998. Standing on the shoulders of giants: Learning from experience in fisheries. Reviews in Fish Biology and Fisheries 8:273–283.

Hilborn, R. and C.J. Walters. 1992. Quantitative fisheries stock assessment: Choice, dynamics and uncertainty. New York: Chapman and Hall.

Hilborn, R., M. Maunder, A. Parma, B. Ernst, J. Payne, and P. Starr. 2003. Coleraine: A generalized age-structured stock assessment model. User's manual version 2.0. University of Washington Report SAFS-UW-0116. http://fish.washington.edu/research/coleraine/coleraine.pdf.

Maunder, M. 2000. Software review of AD Model Builder 3.10. AFSCUS Newsletter 14(2):10–14. http://fisheries.org/cus/newsltrs/cus-news-f2000.pdf

Maunder, M.N. and A.D. Langley. 2004. Integrating the standardization of catch-per-unit-of-effort into stock assessment models: Testing a population dynamics model and using multiple data types. Fisheries Research 70:389–395.

Maunder, M.N. and A.E. Punt. 2004. Standardizing catch and effort data: A review of recent approaches. Fisheries Research 70:141–159.

Methot, R.D. 2000. Technical description of the stock synthesis assessment program. NOAA Technical Memorandum NMFS–NWFSC–43. http://www.nwfsc.noaa.gov/publications/techmemos/tm43/tm43.pdf

Meyer, R. and R.B. Millar. 1999. BUGS in Bayesian stock assessments. Canadian Journal of Fisheries and Aquatic Sciences 56:1078–1086.

Millar, R.B. and R. Meyer. 2000. Non-linear state space modelling of fisheries biomass dynamics by using Metropolis-Hastings within-Gibbs sampling. Applied Statistics 49:327–342.

Myers, R.A., K.G. Bowen, and N.J. Barrowman. 1999. Maximum reproductive rate of fish at low population sizes. Canadian Journal of Fisheries and Aquatic Sciences 56:2404–2419.

Otter Research. 2004. An introduction to AD Model Builder for use in nonlinear modeling and statistics. Version 7.0.1. Sidney, BC: Otter Research. http://otter-rsch.com/admodel.htm

Patterson, K., R.Cook, C. Darby, S. Gavaris, L. Kell, P. Lewy, B. Mesnil, A. Punt, V. Restrepo, D.W. Skagen, and G. Stefansson. 2001. Estimating uncertainty in fish stock assessment and forecasting. Fish and Fisheries 2:125–157.

Plummer, M., N. Best, K. Cowles, and K. Vines. 2005. The coda package. Vienna: R Foundation for Statistical Computing. http://cran.r-project.org/doc/packages/coda.pdf.

Prager, M.H. and E.H. Williams. 2005. A flexible, extensible interface between ADMB and R. Presented at NOAA Beaufort Laboratory. http://shrimp.ccfhrb.noaa.gov/~mprager/R-interface-MHP.pdf

Punt, A.E. and R. Hilborn. 1997. Fisheries stock assessment and decision analysis: The Bayesian approach. Reviews in Fish Biology and Fisheries 7:35–63.

Quinn, T.J., II and R.B. Deriso. 1999. Quantitative fish dynamics. New York: Oxford University Press.

Richards, L.J., J.T. Schnute, and N. Olsen. 1997. Visualizing catch-age analysis: A case study. Canadian Journal of Fisheries and Aquatic Sciences 54:1646–1658.

Salthaug, A. 2003. Dynamic age-length keys. Fishery Bulletin 101:451–456.

Sarkar, D. 2005. The lattice package. Vienna: R Foundation for Statistical Computing. http://cran.r-project.org/doc/packages/lattice.pdf.

Shepherd, J.G. 1999. Extended survivors analysis: An improved method for the analysis of catch-at-age data and abundance indices. ICES Journal of Marine Science 56:584–591.

Skaug, H.J. 2002. Automatic differentiation to facilitate maximum likelihood estimation in nonlinear random effects model. Journal of Computational and Graphical Statistics 11:458–470.

Smith, M.T. and J.T. Addison. 2003. Methods for stock assessment of crustacean fisheries. Fisheries Research 65:231–256.

Walters, C.J. and S.J.D. Martell. 2004. Fisheries ecology and management. Princeton: Princeton University Press.