1 The Vignette for the spm Model

Table of Contents

1.1 Introduction

Which Assessment to Apply

Which stock assessment method to apply to fisheries for data-poor to data-moderate species will depend upon what fisheries and biological data are available but also, importantly, on what management objectives need to be met within the jurisdiction in question. It may be the case that the fishery for a particular species is of sufficient size and value to warrant on-going monitoring and management towards some defined goal for the stock. In such a case the assessment used should obviously be capable of generating some notion of the current state of the fishery and indicate what management actions may be required to eventually achieve the agreed management goals. But some fisheries may be so minor that trying to actively manage them would be inefficient. Nevertheless, to meet the requirements of the Status of key Australian Fish Stocks (SAFS) one still requires some form of defensible stock assessment capable of determining whether the current level of fishing is sustainable.

1.2 Surplus Production Modelling = spm

Surplus production models are one of the simplest analytical methods available that provides for a full fish stock assessment of the population dynamics of the stock being examined. First described in the 1950s (Schaefer, 1954, 1957), modern versions with discrete dynamics are relatively simple to apply. This is partly because they pool the overall effects of recruitment, growth, and mortality (all the aspects of positive production) into a single production function. The stock is considered solely as undifferentiated biomass; that is, age and size structure, along with sexual and other differences, are ignored (one reason these models are also called biomass-dynamic models). Details of the equations used in the following analyses are provided in the Appendix of this vignette. You will also find details of the parameters and

other aspects in the help files for each of the functions (try *?spm* or *?simpspm*, or even *simpfox*). In brief, the model parameters are r the net population rate of increase (combined individual growth in weight, recruitment, and natural mortality), K the population carrying capacity or unfished biomass (B_0), which is not to be confused with B_{init} (sometimes, confusingly, also called B_0), which is only required if the index of relative abundance data (usually cpue) only becomes available after the fishery has been running for a few years and after the stock has been depleted to some extent. B_{init} is set equal to K if no initial depletion is assumed. If early catches are known to be small relative to the maximum catches taken (< 10 - 25% of maximum) then it may well be true that initial depletion is only minor and might not be distinguishable from unfished.

The minimum data requirements needed to estimate parameters for such models are time-series of an index of relative abundance and of associated catch data. The catch data can extend either end of the index data if it is available. In Australia the index of relative stock abundance is most often catch-per-unit-effort (cpue), but could be some fishery-independent abundance index (e.g., from trawl surveys, acoustic surveys), or both could be used. The analysis will permit the production of on-going management advice as well as a determination of stock status.

In this section we will describe the details of how to conduct a surplus production analysis, how to extract the results from that analysis as well as plot out illustrations of those results. Example code will be developed that a user can take and modify to suit their own needs.
A standard workflow might consist of:

1. read in time-series of catch and relative abundance data. It would also be possible to use the function *checkdata* to ensure all data columns required are present; you could also consider the functions **makespmdat** and **checkspmdat** for similar purposes, check their help files.
2. use a **ccf** analysis to determine whether the cpue data are likely to be informative and thus provide plausible results from the spm (or not!).
3. define/guess a set of initial parameters containing r and K, and optionally B_{init} = initial biomass - used if it is suspected that the fishery data starts after the stock has been somewhat depleted. The mean values from a catch-MSY analysis might work.
4. use **DisplayModel** to plot up the implications of the assumed initial parameter set for the dynamics. This is useful when searching for plausible initial parameters sets.
5. use **optim** or **fitSPM** to search for the optimum parameters once a potentially viable initial parameter set are input. See discussion.
6. use **DisplayModel** using the optimum parameters to illustrate the implications of the optimum model and its relative fit (especially using the residual plot).
7. ideally one should examine the robustness of the model fit by using multiple different initial parameter sets as starting points for the model fitting procedure. Example code for doing that is given in the vignette.
8. once satisfied with the robustness of the model fit use **spmphaseplot** to plot out the phase diagram so as to determine and illustrate the stock status visually .
9. use **bootspm** to characterize uncertainty in the model fit and outputs.
10. Document and defend any conclusions reached.

Two versions of the dynamics are currently available: the classical Schaefer model (Schaefer, 1954) and the Fox model (Fox, 1970), both are described in Haddon (2011). Prager (1994) provides many additional forms of analysis that are possible using surplus production models

and practical implementations are also provided in Haddon (2011). See the model equations in the appendix for more details of each model.

1.3 Application of the spm assessment

The minimum data requirements for a surplus production analysis is a time series of catches and a time series of an index of relative abundance (cpue). The years of catch information can be longer than the years of cpue data. We will use the *dataspm* data set built into the package to illustrate the use of the *spm* function. The *checkdata* function indicates it would be possible to apply the catch-MSY, teh spm, and the aspm analyses to this data.

```
# library(simpleSA)  # include the library before starting analysis
data(dataspm)
fish <- dataspm$fish
# check data.frame has all required columns; not usually required; use hea
d(fish,10)
checkdata(dataspm)
##                     Method Possible
## catch-MSY             TRUE     TRUE
## spm                   TRUE     TRUE
## aspm                  TRUE     TRUE
## catch-curves catch-curves    FALSE
```

Are the CPUE data informative?

The first step is to try to discover whether or not the available index data provide more information than the catches alone. With a developed fishery if the fishing, that is the catches, do indeed influence the stock dynamics, which would in turn affect the cpue, then we would expect that if catches increased the catch rates would eventually respond by declining and also if catches declined then the cpue would, in turn, eventually increase. Such a time-lagged negative correlation is something we can look for statistically if we have sufficiently long time-series of catches and associated cpue. To determine whether there are any signficant negative correlations of cpue with catches we need to conduct a sequence of correlation analyses on the original data (timelag=0), then lag the cpue data backwards by one year and repeat the correlation analysis (with n-1 years this time, lag=-1), and repeat that for an array of negative time lags. Rather than do this manually it is possible to use a built in R function *ccf*. To simplify things further there is a wrapper function in simpleSA called *getlag*. This uses the *ccf* function to run the analysis, which can also generate a plot immediately or its output can be used to produce a more visually appealing plot for inclusion in a report (check out *str(ans)*).

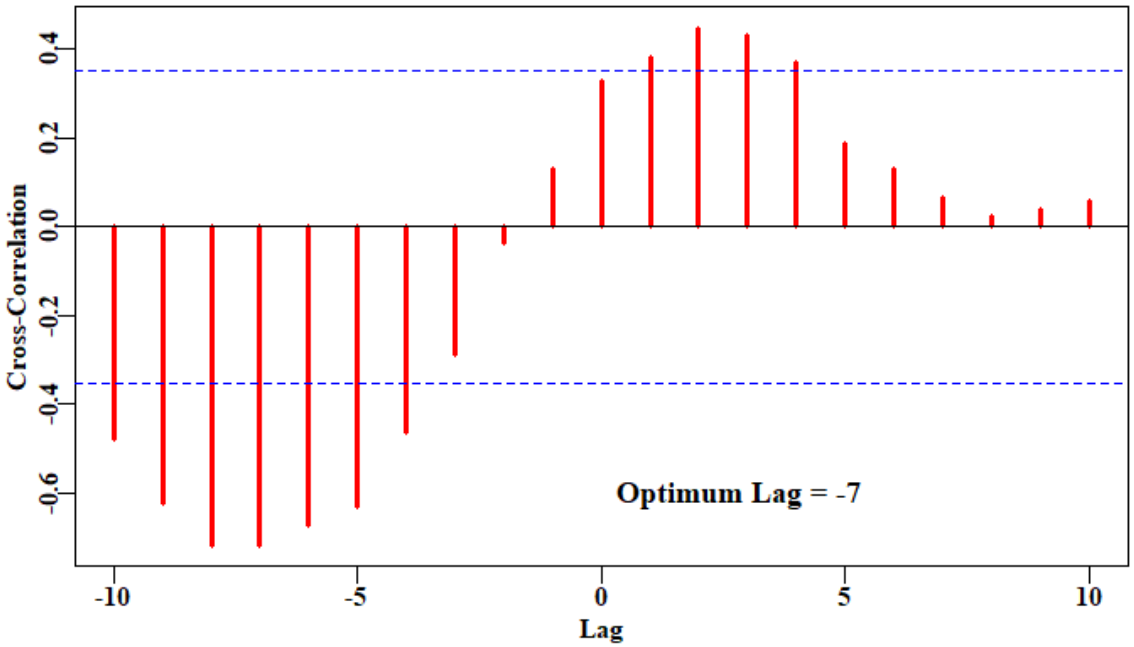

Figure 1. A plot of the *ccf* function output when applied to the fishery data within the built in *dataspm* data set. The strong negative correlation at lags of -7 and -8 suggest an spm or aspm analysis will likely generate plausible results, although the relatively strong correlation at a lag of 0, may weaken any valid relationship between the catches and subsequent cpue values.

It should not be forgotten that with catches and cpue, we have catches on both sides of the correlation analysis so we should expect to see correlations. If effort were relatively constant however then if availability increased and catchability remained the same then catches would increase and so would cpue, thus the correlation at lag = 0 would be high. But despite such complications it still appears to be the case that if no significant correlations at negative time lags occur then valid spm or aspm model fits to the cpue data do not seem to occur.

1.4 Fitting the spm Model

The first step is to start guessing initial parameter sets until an approximate solution presents itself. This eases the way for the minimization of the negative log-likelihood (-verLL) used to fit the model to the data. If the initial guesses are too far from the final solution then the numerical methods sued to conduct the minimization may fail to converge on a viable solution. If the stock is thought to have been depleted before the collection of data began then three parameters are required, otherwise only two are needed. In the example below these parameters are $r = 0.16$, $K = 6700$, and $Binit = 3500$ (which implies an initial depletion of 3500/6700, which is ~52%). When making these initial guesses it is helpful to both plot the implied predicted trajectory of cpue against the observed cpue, and include the estimation of the negative log-likelihood. In that way, if the guessed model fit to the data begins to improve it is easily detected by the -veLL getting smaller (more negative in this case).

The plot in **Figure 2** merely represents the first guess at the parameters so it is not surprising that the model fit to data is not particularly impressive. Guessing a set of parameters and displaying their implications is an important first step to get the predicted cpue even approximating the observed cpue. Only once such an approximation is produced should you think about trying to fit the model to the data formally. Model fitting may produce sensible results before an approximate solution is found but often it does not. It is possible to use the

catch-MSY method to generate first guesses at the *r* and *K* parameters, but producing initial values for *Binit*, if it is to be used, is currently down to trial and error.

```
# display the dynamics for a set of guessed initial parameters
pars <- c(0.16,6700,3500)  # r, K, and Binit
negLL(pars,fish,simpspm)
## [1] -2.862565
ans <- displayModel(pars,fish,schaefer=TRUE,target=0.48,addrmse=TRUE)
```

Rules of thumb can be used when attempting to find suitable starting points. A possible approximation for the unfished biomass (*K* parameter) can be obtained by multiplying the maximum catch by 10 to 15. As in the catch-MSY method, if the initial catches in the available data are a reasonable proportion of the maximum catch then the stock can be assumed to have been somewhat depleted prior to data collection. A value between 0.5 to 0.7 times the selected *K* value could be used for a *Binit* value. One might then search for an *r* value perhaps starting at something like 0.3. Remember that the *r* parameter as a combination of both the reproductive rate and the natural mortality rate is a net population growth rate not only the reproductive rate. This means it may be lower than a simple consideration of the species' life history might suggest. A plausible set of parameters would be where the implied predicted cpue trajectory even roughy approximates the observed trajectory. The next step after finding a plausible starting point is to attempt to use numerical methods to fit the data to the model.

The production curves are an expression of the *r* - *K* parameters where, for the Schaefer model, the MSY is assumed to occur at half the unfished biomass (*K*; see the model equations appendix). Using the production curve it will be possible to estimate the predicted catch at B_{Target} and B_{Limit}, and from those it will be possible to estimate the harvest rates at the target and limit reference points. Two of the potential input parameters to the *displayModel* function are the *limit* and *target*, which default to 0.2 and 0.48, obviously these can be set to whatever obtains in the jurisdiction involved.

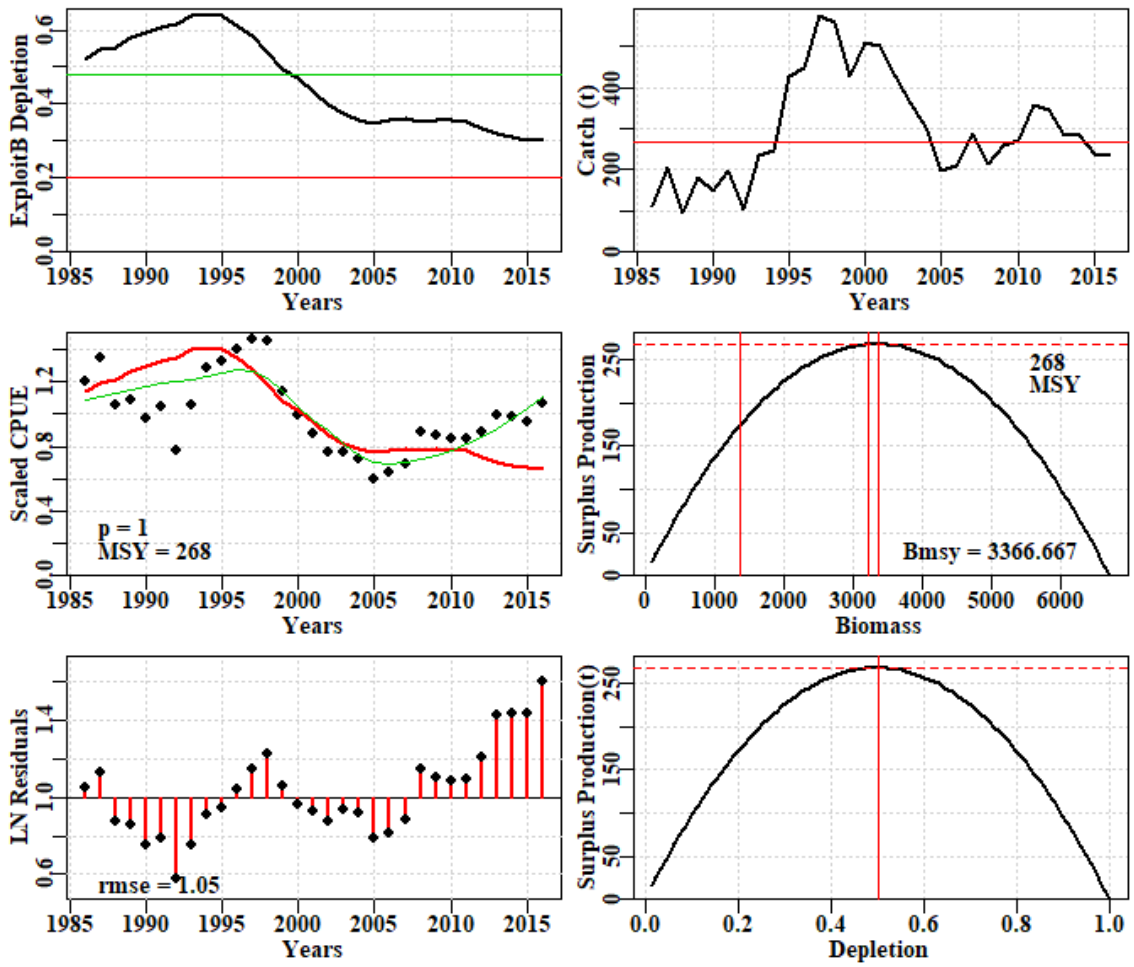

Figure 2. A summary plot depicting the fit of the model using the guessed input parameters. The plots include the predicted depletion and catch through time. Then the fit to cpue, where the green line is a loess fit to the observed cpue and the red line is the fit using the parameters. The surplus production curve is given twice, once with biomass and once with depletion on the x-axis, finally the log-normal residuals between the fit and the data are illustrated. These initial parameters have not been fitted.

1.5 Fitting the Model to the Data

There are three functions available for working with the dynamics of surplus production models. The most comprehensive in its output is *spm*, which generates the full dynamics and returns a matrix of all values (try *out <- spm(inp=pars, indat=fish, schaefer = TRUE)*. The default Schaefer model dynamics is defined by the *schaefer = TRUE* parameter value, to use the Fox model the option *schaefer = FALSE* should be used. The model fitting only compares the observed cpue with the predicted cpue and does not require the full dynamics to be produced, so, in order to speed up the iterative process of numerically searching for a model fit, we also have a simplified *simpspm* function that only outputs the predicted cpue. This function also has a *schaefer* parameter to determine whether to use a Schaefer or the Fox model. The *simpspm* function is used in the model fitting whereas, if you inspect the code for *displayModel* you will find that it uses *spm*. In the following we will use *optim* which is a solver or minimizer that comes as part of the base R installation. Check out *?optim* or *formals(optim)*, or more thoroughly you could read the *Optimization* task view at https://cran.r-project.org/. The code below is a template that should work for you. The control list is not always needed but it usually helps to keep the solver stable. If you check the

code for *negLL* by typing that in the R console without brackets, you will see there is code (*na.rm=TRUE*)to exclude records where there may be catch data but no observed cpue data. Ideally one has data for both in each year, which generally leads to more stable outcomes, but in reality this does not always occur.

Instead of using *optim* or one of the other non-linear minimizers in R it is possible to use *fitSPM*, which is a wrapper within **simpleSA** that contains two runs through the optim fitting routine (see later).

```
pars <- c(0.16,6700,3500) # r, K, and Binit, implies a depletion of ~0.52
cat("initial -ve log-likelihood ",negLL(pars,fish,simpspm),"\n")
## initial -ve log-likelihood  -2.862565
bestSP <- optim(par=pars,fn=negLL,callfun=simpspm,indat=fish,
                control=list(maxit = 1000, parscale = c(1,1000,1000)))
outoptim(bestSP) # outoptim just prints the results more compactly.
##
## $par         :  0.2424071 5173.363 2845.831
## $value       :  -12.12879
## $counts      :  192 NA   iterations, gradient
## $convergence :  0
## $message     :
# read the help files to understand what each component means.
# Schaefer model dynamics is the default so no need for schaefer=TRUE
```

Now plot up the optimum solution, in this case using the Schaefer model. Note in this case that the final depletion is close to the approximate target green line defined by the loess smoother fit to the observed data (a result of the *addrmse=TRUE*). Note also that the years in which catches were greater than the MSY (the red line in the top right plot) were the years in which the stock declined.

There are some patterns in the residuals which may suggest there are other factors at play on the fishery dynamics, but it is clear there are some changes occurring which the model is not sufficiently flexible to enable a fit. Such simple surplus production models ignore good and bad recruitment years and so some deviations from observed are to be expected. Whether this is a sufficient reason for such patterned deviations would require extra independent evidence to be determined.

The object output from `displayModel` has a number of parts including the dynamics, various statistics relating to the model fit, the model parameters, and the production curve. These are used by other functions to summarize and plot the outcomes. However, first it would be best to be confident that one has a unique and optimum model fit.

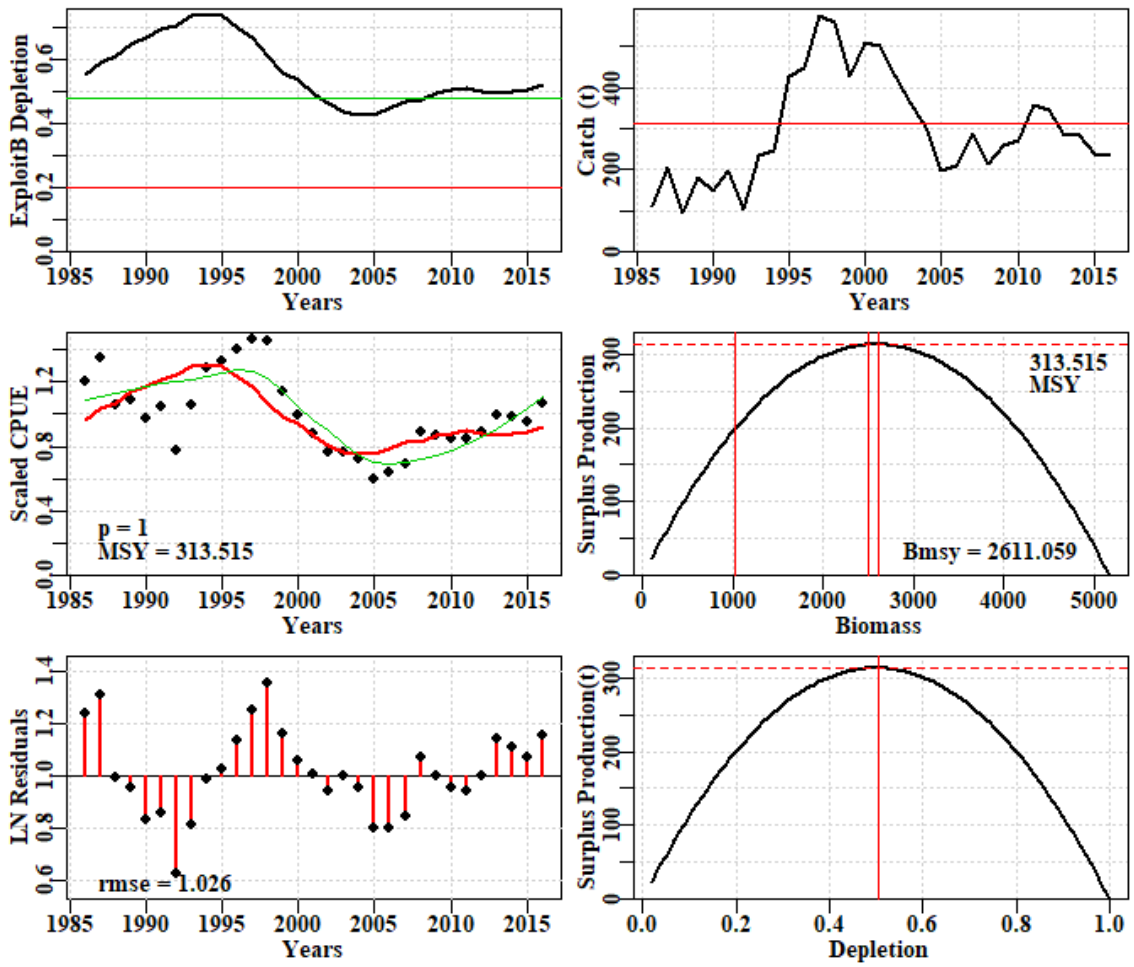

Figure 3. A summary plot depicting the fit of the optimum parameters. The plots include the predicted depletion and catch through time. Then the fit to cpue, where the green line is a loess fit to the data and the red line is the fit using the parameters. The surplus production curve is given with biomass and depletion on the x-axes (note the symmetry of the production curve), finally the residuals between the fit and the data are illustrated.

```
str(ans)
## List of 11
##  $ Dynamics :List of 5
##   ..$ outmat    : num [1:31, 1:8] 2846 3043 3141 3344 3448 ...
##   .. ..- attr(*, "dimnames")=List of 2
##   .. .. ..$ : chr [1:31] "1986" "1987" "1988" "1989" ...
##   .. .. ..$ : chr [1:8] "ModelB" "Catch" "CPUE" "PredCE" ...
##   ..$ q         : num 0.00034
##   ..$ msy       : num 314
##   ..$ parameters: num [1:3] 0.242 5173.363 2845.831
##   ..$ sumout    : Named num [1:9] 0.242 5173.363 2845.831 313.515 1 ...
##   .. ..- attr(*, "names")= chr [1:9] "r" "K" "B0" "msy" ...
##  $ BiomProd : num [1:100, 1:2] 100 151 202 254 305 ...
##   ..- attr(*, "dimnames")=List of 2
##   .. ..$ : NULL
##   .. ..$ : chr [1:2] "x" "y"
##  $ rmseresid: num 1.03
##  $ MSY      : num 313
##  $ Bmsy     : num 2611
```

```
##  $ Dmsy     : num 0.505
##  $ Blim     : num 1022
##  $ Btarg    : num 2509
##  $ Ctarg    : num 313
##  $ Dcurr    : Named num 0.522
##   ..- attr(*, "names")= chr "2016"
##  $ rmse     : num 0.137
```

The model fitted was the Schaefer model so it may be worth considering the outcome had the alternative Fox model been used. This can be implemented by making simple changes to the same functions as before. In this particular case the initial values that enabled a solution for the Schaefer model also work for the Fox model, which is not always the case. Note the fit is slightly better but in fact the predicted cpue trajectories barely differ. Note also the use of magnitude to estimate the scale of each parameter and hence simplify the use of the *parscale* option within the control list.

```
pars <- c(0.16,6700,3500) # r, K, and Binit, implies a depletion of ~0.52
cat("initial -ve log-likelihood ",negLL(pars,fish,simpspm,schaefer=FALSE),
"\n")
## initial -ve log-likelihood  -5.998584
parscl <- magnitude(pars)
bestSP <- optim(par=pars,fn=negLL,callfun=simpspm,indat=fish,schaefer=FALS
E,
                control=list(maxit = 1000, parscale = parscl))
outoptim(bestSP   )
##
## $par         :  0.1382109 6129.655 2757.171
## $value       :  -12.35283
## $counts      :  160 NA   iterations, gradient
## $convergence :  0
## $message     :
# Fox model dynamics is set by schaefer=FALSE
```

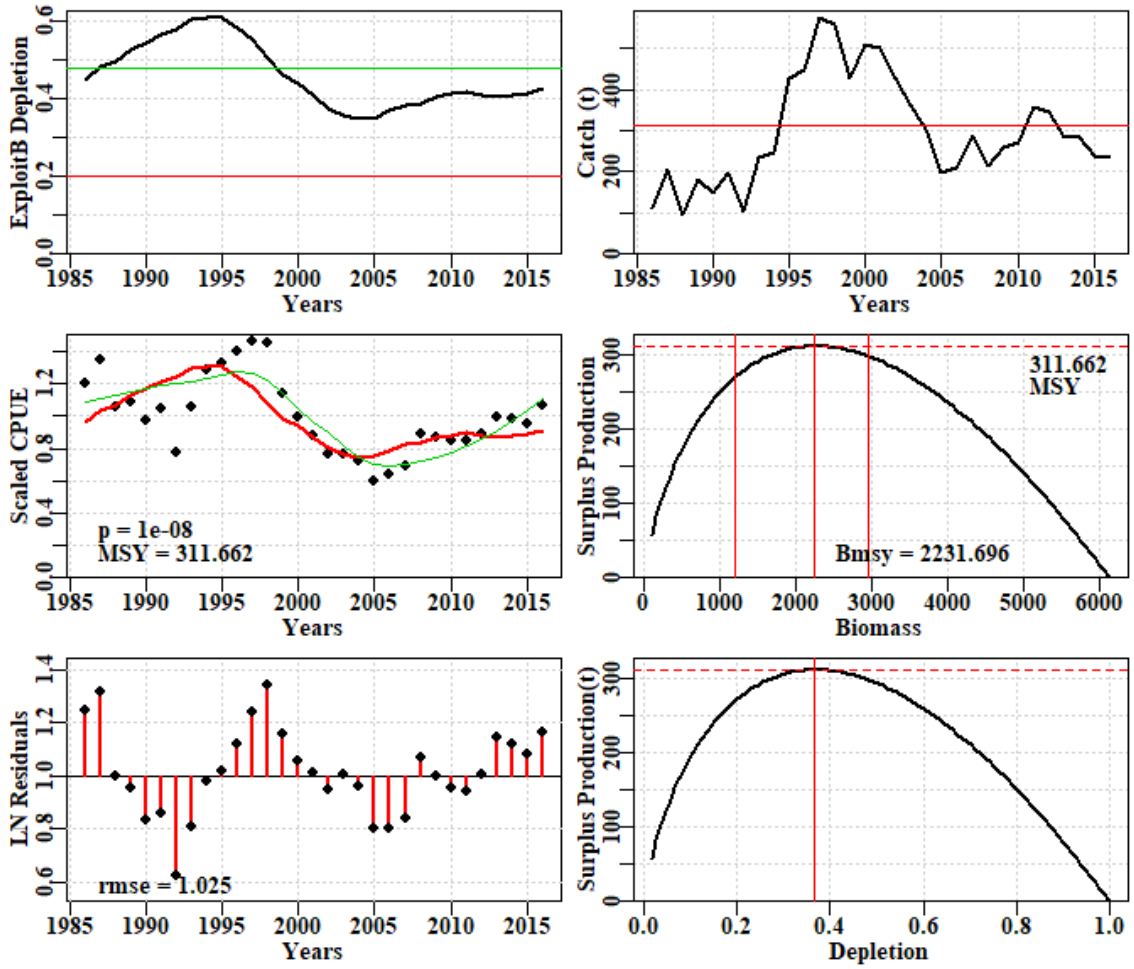

Figure 4. A summary plot depicting the fit of the optimum parameters. The plots include the predicted depletion and catch through time. Then the fit to cpue, where the green line is a loess fit to the data and the red line is the fit using the parameters. The surplus production curve is given with biomass and depletion on the x-axes (note the asymmetry of the production curve), finally the residuals between the fit and the data are illustrated.

```
str(ans)
## List of 11
##  $ Dynamics :List of 5
##   ..$ outmat    : num [1:31, 1:8] 2757 2949 3041 3240 3342 ...
##   .. ..- attr(*, "dimnames")=List of 2
##   .. .. ..$ : chr [1:31] "1986" "1987" "1988" "1989" ...
##   .. .. ..$ : chr [1:8] "ModelB" "Catch" "CPUE" "PredCE" ...
##   ..$ q         : num 0.00035
##   ..$ msy       : num 312
##   ..$ parameters: num [1:3] 0.138 6129.655 2757.171
##   ..$ sumout    : Named num [1:9] 1.38e-01 6.13e+03 2.76e+03 3.12e+02 1
.00e-08 ...
##   .. ..- attr(*, "names")= chr [1:9] "r" "K" "B0" "msy" ...
##  $ BiomProd : num [1:100, 1:2] 100 161 222 283 344 ...
##   ..- attr(*, "dimnames")=List of 2
##   .. ..$ : NULL
##   .. ..$ : chr [1:2] "x" "y"
##  $ rmseresid: num 1.03
##  $ MSY      : num 312
```

```
##  $ Bmsy     : num 2232
##  $ Dmsy     : num 0.364
##  $ Blim     : num 1196
##  $ Btarg    : num 2963
##  $ Ctarg    : num 298
##  $ Dcurr    : Named num 0.425
##   ..- attr(*, "names")= chr "2016"
##  $ rmse     : num 0.137
```

1.6 Is the Solution the Global Optimum?

In the ideal world you should always try solving from multiple starting points. A general function is under development but in the meantime you could try using something like this. Here we have used the fitSPM function, which runs through the fitting process twice in case the first run through failed to find the best solution (try *?fitSPM* or just *fitSPM* with no brackets):

```
  pars <- c(0.16,6700,3500) # the original starting point
  origpar <- pars
  N <-  20       # the number of random starting points to try
  scaler <- 10   # how variable; smaller = more variable
  # define a matrix for the results
  columns <- c("ir","iK","iB0","iLike","r","K","Binit","-veLL","MSY","Iters")
  results <- matrix(0,nrow=N,ncol=length(columns),dimnames=list(1:N,columns))
  pars <- cbind(rnorm(N,mean=origpar[1],sd=origpar[1]/scaler),
                rnorm(N,mean=origpar[2],sd=origpar[2]/scaler),
                rnorm(N,mean=origpar[3],sd=origpar[3]/scaler))
  # this randomness ignores the strong correlation between r and K
  for (i in 1:N) {
    bestSP <- fitSPM(pars[i,],fish,schaefer=TRUE)
    opar <- bestSP$par
    msy <- opar[1]*opar[2]/4
    origLL <- negLL(pars[i,],fish,simpspm)
    results[i,] <- c(pars[i,],origLL,bestSP$par,bestSP$value,msy,bestSP$counts[1])
  }
  cat("\n\n\n")
kable(results[order(results[,"-veLL"]),],digits=c(0,3,3,3,3,3,3,3,3,3,0))
```

By repeating that analysis multiple times (more than 20, and try changing the *scaler* variable) you will eventually find solutions that fall far from the most common optimum. This is valuable in demonstrating that the initial starting values can be influential on the final solution and that one should not immediately trust a solution found numerically until it has been tested and pushed around to determine how robust it is to initial conditions. Notice that any outlying values, if they occur, are not common, so the use of the median will usually exclude them from an estimate of the central tendency for the parameters and stock properties such as MSY. We can use the median values of the optimum estimates when we are plotting up a phase plot of biomass against fishing mortality to illustrate the stock status. Had we used a single *optim* run instead of the two found inside *fitSPM* more failures to find the optimum are likely to have occurred.

	ir	iK	iB0	iLike	r	K	Binit	-veLL	MSY	Iters
7	0	6707.229	3302.347	-3.787	0.242	5173.504	2845.948	-12.129	313.512	298
3	0	6299.441	3899.234	5.853	0.242	5173.394	2846.055	-12.129	313.517	138
6	0	7621.682	3482.378	-4.289	0.242	5173.701	2846.085	-12.129	313.512	216
19	0	6676.613	3447.190	16.344	0.242	5173.335	2845.830	-12.129	313.517	140
10	0	6504.165	3879.736	-7.612	0.242	5173.189	2845.937	-12.129	313.519	160
13	0	6555.213	3519.619	-11.182	0.242	5173.902	2846.216	-12.129	313.510	152
14	0	6049.977	3730.949	-11.350	0.242	5173.331	2845.803	-12.129	313.518	122
5	0	6844.054	4442.201	-10.352	0.242	5173.853	2846.376	-12.129	313.510	196
4	0	7049.346	3479.881	-10.789	0.242	5173.849	2846.329	-12.129	313.509	236
9	0	6730.954	3336.066	9.109	0.242	5173.713	2846.109	-12.129	313.515	160
16	0	7350.059	3275.202	-9.231	0.242	5173.477	2846.287	-12.129	313.514	198
18	0	6980.252	3540.447	-10.290	0.242	5173.129	2845.611	-12.129	313.516	180
20	0	7433.094	3499.097	-11.116	0.242	5173.709	2846.024	-12.129	313.515	142
1	0	5226.180	3587.453	60.753	0.242	5173.015	2845.980	-12.129	313.524	114
12	0	5868.446	3492.770	22.569	0.242	5173.652	2845.947	-12.129	313.517	196
15	0	6153.620	3828.109	55.555	0.242	5173.099	2845.786	-12.129	313.524	186
8	0	6728.774	3050.008	52.117	0.242	5172.546	2845.551	-12.129	313.523	124
17	0	5364.663	3703.252	57.705	0.242	5173.370	2846.353	-12.129	313.524	112
11	0	6629.039	2943.251	54.507	0.242	5173.136	2845.771	-12.129	313.528	126
2	0	6907.240	3493.210	-9.181	0.242	5174.853	2848.677	-12.129	313.500	186

```
cat("\n\n")
kable(apply(results,2,range),digits=c(0,3,3,3,3,3,3,3,3,3,0))
```

ir	iK	iB0	iLike	r	K	Binit	-veLL	MSY	Iters
0	5226.180	2943.251	-11.350	0.242	5172.546	2845.551	-12.129	313.500	112
0	7621.682	4442.201	60.753	0.242	5174.853	2848.677	-12.129	313.528	298

```
cat("\n\n")
round(apply(results,2,median),3)
```

```
ir       iK      iB0    iLike     r     K       Binit    -veLL      MSY    Iters
0.161 6691.921 3496.153 -4.038 0.242 5173.435 2846.002 -12.129  313.516     160
```

1.7 Produce a Phase Plot

In the SAFS process determining the stock status is an important component. This is simply achieved once you have run the function *displayModel* using the optimum parameters. First fit an optimum model (here we use the median estimates of a set of runs like those produced in the previous example). Once we have applied the *displayModel* function using the optimum parameter estimates we can produce the phase plot.

```
pars <- c(0.2424, 5173.5972, 2846.0953) # r, K, and Binit, median values
bestSP <- fitSPM(pars,fish,schaefer=TRUE)
ans <- displayModel(bestSP$par,fish,schaefer=TRUE,addrmse=TRUE)
```

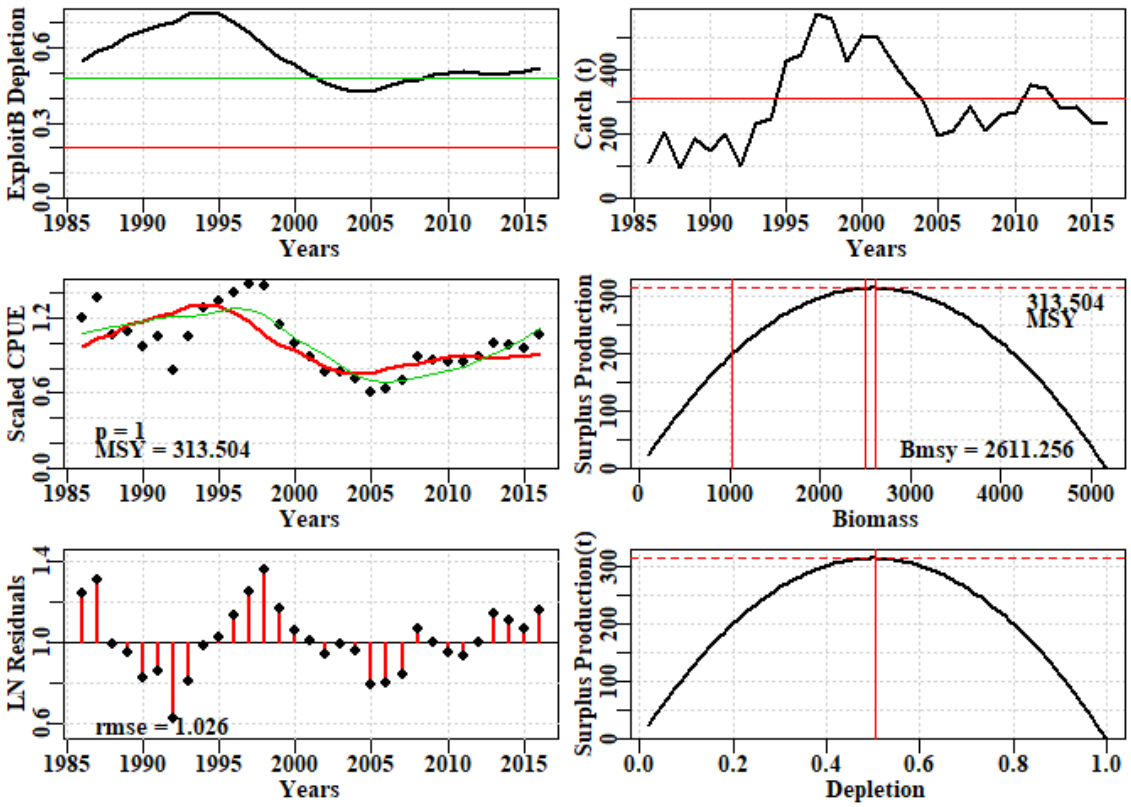

Figure 5. A summary plot depicting the fit after using the median optimum parameters from the repeated starting points depicted in the analysis above. The plots include the predicted depletion and catch through time. Then the fit to cpue, where the green line is a loess fit to the data and the red line is the fit using the parameters. The surplus production curve is given with biomass and depletion on the x-axes, finally the residuals between the fit and the data are illustrated.

The object output by *displayModel* is used to generate a phase plot of the trajectory of stock status implied by the surplus production model's description of the fishery's dynamics.

```
# plotprep(width=7,height=5.5) # to avoid using the RStudio plot window
spmphaseplot(ans,fnt=7)
```

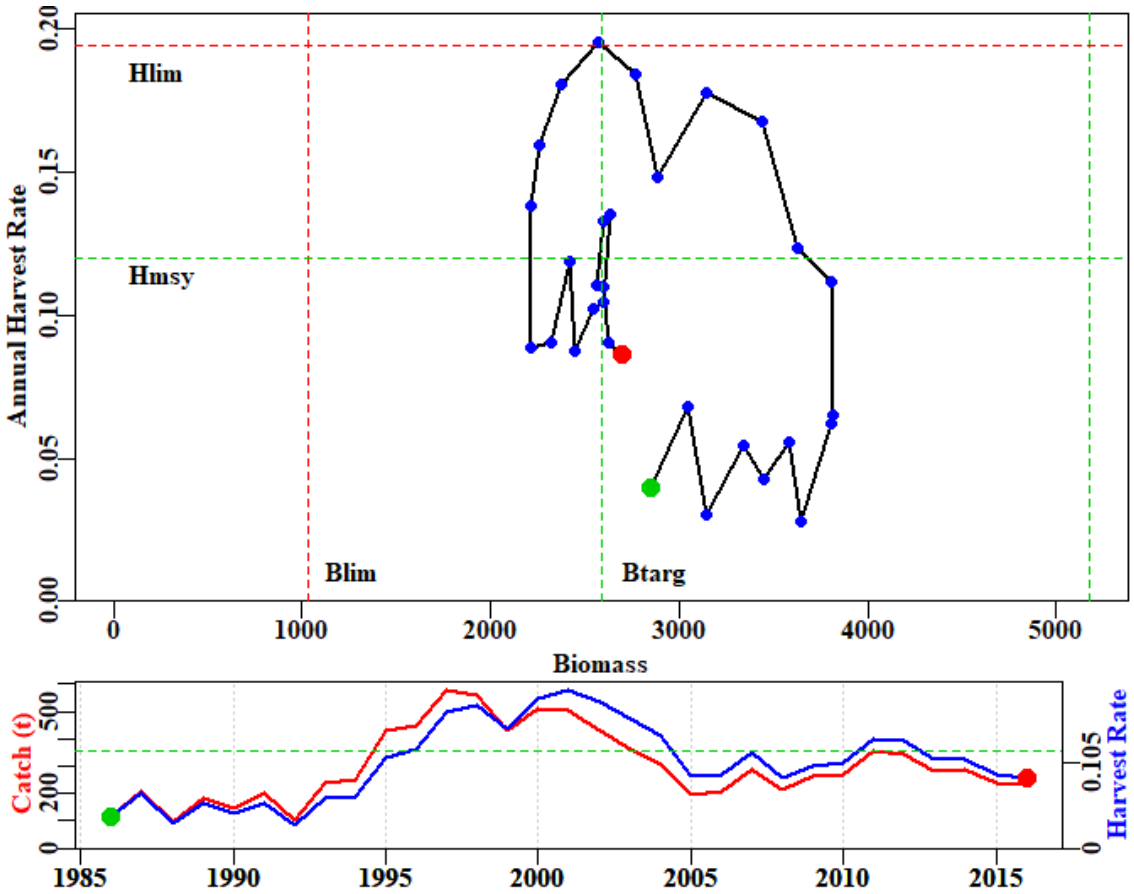

Figure 6. The top plot is a phase plot of the predicted stock biomass versus the predicted annual harvest rate for the optimum model fit. The green dot is the starting year and the red dot the final year. the limit reference points are included and tentative target reference points also to aid interpretation. The catch history and related harvest rates are illustrated in the lower plot also with an aim to aid interpretation of the status trajectory (in which time is only implied).

In the case illustrated the current status is both above the limit and the target biomass reference points, and below the limit and target harvest rate reference points and so can be safely concluded to be sustainable. A more detailed summary would conclude no over-fishing and not over-fished. The catch history indicates that the first eight years of catches led to harvest rates sufficiently low as to allow the stock to increase in size but that as catches increased from 1994 onwards they quickly rose above the harvest rate that would, in theory, maintain the stock at its target and that led to the stock declining over the following 10 years, after which catches declined again so that the stock slowly recovered back up to the target biomass over the next 10 years, with occasional increases in harvest rate up close to the target rate.

1.8 Generate Bootstrap Confidence Intervals

Every fishery analysis is invariably uncertain and an effort should always be made to include at least an indication of that uncertainty into any report of a stock assessment. One such characterization included in **simpleSA** for the *spm* and *aspm* models is the generation of percentile confidence intervals around parameters and model outputs (MSY, etc) by taking bootstrap samples of the log-normal residuals associated with the cpue and using those to generate new bootstrap cpue samples with which to replace the original cpue time-series

(Haddon, 2011). Each time such a bootstrap sample is made the model is re-fit and the solutions obtained stored for further analysis to conduct such an analysis one uses the *bootspm* function. Once we have found suitable starting parameters we use the *fitSPM* function, which uses optim twice sequentially with a negative likelihood approach to obtain an optimum fit.

```
data(dataspm)
fish <- dataspm$fish
colnames(fish) <- tolower(colnames(fish))
pars <- c(r=0.25,K=5500,Binit=2900)
ans <- fitSPM(pars,fish,schaefer=TRUE,maxiter=1000) #Schaefer version
answer <- displayModel(ans$par,fish,schaefer=TRUE,addrmse=TRUE)
```

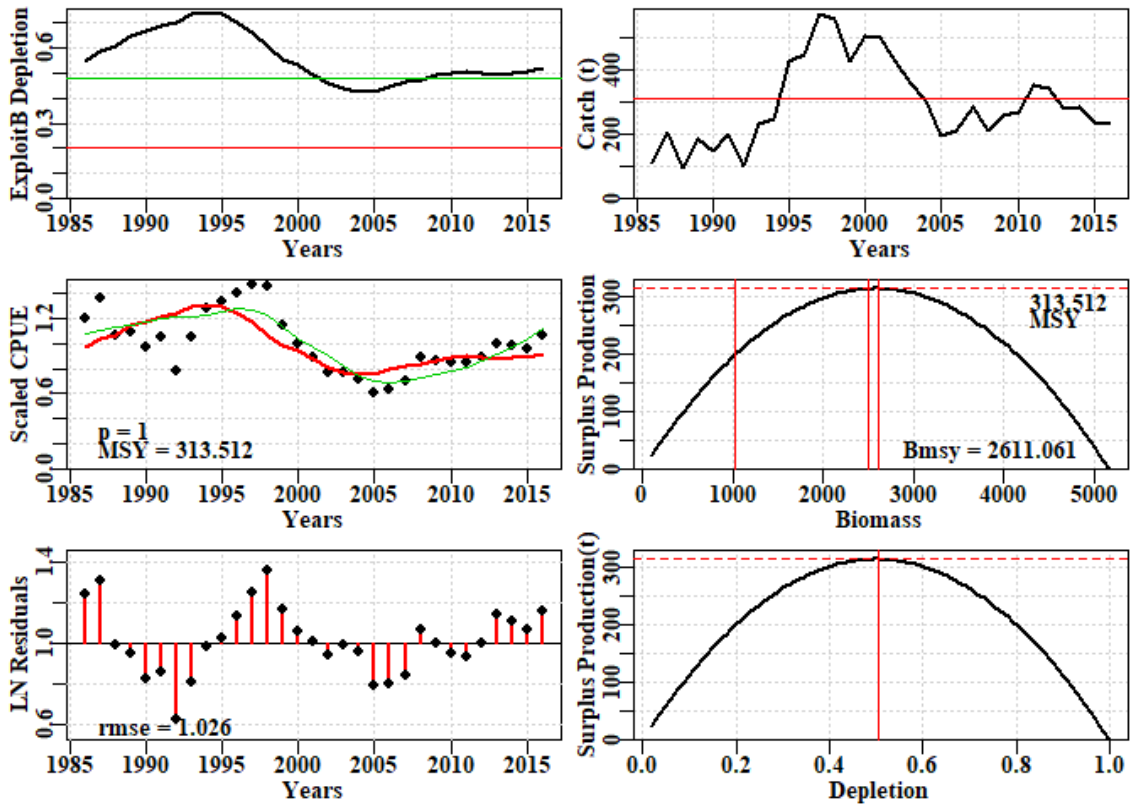

Figure 7. A summary plot depicting the fit of the optimum parameters to the dataspm dataset. The residuals between the fit and the cpue data are illustrated at the bottom left. These are what are bootstrapped and each sample multiplied by the optimum predicted cpue time-series to obtain each bootstrap cpue time-series.

Once we have an optimum fit we can proceed to conduct a bootstrap analysis. Here we are only running 100 replicates for speed but you would usually run at least 1000 even though that might take a few minutes to complete.

```
reps <- 100       # this might take ~60 seconds, be patient
startime <- Sys.time()  # how long will this take
boots <- bootspm(ans$par,fishery=fish,iter=reps,schaefer=TRUE)
print(Sys.time() - startime)
## Time difference of 5.285956 secs
str(boots)
```

```
## List of 3
##  $ dynam       : num [1:100, 1:31, 1:5] 2846 2456 2085 2181 2197 ...
##   ..- attr(*, "dimnames")=List of 3
##   .. ..$ : chr [1:100] "1" "2" "3" "4" ...
##   .. ..$ : chr [1:31] "1986" "1987" "1988" "1989" ...
##   .. ..$ : chr [1:5] "ModelB" "BootCE" "PredCE" "Depletion" ...
##  $ bootpar     : num [1:100, 1:6] 0.242 0.207 0.354 0.375 0.262 ...
##   ..- attr(*, "dimnames")=List of 2
##   .. ..$ : chr [1:100] "1" "2" "3" "4" ...
##   .. ..$ : chr [1:6] "r" "K" "Binit" "MSY" ...
##  $ negativepars: num 0
```

The output contains the dynamics of each run with the predicted model biomass, each bootstrap cpue sample, the predicted cpue for each bootstrap sample, the depletion time-series, and the annual harvest rate time-series. Each of these can be used to illustrate and summarize the outcomes and uncertainty within the analysis. Given the relatively large residuals in **Figure 7** one might expect a relatively high degree of uncertainty.

```
dynam <- boots$dynam
bootpar <- boots$bootpar
rows <- colnames(bootpar)
columns <- c(c(0.025,0.05,0.5,0.95,0.975),"Mean")
bootCI <- matrix(NA,nrow=length(rows),ncol=length(columns),
                 dimnames=list(rows,columns))
for (i in 1:length(rows)) { # i=1
   tmp <- sort(bootpar[,i])
   qtil <-  quantile(tmp,probs=c(0.025,0.05,0.5,0.95,0.975),na.rm=TRUE)
   bootCI[i,] <- c(qtil,mean(tmp,na.rm=TRUE))
}
kable(bootCI,digits=c(3,3,3,3,3,3))
```

	0.025	0.05	0.5	0.95	0.975	Mean
r	0.143	0.158	0.260	0.364	0.374	0.257
K	3601.107	3730.491	4905.762	7799.845	8879.580	5287.554
Binit	1637.580	1809.071	2653.912	4384.197	5201.982	2843.718
MSY	286.937	292.590	318.713	344.801	347.655	318.981
Depl	0.343	0.367	0.527	0.637	0.644	0.518
Harv	0.054	0.062	0.093	0.119	0.125	0.091

Such percentile confidence intervals can be visualized using histograms and including the respective selected percentile CI.

```
par(mfrow=c(3,2),mai=c(0.45,0.45,0.15,0.05),oma=c(0.0,0,0.0,0.0))
par(cex=0.85, mgp=c(1.35,0.35,0), font.axis=7,font=7,font.lab=7)
hist(bootpar[,"r"],breaks=25,col=2,main="",xlab="r")
abline(v=c(bootCI["r",c(2,3,4,6)]),col=c(3,3,3,4),lwd=c(1,2,1,2))
hist(bootpar[,"K"],breaks=25,col=2,main="",xlab="K")
abline(v=c(bootCI["K",c(2,3,4,6)]),col=c(3,3,3,4),lwd=c(1,2,1,2))
hist(bootpar[,"Binit"],breaks=25,col=2,main="",xlab="Binit")
abline(v=c(bootCI["Binit",c(2,3,4,6)]),col=c(3,3,3,4),lwd=c(1,2,1,2))
hist(bootpar[,"MSY"],breaks=25,col=2,main="",xlab="MSY")
```

```
abline(v=c(bootCI["MSY",c(2,3,4,6)]),col=c(3,3,3,4),lwd=c(1,2,1,2))
hist(bootpar[,"Depl"],breaks=25,col=2,main="",xlab="Final Depletion")
abline(v=c(bootCI["Depl",c(2,3,4,6)]),col=c(3,3,3,4),lwd=c(1,2,1,2))
hist(bootpar[,"Harv"],breaks=25,col=2,main="",xlab="Final Harvest Rate")
abline(v=c(bootCI["Harv",c(2,3,4,6)]),col=c(3,3,3,4),lwd=c(1,2,1,2))
```

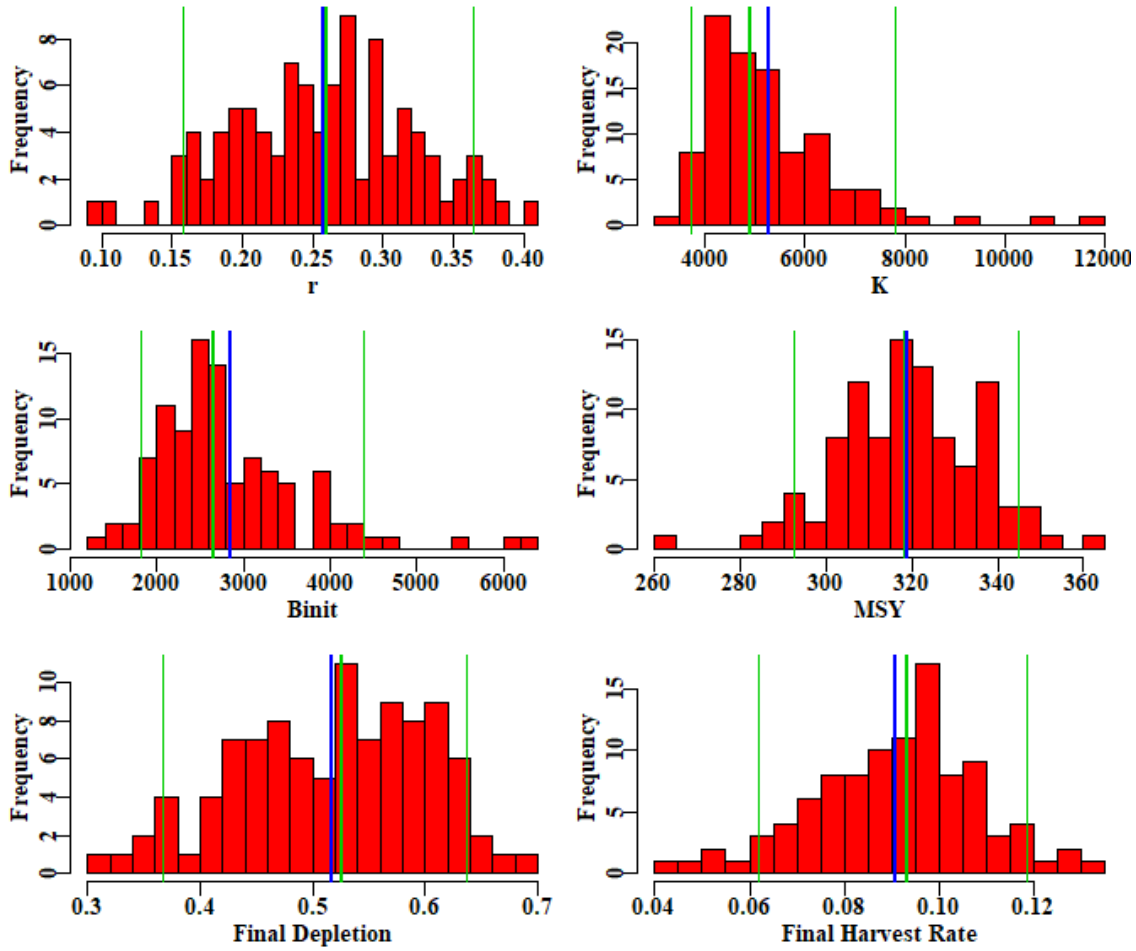

Figure 8. The bootstrap replicates from the optimum spm fit to the dataspm data set. The vertical green lines, in each case, are the median and 90th percentile confidence intervals and the blue lines are the mean values. There is little evidence of bias with the 'r' and 'MSY', estimates, although both the median 'K' and 'Binit' values appear somewhat biased low, while the depletion and harvest rates appear slightly biased high.

Of course, with only 100 replicates in such a variable analysis these results remain only very approximate. One would expect 1000 replicates would provide for a smoother response and more representative confidence bounds (100 were used simply to demonstrate the principle within a short time; this may be sped up later using C++). Note that the confidence bounds are not necessarily symmetrical around either the mean or the median estimates. Notice also that with the final year depletion estimates the 10th percentile CI is well above 20%B_0, implying that even though this analysis is uncertain the current depletion level is above the default limit reference point with more than a 90% likelihood (again more replicates are required before we could validly claim this).

The fitted trajectories can also provide an indication of the uncertainty surrounding the analysis.

```
years <- fish[,"year"]
par(mfrow=c(1,1),mai=c(0.45,0.45,0.05,0.05),oma=c(0.0,0,0.0,0.0))
par(cex=0.85, mgp=c(1.35,0.35,0), font.axis=7,font=7,font.lab=7)
ymax <- getmaxy(c(dynam[,,"PredCE"],fish[,"cpue"]))
plot(fish[,"year"],fish[,"cpue"],type="n",ylim=c(0,ymax),xlab="Year",
     ylab="CPUE",panel.first = grid())
for (i in 1:reps) lines(years,dynam[i,,"PredCE"],lwd=1,col="grey")
lines(years,answer$Dynamics$outmat[,"PredCE"],lwd=2,col=2)
points(years,fish[,"cpue"],cex=1.1,pch=16,col=4)
percs <- apply(dynam[,,"PredCE"],2,quants)
arrows(x0=years,y0=percs["5%",],y1=percs["95%",],length=0.03,angle=90,code
=3,col=2)
```

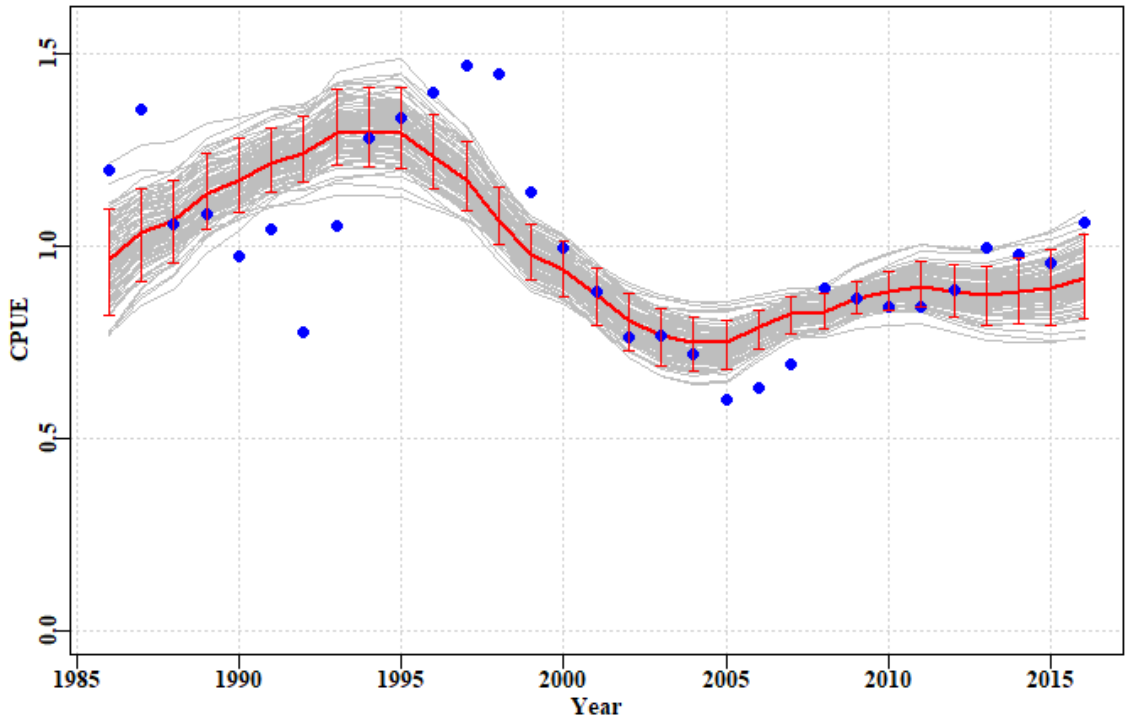

Figure 9. A plot of the original observed CPUE (blue dots), the optimum predicted CPUE (solid red line), the bootstrap predicted CPUE (the grey lines), and the 90th precentile confidence intervals around those predicted values.

There are clearly some major deviations between the predicted and the observed CPUE values but the median estimates and the confidence bounds around them remain well defined.

1.9 Discussion

In the Southern and Eastern Scalefish and Shark Fishery (SESSF), rather than using surplus production models or other simple dynamics approaches, empirical harvest strategies have been developed that use such time-series in empirical relationships that give rise directly to management related advice on catch levels (Little et al., 2011; Haddon, 2014). Such empirical harvest strategies can provide the needed management advice but do not determine stock status unless the reference period, used in such approaches, is assumed to be a proxy for the target reference point (and associated limit reference point) for sustainability. In the SESSF, this Tier 4 harvest strategy is used to determine whether a stock is over-fished or not, but currently cannot be used to determine whether over-fishing is occurring. In addition, there

is the strong assumption made that the commercial catch rate are a direct reflection of the stock biomass. There are, however, some species, for example mirror dory (*Zenopsis nebulosa*) where catch rates increase when catches increase and then decline once catches begin to decline (see the discussion above about whether the CPUE is informative). They appear to be fisheries based on availability rather than the fishery being the major influence on the stock biomass other aspects of the environment of the species appear to be driving its dynamics. The use of CPUE may this be misleading in such cases.

1.10 Management Advice with spm

If working with a species that requires on-going management then it is necessary to produce advice with respect to acceptable catches that will lead to a sustainable fishery or whatever other management goal is in place for the fishery. To generate such advice some form of harvest strategy (HS) is required to allow the outputs from the assessment to be converted into a recommended biological catch (RBC; informal harvest strategies are less repeatable than formal HS). This RBC may then be modified by fishery managers taking into account potential rates of change within a fishery or social or economic drivers of management decisions. It was possible to put forward suggestions for new harvest strategies using the catch-MSY method because none were available previously and that put forward was only a suggestion for a possible consideration. Putting forward a proposed harvest control rule for the spm approach without consultation with jurisdictional fisheries managers could produce suggestions incompatible with a particular jurisdictions objectives. There are harvest control rules that can be used once limit and target reference points are agreed upon and these can be utilized where considered appropriate. The SAFS process, however, does not currently require a target reference point even though most harvest control rules do require one.

1.11 References

Dick, E.J. and A.D. MacCall (2011) Depletion-based stock reduction analysis: a catch-based method for determining sustainable yields for data-poor fish stocks. *Fisheries Research* **110**(2): 331-341

Haddon, M. (2014) Tier 4 analyses in the SESSF, including deep water species. Data from 1986 – 2012. Pp 352 – 461 in Tuck, G.N. (ed) (2014) *Stock Assessment for the Southern and Eastern Scalefish and Shark Fishery 2013. Part 2.* Australian Fisheries Management Authority and CSIRO Marine and Atmospheric Research, Hobart. 313p.

Haddon, M., Klaer, N., Wayte, S., and G. Tuck (2015) *Options for Tier 5 approaches in the SESSF and identification of when data support for harvest strategies are inappro-priate.* CSIRO. FRDC Final Report 2013/200. Hobart. 115p.

Kimura, D.K. and J.V. Tagart (1982) Stock Reduction Analysis, another solution to the catch equations. *Canadian Journal of Fisheries and Aquatic Sciences* **39**: 1467 - 1472.

Kimura, D.K., Balsiger, J.W., and Ito, D.H. 1984. Generalized stock reduction analysis. *Canadian Journal of Fisheries and Aquatic Sciences* **41**: 1325–1333.

Little, L.R., Wayte, S.E., Tuck, G.N., Smith, A.D.M., Klaer, N., Haddon, M., Punt, A.E., Thomson, R., Day, J. and M. Fuller (2011) Development and evaluation of a cpue-based harvest control rule for the southern and eastern scalefish and shark fishery of Australia. *ICES Journal of Marine Science* **68**(8): 1699-1705.

Martell, S. and R. Froese (2013) A simple method for estimating MSY from catch and resilience. *Fish and Fisheries* **14**: 504-514

Polacheck, T., Hilborn, R. and A.E. Punt (1993) Fitting surplus production models: Comparing methods and measuring uncertainty. *Canadian Journal of Fisheries and Aquatic Sciences* **50**: 2597–2607.

Punt, A.E., Butterworth, D.S. and A.J. Penney (1995) Stock assessment and risk analysis for the South Atlantic population of albacore *Thunnus alalunga* using an age-structured production model *South African Journal of Marine Science* **16**: 287-310. http://dx.doi.org/10.2989/025776195784156476
R Core Team (2017). *R: A language and environment for statistical computing.* R Foundation for Statistical Computing, Vienna, Austria. URL https://www.R-project.org/. see also https://cran.r-project.org/
RStudio (2016) www.rstudio.com
Schaefer, M.B. (1954) Some aspects of the dynamics of populations important to the management of the commercial marine fisheries. *Bulletin, Inter-American Tropical Tuna Commission*, **1**: 25-56.
Schaefer, M.B. (1957) A study of the dynamics of the fishery for yellowfin tuna in the Eastern Tropical Pacific Ocean. *Bulletin, Inter-American Tropical Tuna Commission*, **2**: 247-285
Walters, C.J., Martell, S.J.D. and J. Korman (2006) A stochastic approach to stock reduction analysis. *Canadian Journal of Fisheries and Aquatic Sciences* **63**: 212 - 223.

1.12 Appendix: Surplus Production Model Equations

The surplus production model is used to describe the dynamics of the stock in terms of its exploitable biomass. In general terms where the dynamics are designated a function of the biomass at the start of a given year *t*:

$$B_{t+1} = B_t + f(B_t) - C_t$$

where $B_{t\_}$ represents the stock biomass at the start of year t, $f(B_t)$ represents a production function in terms of stock biomass (accounting for recruitment or new individuals, growth of biomass of current individuals, and natural mortality), and C_t being the catch in year t.
The model dynamics are also used to generate predicted values for the index of relative abundance in each year:

$$\hat{I}_t = \frac{C_t}{E_t} = qB_t$$

where $\hat{I}_t$ is the predicted or estimated value of the index of relative abundance, which is compared to the observed indices to fit the model to the data, E_t is the fishing effort in year t, and q is the catchability coefficient (defined as the amount of biomass/catch taken with one unit of effort).

A number of functional forms have been put forward to describe production. In **simpleSA** we have implemented two, the Schaefer model and a modified form of the Fox model:
The Schaefer (1954) model uses:

$$f(B_t) = rB_t\left(1 - \frac{B_t}{K}\right)$$

while the Fox (1970) uses:

$$f(B_t) = Ln(K)rB_t\left(1 - \frac{Ln(B_t)}{Ln(K)}\right)$$

Alternatively, a formulation from Polacheck *et al*. (1993) provides a general equation for the population dynamics that can be used for both the Schaefer and the Fox model depending on the value of a single parameter *p*.

$$B_{t+1} = B_t + \frac{r}{p}B_t\left(1 - \frac{B_t}{K}\right)^p - C_t$$

Thus, when p is set to 1.0 this equation becomes the same as the Schaefer model. But when p is set to a very small number, say 1e-0.8, then the formulation becomes equivalent to the Fox model's dynamics.

The Schaefer model assumes a symmetrical production curve with maximum surplus production (MSY) at 0.5_K_. The Fox model generates asymmetrical production curves with the maximum production at some lower level of depletion. The Schaefer model can be regarded as more conservative than the Fox in that it requires the stock size to be higher for maximum production and generally leads to somewhat lower levels of catch.

Other production functions could be added later. r represents a population growth rate that includes the balance between recruitment, growth in current biomass, and natural mortality, K is the maximum population size (the carrying capacity),the part in brackets, $\left(1-\frac{B_t}{K}\right)$, represents a density dependent term that trends linearly to zero as B_t tends to wards K in the Schaefer model. In the Schaefer mdoel the production curve is symmetric with maximum production occurring at $0.5K$. In the Fox model the density dependent terms introduces some non-linearity which generates an asymmetrical production curve with the point of maximum production moved to $< 0.5K$ (see later diagrams of the prodution curves; See Haddon (2011) for further details).

Thus for the Schaefer model we would have:

$$B_{t+1} = B_t + rB_t\left(1-\frac{B_t}{K}\right) - C_t$$

where $B_0 = K$ or B_{init} depending on whether the stock was deemed to be depleted when data from the fishery first became available.

It appears that fitting the model to data would require at least three parameters, the r, the K, and the q (B_{init} might also be needed). However, it is possible to used what is known as a "closed-form" method for estimating the catchability coefficient q:

$$\hat{q} = \exp\left(\frac{1}{n}\sum Ln\left(\frac{I_t}{B_t}\right)\right)$$

which is the back-transformed geometric mean of the observed CPUE divided by the exploitable biomass.

Sum of Squared Residuals

Such a model can be fitted using least squares or, more properly, the sum of squared residual errors:

$$ssq = \sum\left(Ln(I_t) - Ln\left(\hat{I}_t\right)\right)^2$$

the log-transformations are required as CPUE is considered typically to be distributed log-normally and the least-squares method implies normal random errors. The least squares approach tends to be more robust when first searching for a set of parameters that enable a model to fit to available data. However, once close to a solution more options become available if one then uses maximum likelihood methods.

Maximum likelihood methods, as the name dictates entail maximizing the likelihood of the available data given the model and a proposed set of parameters. Very often the likelihoods involved when fitting models are very small numbers. To avoid rounding errors (even when

using 64 bit computers) it is standard to use log-likelihoods rather than likelihoods (in that way the log-likelihoods can be individually added together rather than multiply the individual likelihoods). Additionally, rather than maximizing a log-likelihood, minimization often best matches our intuitions about model fitting and this involves minimizing the negative log-likelihood. The full log-normal negative log likelihood is:

$$L(data|B_{init}, r, K, q) = \prod_t \frac{1}{I_t\sqrt{2\pi\,\hat{\sigma}}} e^{\frac{-\left(LnI_t - Ln\hat{I}_t\right)}{2\hat{\sigma}^2}}$$

Fortunately, the negative log-likelihood can be simplified (Haddon, 2011), to become:

$$-veLL = \frac{n}{2}\left(Ln(2\pi) + 2Ln\left(\hat{\sigma}\right) + 1\right)$$

where the maximum likelihood estimate of the standard deviation, $\hat{\sigma}$ is given by:

$$\hat{\sigma} = \sqrt{\frac{\sum\left(Ln(I_t) - Ln\left(\hat{I}_t\right)\right)^2}{n}}$$

Note the division by *n* rather than by *n-1*. Strictly the -veLL is followed by an additional term:

$$-\sum Ln(I_t)$$

the sum of the log-transformed observed catch rates. But as this will be constant it is usually omitted.

Estimating Management Statistics

The Maximum Sustainable Yield can be calculated for the Schaefer model simply by using:

$$MSY = \frac{rK}{4}$$

However, for the more general equation using the *p* parameter from Polacheck *et al* (1993) one needs to use:

$$MSY = \frac{rK}{(p+1)^{\frac{(p+1)}{p}}}$$