

# R documentation

of all in ‘PBSawatea’

July 20, 2011

## R topics documented:

allEqual . . . . .	2
calc.projExpect . . . . .	3
calc.refVal . . . . .	4
close.allWin . . . . .	4
compB0 . . . . .	5
cquantile . . . . .	6
findTarget . . . . .	7
get.resFile . . . . .	8
getYrIdx . . . . .	9
graphics . . . . .	10
importCol2 . . . . .	10
importMCMC.ddiff . . . . .	12
importProjRec . . . . .	13
importRes . . . . .	14
load.allResFiles . . . . .	15
MAfun2 . . . . .	16
mainMenu . . . . .	17
makeErrMat . . . . .	18
msyCalc . . . . .	19
out.pmTables . . . . .	20
panLab . . . . .	20
panLegend . . . . .	21
PBSawatea . . . . .	22
plotB2 . . . . .	23
plotBmcmcPOP . . . . .	25
plotBox . . . . .	26
plotChains . . . . .	29
plotCPUE . . . . .	30
plotDensPOP . . . . .	31
plotIndex2 . . . . .	34
plotIndexNotLattice . . . . .	35

plotRmcmcPOP . . . . .	36
plotSnail . . . . .	37
plotTracePOP . . . . .	38
plotVBcatch . . . . .	40
plotVBnorm . . . . .	41
plt.ageResidsPOP . . . . .	42
plt.allTraces . . . . .	43
plt.expRate . . . . .	44
plt.idx . . . . .	45
plt.mpdGraphs . . . . .	45
plt.numR . . . . .	47
plt.quantBio . . . . .	48
plt.ssbVbCatch . . . . .	49
plt.stdResids . . . . .	50
readAD . . . . .	50
refPoints . . . . .	51
reweight . . . . .	52
runADMB . . . . .	54
runMCMC . . . . .	55
runMPD . . . . .	56
runSweave . . . . .	57
runSweaveMCMC . . . . .	58
srFun . . . . .	59
stdRes.CA . . . . .	60
stdRes.index . . . . .	61
tabSAR . . . . .	62

<b>Index</b>	<b>63</b>
--------------	-----------

---

allEqual	<i>Are All Values Equal to the First?</i>
----------	---

---

**Description**

A short-cut function for `all (x==x[1])`, which asks are all values in *x* equal to the first value in *x*.

**Usage**

`allEqual(x)`

**Arguments**

*x*                      vector of values.

**Value**

TRUE or FALSE

**See Also**

[all](#), [clearAll](#), [clipVector](#)

---

calc.projExpect	<i>Calculate Expectations and Probabilities</i>
-----------------	---

---

**Description**

Calculate the expectation of projection to reference, and probability of being greater than reference.

**Usage**

```
calc.projExpect ( obj, projObj, refYrs )
calc.projExpect2( obj, projObj, refList )
calc.projProbs   ( obj, projObj, refYrs )
calc.projProbs2  ( obj, projObj, refList )
calc.refProbs    ( projObj=currentProj$B, refPlist=refPointsList )
```

**Arguments**

obj	matrix of biomass MCMCs.
projObj	matrix of biomass projections.
refYrs	numeric vector of reference years
refList	list of reference years (numeric vectors).
refPList	list of reference points.

**Details**

```
calc.projExpect...Calculate the expectation of projection to reference.
.....Compare reference years to projection years.
calc.projExpect2...Calculate expectation (projection biomass / reference biomass).
calc.projProbs...Calculate the probability of being greater than refYrs.
.....Compare reference years to projection years.
calc.projProbs2...Calculate the probability of being greater than refYrs.
.....Compare reference years to projection years.
calc.refProbs...Calculate the reference probabilities (based on calc.projProbs2).
```

**Value**

Decision tables

**Author(s)**

Andrew Edwards, Pacific Biological Station, Fisheries and Oceans Canada, Nanaimo BC

**See Also**

[calc.refVal](#)

---

calc.refVal	<i>Calculate Reference Value for Performance Measure</i>
-------------	--

---

**Description**

Calculate the reference value for performance measures.

**Usage**

```
calc.refVal(obj, refYrs, fun=mean)
```

**Arguments**

obj	scape Biomass matrix with $n$ rows and $m$ columns, where $n$ = number of MCMC samples, and $m$ = number of years.
refYrs	numeric years in reference period.
fun	the function to apply to reference period $i$ .

**Value**

Returns a vector of length `nrow(obj)` reference values.

**Author(s)**

Andrew Edwards, Pacific Biological Station, Fisheries and Oceans Canada, Nanaimo BC

**See Also**

[calc.projExpect](#), [findTarget](#)

---

close.allWin	<i>Close All Open Devices</i>
--------------	-------------------------------

---

**Description**

Close all open devices.

**Usage**

```
close.allWin()
```

**See Also**

[closeWin](#)

## Description

Compare COSWEIC reference criteria and DFO reference points relative to  $B_0$ .  
The figure concept comes from Chris Woods (PBS).

## Usage

```
compB0(B, Mnam=c("Est M", "Fix M"), ratios=c(0.4, 0.8),
       incl.Bt=TRUE, boxwidth=0.6, figgy=FALSE, width=12, height=9, ...)
```

## Arguments

B	list of list of MCMC samples (see <b>Details</b> ); the first level of the list is the model run, while the second level contains MCMC samples (one of which should be $B_0$ which acts as the divisor to the other MCMCs).
Mnam	model names for the boxplot.
ratios	reference levels of $B_{MSY}$ (usually 0.4 and 0.8).
incl.Bt	logical: if TRUE, include the current spawning stock biomass in terms of $B_0$ .
boxwidth	width of the box in x-units.
figgy	logical: if TRUE, send figure to four output files (.eps, .pdf, .png, and .wmf).
width	width of the output files in inches.
height	height of the output files in inches.
...	additional values for plotBox's argument pars.

## Details

An example of the input list B:

```
List of 2
..$ 29.01:List of 3
....$ B0.MCMC : num [1:1000]
....$ Bt.MCMC : num [1:1000]
....$ Bmsy.MCMC: num [1:1000]
..$ 30.01:List of 3
....$ B0.MCMC : num [1:1000]
....$ Bt.MCMC : num [1:1000]
....$ Bmsy.MCMC: num [1:1000]
```

The function creates a figure comparing COSEWIC criteria and reference points for the Schaefer surplus production model and specified multiples of  $B_{MSY}$  from catch-at-age models. The coordinate space (y-axis) is relative to  $B_0$ .

**Value**

Invisibly returns the list object `xBox` used to create the boxplot.

**Note**

Uses a modified version of `boxplot` called `plotBox`.

**Author(s)**

Rowan Haigh, Pacific Biological Station, Fisheries and Oceans Canada, Nanaimo BC

**See Also**

`plotBox`, `importMCMC`, `msyCalc`  
`plotmath`, `boxplot`, `bxp`

---

cquantile

*Running Quantile*


---

**Description**

Creates a set of running quantiles from MCMC traces.  
 (Uses subfunction found in **coda**'s function `cumuplot`.)

**Usage**

```
cquantile(z, probs)
cquantile.vec(z, prob)
```

**Arguments**

<code>z</code>	an MCMC object.
<code>probs</code>	vector of quantiles.
<code>prob</code>	single quantile.

**Value**

`cquantile`.....running quantile matrix  
`cquantile.vec`...running quantile vector

**Note**

Arni Magnusson describes a running quantile as:  
*“the evolution of the sample quantiles as a function of the number of iterations”*

**Author(s)**

Andrew Edwards, Pacific Biological Station, Fisheries and Oceans Canada, Nanaimo BC

**See Also**

[cumuplot](#), [plotTracePOP](#)

---

findTarget

*Find Time to Achieve a Target Reference Point*


---

**Description**

Find the time (years) to achieve a recovery target (including a moving target) with a given confidence. Produce decision tables showing the probability of exceeding the reference point.

**Usage**

```
findTarget(Vmat, yrU=as.numeric(dimnames(Vmat)[[2]]), yrG=90,
           ratio=0.5, target=B0.MCMC, conf=0.95, plotit=FALSE, retVal="N")
```

**Arguments**

Vmat	matrix of projected biomass values $B_{Nt}$ , where $N$ = number of MCMCs and $t$ = projection year.
yrU	user-specified projection years.
yrG	number of years $G$ for a moving target window ( <i>e.g.</i> , 3 YMR generations = 90y); might not work for all possibilities.
ratio	recovery target ratio $R$ .
target	recovery target values $T_N$ = B0.MCMC for ratios of $B_0$ ; = Bmsy.MCMC for ratios of $B_{MSY}$ ; = Bt.MCMC for moving window of $B_{N,t-G}$ .
conf	confidence level $C$ required.
plotit	logical: if TRUE, plot the probability $p_t$ of exceeding target reference point.
retVal	character name of object to return: retVal="N" : creates global object "Ttab" (see below); retVal="p.hi" : creates global object "Ptab" (see below).

**Details**

As this function uses Bayesian output, there are  $N$  (*e.g.*, 1000) values of some target  $T_N$ , which can remain fixed ( $B_0$ ,  $B_{MSY}$ ) or move forward in time  $G$  years before the projection year  $t$  (that is  $T_{N,t-G}$ ). For simplification, we'll just call all targets  $T_N$ .

The probability of exceeding a target ratio  $R$  is:

$$p_t = \frac{1}{N} \sum^N \left[ \frac{B_{Nt}}{T_N} \geq R \right],$$

where  $R$  = target ratio of the reference point (e.g.,  $0.4B_{MSY}$  ( $R=0.4$ ),  $0.2B_0$  ( $R=0.2$ ),  $0.5B_{t-G}$  ( $R=0.5$ )).

At a glance, we can see for any given projection year  $t$  whether the probability of achieving a target ratio is greater than the confidence required:

$$p_t \geq C,$$

where  $C$  is the confidence level acceptable.

### Value

If `retVal="N"` then the function returns a data frame object called "`Ttab`" in the user's global environment. This table reports the number of years to achieve the target reference point at various catch levels with a specified confidence.

If `retVal="p.hi"` then the function returns a list object called "`Ptab`" in the user's global environment. This list contains data frames (tables) that report the probability of achieving various reference points at specified catch levels.

Any other `retVal` will return a list of the specified object, if it exists in the function.

### Author(s)

Rowan Haigh, Pacific Biological Station, Fisheries and Oceans Canada, Nanaimo BC

### See Also

[runSweaveMCMC](#)

---

`get.resFile`

*Get Awatea Results Files for Menu*

---

### Description

A function that retrieves the names of Awatea results files (`.res$`) for use in the `mainMenu` command. When choice is made, the function loads the results file and assigns it to the global environment as `currentRes`.

### Usage

```
get.resFile(resFile=NULL)
```

### Arguments

`resFile` supposedly the name of a results file, but the code suggests that argument is ignored.



**Value**

A results file chosen from a menu.

**Note**

AME: made changes so that options are compatible with those in `load.allResFiles`. Previously, trouble occurred when overwriting.

**See Also**

[mainMenu](#), [importCol2](#)

---

`getYrIdx`*Select a Subset of Years for Plotting*

---

**Description**

Select a subset of years for which many years are available. The default is to select 5-year increments.

**Usage**

```
getYrIdx(yrNames, mod=5)
```

**Arguments**

<code>yrNames</code>	vector (character or numeric) of years.
<code>mod</code>	select the years modulo <code>mod</code> .

**Value**

Subset of input years that are modulo `mod`.

**Author(s)**

Andrew Edwards, Pacific Biological Station, Fisheries and Oceans Canada, Nanaimo BC

**See Also**

[findPat](#), [pad0](#)

---

graphics

---

*Open a Windows Device*


---

**Description**

Open a windows device in portrait or landscape mode.

**Usage**

```
graphics(view = "portrait")
```

**Arguments**

view	if "portrait", set width = 8.5 in and height = 11 in if "landscape", set width = 11 in and height = 8.5 in
------	---

**See Also**

[resetGraph](#)

---

importCol2

---

*Import Coleraine Model Results (AME version)*


---

**Description**

Import Coleraine model results from .res file, and rearrange into a standard format suitable for plotting.

**Usage**

```
importCol2(res.file, info="", Dev=FALSE, CPUE=FALSE, Survey=FALSE,
  CAC=FALSE, CAs=FALSE, CLc=FALSE, CLs=FALSE, LA=FALSE,
  quiet=TRUE, extra=TRUE)
```

**Arguments**

res.file	name of Coleraine model results file to import.
info	optional string containing information to store with model results.
Dev	logical: whether recruitment deviates were estimated in model.
CPUE	logical: whether model was fitted to catch-per-unit-effort data.
Survey	logical: whether model was fitted to survey abundance index data.
CAC	logical: whether model was fitted to commercial catch-at-age data.
CAs	logical: whether model was fitted to survey catch-at-age data.
CLc	logical: whether model was fitted to commercial catch-at-length data.

CLs	logical: whether model was fitted to survey catch-at-length data.
LA	logical: whether model was fitted to length-at-age data.
quiet	logical: whether to report progress while parsing file.
extra	logical: if TRUE, import likelihoods, parameters, priors, and recruitment residuals.

## Details

This function was modified from the original `importCol` function in the **scape** package to grab extra data.

## Value

A list of class `scape` containing at least `N`, `B`, and `Sel`. The other elements may or may not be included in the list, depending on how `importRes` was called:

<code>N</code>	predicted numbers at age
<code>B</code>	predicted biomass, recruitment, and observed landings (year things)
<code>Sel</code>	predicted selectivity and observed maturity (age things)
<code>Dev</code>	predicted recruitment deviates from the stock-recruitment curve
<code>CPUE</code> , <code>Survey</code>	commercial and survey abundance index and fit
<code>CAC</code> , <code>CAs</code>	commercial and survey C@A (catch at age) and fit
<code>CLC</code> , <code>CLs</code>	commercial and survey C@L (catch at length) and fit
<code>LA</code>	observed L@A and fit

## Note

This import function is implemented for the Coleraine statistical catch-at-age software, and can serve as a template for **scape** users who would like to implement import functions for specific stock assessment software.

The functions `ll` (package **gdata**) and `head` are recommended for browsing model results, e.g. `ll(x.cod)`; `ll(x.cod$N)`; `head(x.cod$N)`.

## References

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

## See Also

`importRes`, `read.table`, `readLines`, and `scan` to import any data.  
[scape-package](#) gives an overview of the package **scape**.

---

importMCMC.ddiff	<i>Import Functions for PJS Delay Difference Model</i>
------------------	--

---

## Description

Make a **scapeMCMC** object identical in format to the result of `importMCMC` (or `importProj`) from PJS delay difference model output.

The difference is that *B* is biomass defined by a delay difference model.

## Usage

```
importMCMC.ddiff()
importProj.ddiff()
```

## Value

`importMCMC.ddiff` returns a list object containing:

- L . . . likelihood MCMCs,
- P . . . parameter MCMCs,
- B . . . spawning biomass MCMCs,
- R . . . recruitment MCMCs.

`importProj.ddiff` returns a list object containing:

- B . . . projected biomass,
- Y . . . projected yields.

## Note

Get the biomass projection – PJS does 1 year ahead projection. The column "X" appears as the last column because trailing ", " exist in the `mcmcprojbiom.csv` file.

Note also that "cat=" in .csv file becomes "cat ." in `read.table`.

## Author(s)

Andrew Edwards, Pacific Biological Station, Fisheries and Oceans Canada, Nanaimo BC

## See Also

[importMCMC](#), [importProj](#), [read.table](#)

---

importProjRec      *Import Projected Recruitment*


---

**Description**

Import the projected recruitments (actually, the values are random normals  $N(0, 1)$ ).

**Usage**

```
importProjRec(dir, info="", coda=FALSE, quiet=TRUE)
```

**Arguments**

dir	directory where MCMC projections reside.
info	user-supplied information, if desired.
coda	logical: if TRUE, use the function <code>mcmc</code> in the package <b>coda</b> to generate MCMCs.
quiet	logical: if TRUE, print progress messages to the R console.

**Details**

The values saved by the Awatea code are random normals  $N(0, 1)$ , which for a particular MCMC sample are the same for all the catch strategies.

**Value**

A list object comprising:

B	data frame of spawning biomass (dim = MCMC samples by projected years)
Y	data frame of yield (dim = MCMC samples by projected years)
eps	data frame of $\epsilon_t$ (dim = MCMC samples by projected years)

**Note**

The function `importProj` does not import recruitment residuals.

This function grabs the `tempdev` values from Awatea, which are just  $N(0, 1)$  values, then multiplies them by  $\sigma_R$  to yield  $\epsilon_t \sim N(0, \sigma_R^2)$ .

The parameter value for  $\sigma_R$  can be found in `currentRes$extra$residuals$p_log_RecDev[6]`.

**Author(s)**

Andrew Edwards, Pacific Biological Station, Fisheries and Oceans Canada, Nanaimo BC

**See Also**

[importProj](#)  
**coda:** [mcmc](#)

importRes

*Import Coleraine Model Results (RH version)***Description**

Import Coleraine model results from `.res` file, and rearrange into a standard format suitable for plotting.

**Usage**

```
importRes(res.file, info="", Dev=FALSE, CPUE=FALSE, Survey=FALSE,
          CAC=FALSE, CAs=FALSE, CLc=FALSE, CLs=FALSE, LA=FALSE, quiet=TRUE,
          extra=TRUE, sep=" ")
```

**Arguments**

<code>res.file</code>	name of Coleraine model results file to import.
<code>info</code>	optional string containing information to store with model results.
<code>Dev</code>	logical: whether recruitment deviates were estimated in model.
<code>CPUE</code>	logical: whether model was fitted to catch-per-unit-effort data.
<code>Survey</code>	logical: whether model was fitted to survey abundance index data.
<code>CAC</code>	logical: whether model was fitted to commercial catch-at-age data.
<code>CAs</code>	logical: whether model was fitted to survey catch-at-age data.
<code>CLc</code>	logical: whether model was fitted to commercial catch-at-length data.
<code>CLs</code>	logical: whether model was fitted to survey catch-at-length data.
<code>LA</code>	logical: whether model was fitted to length-at-age data.
<code>quiet</code>	logical: whether to report progress while parsing file.
<code>extra</code>	logical: if <code>TRUE</code> , import likelihoods, parameters, priors, and recruitment residuals.
<code>sep</code>	the field separator character (usually <code>" "</code> or <code>"\t"</code> ).

**Details**

This function was modified from the original `importCol` function in the **scape** package to grab extra data and to deal with anomalous characters in Coleraine results files.

**Value**

A list of class `list` containing at least `N`, `B`, and `Sel`. The other elements may or may not be included in the list, depending on how `importRes` was called:

<code>N</code>	predicted numbers at age
<code>B</code>	predicted biomass, recruitment, and observed landings (year things)

Sel	predicted selectivity and observed maturity (age things)
Dev	predicted recruitment deviates from the stock-recruitment curve
CPUE, Survey	commercial and survey abundance index 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	observed L@A and fit

### Note

This import function is implemented for the Coleraine statistical catch-at-age software, and can serve as a template for **scape** users who would like to implement import functions for specific stock assessment software.

The functions `ll` (package **gdata**) and `head` are recommended for browsing model results, e.g. `ll(x.cod)`; `ll(x.cod$N)`; `head(x.cod$N)`.

### References

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

### See Also

`runADMB`, `readAD`, `reweight`, `importCol2`  
`read.table`, `readLines`, and `scan` to import any data.  
[scape-package](#) gives an overview of the package **scape**.

---

load.allResFiles      *Load All Awatea .res Files*

---

### Description

Load all Awatea `.res` files in the working directory into a list object.

### Usage

```
load.allResFiles(resList = NULL)
```

### Arguments

`resList`      AME: sets directory to path above current and sets the pattern to "results.dat\$"; probably deprecated.

### Value

List of multiple calls to `importCol2`.

**Note**

If deprecated, remove from package **PBSawatea**.

**Author(s)**

Andrew Edwards, Pacific Biological Station, Fisheries and Oceans Canada, Nanaimo BC

**See Also**

`importCol2`

---

MAfun2

---

*Calculate Mean Age by Year*


---

**Description**

Calculate mean ages from proportions-at-age (modified from a subfunction in `runADMB`).

**Usage**

```
MAfun2(padata, brks=NULL)
```

**Arguments**

<code>padata</code>	proportion-at-age data CAc or CAs from a call to <code>importCol2</code> .
<code>brks</code>	breaks specified as numeric years to split the commercial data up into regimes that may account for index discontinuities ( <b>not used</b> ).

**Details**

Mean age function supplied by Chris Francis (2011).

`padata` has fields:

`Series...` series identifier

`Year.....` numeric year

`Age.....` age bin

`Obs.....` observed proportions

`Fit.....` predicted (fitted) proportions

`SS.....` sample size (effective  $N$ )

**Value**

List object of observed and expected mean ages, variance of expected ages, and a few bits and bobs.

**Author(s)**

Rowan Haigh, Pacific Biological Station, Fisheries and Oceans Canada, Nanaimo BC.



## References

Francis, R.I.C.C. (2011, in press) Data weighting in statistical fisheries stock assessment models. *Canadian Journal of Fisheries and Aquatic Sciences*.

## See Also

[runADMB](#), [importCol2](#)

---

mainMenu

*Create a Menu of Options and Actions*

---

## Description

From a main menu, the user can choose various options and actions.

We tend to do everything from the command line so the menu functionality fosters the warning: *CAVEAT EMPTOR*.

## Usage

```
mainMenu()  
loadMenu()  
mpdMenu()  
mcmcMenu()  
utilMenu()
```

## Details

### Main menu items:

```
Import files  
MPD plots  
Plot all MPD graphs  
Save all MPD plots to PNG  
MCMC plots  
Plot all MCMC plots  
Save all MCMC plots to PNG  
Close all graphics windows  
Help & Utilities
```

### Load menu items:

```
Get Awatea res file  
Get Awatea MCMC file  
Get Awatea projection file  
Load all res files in working directory  
Get PJS Delay Difference MCMC+Projection
```

### MPD menu items:

```
Plot biomass, recruitment, catch  
Plot numbers at age
```

Plot selectivity and maturity  
 Plot commercial catch-at-age results  
 Plot survey catch-at-age results  
 Plot survey catch-at-length results  
 Plot abundance index  
 All residual plots  
 Plot multi-panel biomass, recruitment, catch  
 Plot multi-panel exploitation rate  
 Plot alternative numbers at age

#### **MCMC menu items:**

Plot biomass and projections by policy  
 Probability of projection biomass > reference  
 Expectation of projection biomass / reference  
 Plot biomass posterior densities (plotDens)  
 Plot recruitment posterior densities (plotDens)  
 Plot parameter posterior densities (plotDens)  
 Plot cumulative quantiles (plotCumu)  
 Plot traces (plotTrace)  
 Plot PJS traces (plt.allTraces)

#### **Utils menu items:**

scape Help  
 scapeMCMC Help  
 Portrait graphsheet  
 Landscape graphsheet

#### **See Also**

[get.resFile](#), [importCol2](#)

---

makeErrMat

*Make Ageing Error Matrix for Awatea*

---

#### **Description**

Make a simple ageing error matrix for Awatea.

#### **Usage**

```
makeErrMat(N=60, ondiag=0.8, offdiag=0.1, corner=0.9)
```

#### **Arguments**

N	numeric scalar indicating number of age classes, which determines the dimension of the matrix.
ondiag	numeric value to appear along the matrix diagonal.
offdiag	numeric value to appear one cell to the left and right of the matrix diagonal.
corner	numeric value to appear in the top left and bottom right corners of the matrix.

**Value**

Simple symmetric ageing error matrix.

**Author(s)**

Rowan Haigh, Pacific Biological Station, Fisheries and Oceans Canada, Nanaimo BC

**See Also**

[plotProp](#), [weightBio](#)

---

msyCalc

*Calculate the Maximum Sustainable Yield*


---

**Description**

Load in `MSY.out` and calculate the MSY (maximum sustainable yield).

**Usage**

```
msyCalc(dir = getwd(), error.rep = 1)
```

**Arguments**

`dir`                      working directory.  
`error.rep`                numeric: if 1, report errors (reaching bounds), if 0 do not.

**Value**

Returns a list object containing:  
`yield...` maximum sustainable yield,  
`u.....` exploitation rate at MSY,  
`VB.....` vulnerable biomass at MSY,  
`B.....` spawning biomass at MSY,  
`nProj...` number of projections needed to reach MSY.

**Note**

See `msyTestCreating.r` for full details when figuring this out.

**Author(s)**

Andrew Edwards, Pacific Biological Station, Fisheries and Oceans Canada, Nanaimo BC

**See Also**

[findTarget](#)

---

<code>out.pmTables</code>	<i>Write Decision Tables to Comma-Delimited Files</i>
---------------------------	---

---

**Description**

Write decision tables to comma-delimited text files (`.csv`).

**Usage**

```
out.pmTables(obj, fileName="pm", dec=3)
```

**Arguments**

<code>obj</code>	list object containing tables (matrices or data frames).
<code>fileName</code>	prefix for output file names.
<code>dec</code>	number of decimal places to retain.

**Value**

Comma-delimited text files (`.csv`).

**Author(s)**

Andrew Edwards, Pacific Biological Station, Fisheries and Oceans Canada, Nanaimo BC

**See Also**

[write.table](#), [writeList](#)

---

<code>panLab</code>	<i>Write Text to Figure in Relative (0:1) Coordinates</i>
---------------------	---

---

**Description**

Write text to a figure by first setting the coordinate space to lie between 0 and 1:  

```
par(usr=c(0,1,0,1)).
```

**Usage**

```
panLab(x, y, txt, ...)
```

**Arguments**

<code>x</code>	relative x-coordinate.
<code>y</code>	relative y-coordinate.
<code>txt</code>	text to add to figure.
<code>...</code>	additional arguments sent to function <code>text</code> .

**Note**

Currently, this function does not reset the coordinate space to the original.  
Use [addLabel](#) instead.

**See Also**

[addLabel](#), [addLegend](#)

---

panLegend

*Place a Legend in a Figure using Relative (0:1) Coordinates*

---

**Description**

Place a legend in a figure by first setting the coordinate space to lie between 0 and 1:  
`par(usr=c(0,1,0,1))`.

**Usage**

```
panLegend(x, y, legTxt, ...)
```

**Arguments**

x	relative x-coordinate.
y	relative y-coordinate.
legTxt	legend text to add to figure.
...	additional arguments sent to function legend.

**Note**

Currently, this function does not reset the coordinate space to the original.  
Use [addLegend](#) instead.

**See Also**

[addLabel](#), [addLegend](#)

## Description

**PBSawatea** contains the code used for the modelling of populations of Pacific Ocean Perach (*Sebastes alutus*) and Yellowmouth Rockfish (*S. reedi*) along the British Columbia (BC) coast.

Implementation is done using a modified version of the Coleraine statistical catch-at-age software (Hilborn *et al.* 2003) called Awatea (Alan Hicks, NOAA, pers. comm.). Awatea is a platform for implementing the AD (Automatic Differentiation) Model Builder software (Otter Research 1999), which provides (a) maximum posterior density estimates using a function minimiser and automatic differentiation, and (b) an approximation of the posterior distribution of the parameters using the Markov Chain Monte Carlo (MCMC) method, specifically using the Hastings-Metropolis algorithm (Gelman *et al.* 2004).

Running of Awatea is streamlined using code written in R (R Development Core Team 2009), rather than the original Microsoft Excel implementation. Figures and tables of output are automatically produced through R using code adapted from the R packages **scape** (Magnusson 2009) and **scapeMCMC** (Magnusson and Stewart 2007). We use the R function `Sweave` (Leisch 2008) in the package **utils** to automatically collate, via LATEX, the large amount of figures and tables into a single portable document file (.pdf) for each model run.

We provide master `Sweave` files used in folder `../library/PBSawatea/snw` to build the .pdf document. The user must copy these to a local working directory if they are not already there.

## References

- Gelman, A., Carlin, J.B., Stern, H.S. and Rubin, D.B. (2004) Bayesian data analysis, 2nd edition. Chapman and Hall/CRC, New York, 668 p.
- Hilborn, R., Maunder, M., Parma, A., Ernst, B. Payne, J., and Starr, P. (2003) Coleraine: a generalized age-structured stock assessment model. *School of Aquatic and Fishery Sciences*, University of Washington, 54 p.
- Leisch, F. (2008) Sweave, R package.
- Magnusson, A. (2009) Scape – statistical catch-at-age plotting environment, R package.
- Magnusson, A. and Stewart, I. (2007) MCMCscape – MCMC diagnostic plots. R package.
- Otter Research Ltd. (1999) An introduction to AD Model Builder for use nonlinear modeling and statistics. Otter Research Ltd., British Columbia. 194 p.
- R Development Core Team (2011) R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria.  
ISBN 3-900051-07-0  
<http://www.R-project.org>

plotB2

*Plot Biomass, Recruitment, and Landings (AME Version)***Description**

Plot scape model predicted biomass, stock recruitment, and landings.

AME: This is an alteration of Arni Magnussons `plotB` function to accommodate PJS's request not to show biomass prior to fishery and survey indices period.

**Usage**

```
plotB2(model, what="d", series=NULL, years=NULL, axes=TRUE, div=1,
       legend="bottom", main="", xlab="", ylab="", cex.main=1.2,
       cex.legend=1, cex.lab=1, cex.axis=0.8, las=1,
       tck=c(1,what=="d")/2, tick.number=5, lty.grid=3, col.grid="white",
       pch=16, cex.points=0.8, col.points="black", lty.lines=1:3,
       lwd.lines=2, col.lines="black", ratio.bars=3, col.bars="grey",
       plot=TRUE, ...)
```

**Arguments**

<code>model</code>	fitted scape model.
<code>what</code>	what to plot: "d"[efault], "s"[tock recruitment], or "l"[andings].
<code>series</code>	vector of strings indicating which column names in <code>model\$B</code> data frame to plot (all by default).
<code>years</code>	vector of numbers indicating which years to include (all by default).
<code>axes</code>	whether to plot axis values.
<code>div</code>	denominator to shorten values on the y axis, or a vector with two elements referring to x and y axis.
<code>legend</code>	legend location: "bottom", "left", "top", "right", or "" to suppress legend.
<code>main</code>	main title.
<code>xlab</code>	x-axis label.
<code>ylab</code>	y-axis label.
<code>cex.main</code>	size of main title.
<code>cex.legend</code>	size of legend text.
<code>cex.lab</code>	size of axis labels.
<code>cex.axis</code>	size of tick labels.
<code>las</code>	orientation of tick labels: 0=parallel, 1=horizontal, 2=perpendicular, 3=vertical.
<code>tck</code>	tick mark length.
<code>tick.number</code>	number of tick marks.

<code>lty.grid</code>	line type of gridlines.
<code>col.grid</code>	color of gridlines.
<code>pch</code>	symbol for points.
<code>cex.points</code>	size of points.
<code>col.points</code>	color of points.
<code>lty.lines</code>	line type of main lines.
<code>lwd.lines</code>	line width of main lines.
<code>col.lines</code>	color of main lines.
<code>ratio.bars</code>	width of bars.
<code>col.bars</code>	color of bars.
<code>plot</code>	whether to draw plot.
<code>...</code>	passed to <code>xyplot</code> and <code>panel.superpose</code> .

### Details

The "d"[efault] plot shows spawning biomass and vulnerable biomass as lines, and landings as bars, on the same scale.

### Value

When `plot=TRUE`, a trellis plot is drawn and a data frame is returned, containing the data used for plotting. When `plot=FALSE`, a trellis object is returned.

### Note

The `Args` function from the **gdata** package is recommended for reviewing the arguments, instead of `args`.

### See Also

[xyplot](#), [panel.barchart](#), and [panel.superpose](#) are the underlying drawing functions.

[plotCA](#), [plotCL](#), [plotIndex](#), [plotIndex2](#) and [plotLA](#) plot model fit and data.

[plotB](#), [plotN](#), and [plotSel](#) plot derived quantities.

[scape-package](#) gives an overview of the **scapeMCMC** package.



plotBmcmcPOP

*Plot Spawning and Vulnerable Biomass***Description**

Plot spawning and vulnerable biomass from posterior as boxplots, and add catch bars on same graph.

**Usage**

```
plotBmcmcPOP( obj, currentRes1=currentRes,
              p=c(0.025,0.25,0.5,0.75,0.975),
              xyType="quantBox",
              lineType=c(3,2,1,2,3),
              refLines=NULL, xLim=NULL, yLim=NULL,
              userPrompt=FALSE, save=T, xLab=c(1939,1939,1939),
              yLab=c(10000,70000,170000),
              textLab=c("catch","spawning","vulnerable"),
              yaxis.by=10000, tcl.val=-0.2, ...)
```

**Arguments**

obj	MCMC data frame of $B$ (currentMCMC\$B).
currentRes1	list/scape object created by <code>importCol2</code> .
p	quantiles to use in <code>quantBox</code> .
xyType	type of plot (currently only uses <code>quantBox</code> ).
lineType	line types to use in <code>quantBox</code> .
refLines	reference lines to add to plot.
xLim	limits of the x-axis.
yLim	limits of the y-axis.
userPrompt	<b>not used</b>
save	<b>not used</b>
xLab	x-coordinates for labels.
yLab	y-coordinates for labels.
textLab	text labels to display on plot.
yaxis.by	increments along the y-axis to place tick marks.
tcl.val	tick length.
...	additional arguments passed to the function <code>rect</code> .

**Note**

Combines ideas from `plt.quantBio` and `plotB2`. Don't need lattice, just one figure, no panels. Vulnerable biomass has no posterior saved, which must be why it's not been done before. Hmmm.... still worth seeing spawning though?

Taking what is needed from `plt.quantBio`, this basically works:

```
plt.quantBio(currentMCMC$B, xyType=rpType),
though it creates 2x3 plots. The object should be the specific MCMC posterior by year (so just a
data frame), e.g., currentMCMC$B.
currentRes1 is local currentRes.
```

**Author(s)**

Andrew Edwards, Pacific Biological Station, Fisheries and Oceans Canada, Nanaimo BC

**See Also**

[plotVBcatch](#), [plotB2](#)

---

plotBox

*Plot Boxes using Quantiles*

---

**Description**

Produce box-and-whisker plot(s) of the given (grouped) values. This function is simply a modified version of [boxplot](#) that sets the whiskers to specified quantiles rather than 1.5 IRQ.

**Usage**

```
plotBox(x, ..., range = 1.5, width = NULL, varwidth = FALSE,
        notch = FALSE, outline = TRUE, names, plot = TRUE,
        border = par("fg"), col = NULL, log = "",
        pars = list(boxwex = 0.8, staplewex = 0.5, outwex = 0.5),
        horizontal = FALSE, add = FALSE, at = NULL,
        quants=c(0.025,0.25,0.5,0.75,0.975), outliers=FALSE)
```

**Arguments**

<code>formula</code>	a formula, such as <code>y ~ grp</code> , where <code>y</code> is a numeric vector of data values to be split into groups according to the grouping variable <code>grp</code> (usually a factor).
<code>data</code>	a data.frame (or list) from which the variables in <code>formula</code> should be taken.
<code>subset</code>	an optional vector specifying a subset of observations to be used for plotting.
<code>na.action</code>	a function which indicates what should happen when the data contain NAs. The default is to ignore missing values in either the response or the group.

<code>x</code>	for specifying data from which the boxplots are to be produced. Either a numeric vector, or a single list containing such vectors. Additional unnamed arguments specify further data as separate vectors (each corresponding to a component boxplot). <a href="#">NAs</a> are allowed in the data.
<code>...</code>	For the <code>formula</code> method, named arguments to be passed to the default method. For the default method, unnamed arguments are additional data vectors (unless <code>x</code> is a list when they are ignored), and named arguments are arguments and <a href="#">graphical parameters</a> to be passed to <code>bxp</code> in addition to the ones given by argument <code>pars</code> (and override those in <code>pars</code> ). Note that <code>bxp</code> may or may not make use of graphical parameters it is passed: see its documentation.
<code>range</code>	this determines how far the plot whiskers extend out from the box. If <code>range</code> is positive, the whiskers extend to the most extreme data point which is no more than <code>range</code> times the interquartile range from the box. A value of zero causes the whiskers to extend to the data extremes.
<code>width</code>	a vector giving the relative widths of the boxes making up the plot.
<code>varwidth</code>	if <code>varwidth</code> is <code>TRUE</code> , the boxes are drawn with widths proportional to the square-roots of the number of observations in the groups.
<code>notch</code>	if <code>notch</code> is <code>TRUE</code> , a notch is drawn in each side of the boxes. If the notches of two plots do not overlap this is ‘strong evidence’ that the two medians differ (Chambers <i>et al.</i> , 1983, p. 62). See <a href="#">boxplot.stats</a> for the calculations used.
<code>outline</code>	if <code>outline</code> is not true, the outliers are not drawn (as points whereas S+ uses lines).
<code>names</code>	group labels which will be printed under each boxplot. Can be a character vector or an <a href="#">expression</a> (see <a href="#">plotmath</a> ).
<code>boxwex</code>	a scale factor to be applied to all boxes. When there are only a few groups, the appearance of the plot can be improved by making the boxes narrower.
<code>staplewex</code>	staple line width expansion, proportional to box width.
<code>outwex</code>	outlier line width expansion, proportional to box width.
<code>plot</code>	if <code>TRUE</code> (the default) then a boxplot is produced. If not, the summaries which the boxplots are based on are returned.
<code>border</code>	an optional vector of colors for the outlines of the boxplots. The values in <code>border</code> are recycled if the length of <code>border</code> is less than the number of plots.
<code>col</code>	if <code>col</code> is non-null it is assumed to contain colors to be used to colour the bodies of the box plots. By default they are in the background colour.
<code>log</code>	character indicating if <code>x</code> or <code>y</code> or both coordinates should be plotted in log scale.
<code>pars</code>	a list of (potentially many) more graphical parameters, e.g., <code>boxwex</code> or <code>outpch</code> ; these are passed to <code>bxp</code> (if <code>plot</code> is true); for details, see there.
<code>horizontal</code>	logical indicating if the boxplots should be horizontal; default <code>FALSE</code> means vertical boxes.
<code>add</code>	logical, if true <code>add</code> boxplot to current plot.
<code>at</code>	numeric vector giving the locations where the boxplots should be drawn, particularly when <code>add = TRUE</code> ; defaults to <code>1:n</code> where <code>n</code> is the number of boxes.

quants	numeric vector of 5 quantiles to specify (i) the extent of the lowest whisker, (ii) the lower boundary of the box, (iii) the middle line of the box, (iv) the upper boundary of the box, and (v) the extent of the upper whisker.
outliers	logical: if TRUE show the outliers (but used primarily to suppress outliers when FALSE).

## Details

The generic function `boxplot` currently has a default method (`boxplot.default`) and a formula interface (`boxplot.formula`).

If multiple groups are supplied either as multiple arguments or via a formula, parallel boxplots will be plotted, in the order of the arguments or the order of the levels of the factor (see [factor](#)).

Missing values are ignored when forming boxplots.

## Value

List with the following components:

stats	a matrix, each column contains the extreme of the lower whisker, the lower hinge, the median, the upper hinge and the extreme of the upper whisker for one group/plot. If all the inputs have the same class attribute, so will this component.
n	a vector with the number of observations in each group.
conf	a matrix where each column contains the lower and upper extremes of the notch.
out	the values of any data points which lie beyond the extremes of the whiskers.
group	a vector of the same length as <code>out</code> whose elements indicate to which group the outlier belongs.
names	a vector of names for the groups.

## References

Becker, R. A., Chambers, J. M. and Wilks, A. R. (1988) *The New S Language*. Wadsworth & Brooks/Cole.

Chambers, J. M., Cleveland, W. S., Kleiner, B. and Tukey, P. A. (1983) *Graphical Methods for Data Analysis*. Wadsworth & Brooks/Cole.

Murrell, P. (2005) *R Graphics*. Chapman & Hall/CRC Press.

See also [boxplot.stats](#).

## See Also

[boxplot.stats](#) which does the computation, [bxp](#) for the plotting and more examples; and [stripchart](#) for an alternative (with small data sets).

plotChains

*Plot Cumulative Frequency of MCMC Chains***Description**

Plot cumulative frequency of  $n$  chains by partitioning one trace.  
(Modified from the function plotTracePOP.)

**Usage**

```
plotChains(mcmc, nchains=3, pdisc=0.1, axes=FALSE, same.limits=FALSE,
           between=list(x=axes,y=axes), div=1, span=1/4, log=FALSE,
           base=10, main=NULL, xlab=NULL, ylab=NULL, cex.main=1.2,
           cex.lab=1, cex.strip=0.8, cex.axis=0.8,
           las=0, tck=0.5, tick.number=5, lty.trace=1, lwd.trace=1,
           col.trace="grey", lty.median=1, lwd.median=1,
           col.median="black", lty.quant=2, lwd.quant=1,
           col.quant="black", plot=TRUE, probs=c(0.025, 0.5, 0.975), ...)
```

**Arguments**

mcmc	MCMC chain(s) as a vector, data frame or mcmc object.
ncchains	number of chains to create from one trace.
pdisc	proportion of the initial trace to discard before creating chains.
axes	whether axis values should be plotted.
same.limits	whether panels should have same x-axis limits.
between	list with x and y indicating panel spacing.
div	denominator to shorten values on the y axis.
span	smoothness parameter ( <b>not used</b> ).
log	whether values should be log-transformed.
base	logarithm base.
main	main title.
xlab	x-axis title.
ylab	y-axis title.
cex.main	size of main title.
cex.lab	size of axis labels.
cex.strip	size of strip labels.
cex.axis	size of tick labels.
las	orientation of tick labels: 0=parallel, 1=horizontal, 2=perpendicular, 3=vertical.
tck	tick mark length.
tick.number	number of tick marks.

<code>lty.trace</code>	line type of trace.
<code>lwd.trace</code>	line width of trace.
<code>col.trace</code>	colour of trace.
<code>lty.median</code>	line type of median.
<code>lwd.median</code>	line width of median.
<code>col.median</code>	colour of median.
<code>lty.quant</code>	line type of quantile trace.
<code>lwd.quant</code>	line width of quantile trace.
<code>col.quant</code>	colour of quantile trace.
<code>plot</code>	whether to draw plot.
<code>probs</code>	quantile values for quantile trace.
<code>...</code>	passed to <code>panel.trace</code> ( <b>not used</b> ).

**Value**

When `plot=TRUE`, a trellis plot is drawn and a data frame is returned, containing the data used for plotting. When `plot=FALSE`, a trellis object is returned.

**Note**

This idea stemmed from a discussion with PJS.

**Author(s)**

Rowan Haigh, Pacific Biological Station, Fisheries and Oceans Canada, Nanaimo BC

**See Also**

[plotTracePOP](#), [plotDensPOP](#)

---

plotCPUE

*Plot CPUE and Add Error Bars*

---

**Description**

Plot CPUE and fit with error bars.

**Usage**

```
plotCPUE(obj, main="", save=NULL, bar=1.96, yLim=NULL, ...)
```

**Arguments**

<code>obj</code>	data frame of CPUE indices from Awatea's results file ( <i>e.g.</i> , <code>currentRes\$CPUE</code> ).
<code>main</code>	title for figure
<code>save</code>	<b>not used</b>
<code>bar</code>	standard deviation of the normal distribution (1.96 is the approximate value of the 97.5 percentile point).
<code>yLim</code>	limits of the y-axis.
<code>...</code>	<b>not used</b>

**Value**

A postscript file:  
`CPUEser.eps` . . . CPUE indices with error bars.

**Note**

Copied code from `plotIndexNotLattice`.

**Author(s)**

Andrew Edwards, Pacific Biological Station, Fisheries and Oceans Canada, Nanaimo BC

**See Also**

[plotIndexNotLattice](#)

---

plotDensPOP	<i>Plot MCMC Density (AME Version)</i>
-------------	--

---

**Description**

Plot Markov-chain Monte Carlo density. This is an approximation of the posterior probability density function.

**Usage**

```
plotDensPOP(mcmc, probs=c(0.025,0.975), points=FALSE, axes=TRUE,
  same.limits=FALSE, between=list(x=axes,y=axes), div=1,
  log=FALSE, base=10, main=NULL, xlab=NULL, ylab=NULL,
  cex.main=1.2, cex.lab=1, cex.strip=0.8, cex.axis=0.7,
  las=0, tck=0.5, tick.number=5,
  lty.density=1, lwd.density=3, col.density="black",
  lty.median=2, lwd.median=1, col.median="darkgrey",
  lty.outer=3, lwd.outer=1, col.outer="darkgrey", pch="|",
  cex.points=1, col.points="black", plot=TRUE, MPD.height=0.04,...)
```

```

plotDensPOPpars(mcmc, probs=c(0.025,0.975), points=FALSE,
  axes=TRUE, same.limits=FALSE, between=list(x=axes,y=axes),
  div=1, log=FALSE, base=10, main=NULL, xlab=NULL, ylab=NULL,
  cex.main=1.2, cex.lab=1, cex.strip=0.8, cex.axis=0.7,
  las=0, tck=0.5, tick.number=5,
  lty.density=1, lwd.density=3, col.density="black",
  lty.median=2, lwd.median=1, col.median="darkgrey",
  lty.outer=3, lwd.outer=1, col.outer="darkgrey", pch="|",
  cex.points=1, col.points="black", plot=TRUE, MPD.height=0.04,...)

plotDensPOPparsPrior(mcmc, probs=c(0.025,0.975), points=FALSE,
  axes=TRUE, same.limits=FALSE, between=list(x=axes,y=axes),
  div=1, log=FALSE, base=10, main=NULL, xlab=NULL, ylab=NULL,
  cex.main=1.2, cex.lab=1, cex.strip=0.8, cex.axis=0.7,
  las=0, tck=0.5, tick.number=5,
  lty.density=1, lwd.density=3, col.density="black",
  lty.median=2, lwd.median=1, col.median="darkgrey",
  lty.outer=3, lwd.outer=1, col.outer="darkgrey", pch="|",
  cex.points=1, col.points="black", plot=TRUE, MPD.height=0.04,...)

```

### Arguments

<code>mcmc</code>	MCMC chain(s) as a vector, data frame or <code>mcmc</code> object.
<code>probs</code>	vector of outer quantiles to draw, besides the median.
<code>points</code>	whether data points should be plotted along the x axis.
<code>axes</code>	whether axis values should be plotted.
<code>same.limits</code>	whether panels should have same x-axis limits.
<code>between</code>	list with <code>x</code> and <code>y</code> indicating panel spacing.
<code>div</code>	denominator to shorten values on the x axis.
<code>log</code>	whether values should be log-transformed.
<code>base</code>	logarithm base.
<code>main</code>	main title.
<code>xlab</code>	x-axis label.
<code>ylab</code>	y-axis label.
<code>cex.main</code>	size of main title.
<code>cex.lab</code>	size of axis labels.
<code>cex.strip</code>	size of strip labels.
<code>cex.axis</code>	size of tick labels.
<code>las</code>	orientation of tick labels: 0=parallel, 1=horizontal, 2=perpendicular, 3=vertical.
<code>tck</code>	tick mark length.
<code>tick.number</code>	number of tick marks.
<code>lty.density</code>	line type of density curve.
<code>lwd.density</code>	line width of density curve.



<code>col.density</code>	colour of density curve.
<code>lty.median</code>	line type of median.
<code>lwd.median</code>	line width of median.
<code>col.median</code>	colour of median.
<code>lty.outer</code>	line type of outer quantiles.
<code>lwd.outer</code>	line width of outer quantiles.
<code>col.outer</code>	colour of outer quantiles.
<code>pch</code>	symbol for data points.
<code>cex.points</code>	size of data points.
<code>col.points</code>	colour of data points.
<code>plot</code>	whether to draw plot.
<code>MPD.height</code>	how far up to put MPD.
<code>...</code>	passed to <code>densityplot</code> and <code>panel.densityplot</code> .

### Details

The function `plotDensPOPpars` differs from `plotDensPOP` only by a few tweaks to the internal list object `myscales`.

The function `plotDensPOPparsPrior` adds the priors automatically.

### Value

When `plot=TRUE`, a trellis plot is drawn and a data frame is returned, containing the data used for plotting. When `plot=FALSE`, a trellis object is returned.

### Note

The `Args` function from the **gdata** package is recommended for reviewing the arguments, instead of `args`.

AME: edited `plotDens` function to have less whitesapce, not repeat x-axis labels, and make y-axes the same scales. Cannot just do it through the options.

For Recruits and Biomass, use `plotDensPOPpars` for parameters. Tried y-axes the same scales, but 1973–1975 are so narrow that they make all the others really small: `same.limits=TRUE`, `ylim=c(0, 0.0005)`.

### See Also

[xyplot](#) and [panel.densityplot](#) are the underlying drawing functions, and [densplot](#) is a similar non-trellis plot.

[plotTrace](#), [plotAuto](#), [plotCumu](#), and [plotSplom](#) are diagnostic plots.

[plotDens](#) and [plotQuant](#) are posterior plots.

[scapeMCMC-package](#) gives an overview of the package.

plotIndex2

*Plot Abundance Index (AME Version)***Description**

Plot scape model fit to abundance index data.

Revised version of Arni's function to confine plotting to data region.

**Usage**

```
plotIndex2(model, what="c", series=NULL, axes=TRUE, same.limits=FALSE,
  between=list(x=axes,y=axes), ylim=NULL, q=1, bar=1, log=FALSE,
  base=10, main="", xlab="", ylab="", cex.main=1.2, cex.lab=1,
  cex.strip=0.8, cex.axis=0.8, las=1, tck=c(1,0)/2,
  tick.number=5, lty.grid=3, col.grid="white", pch=16,
  cex.points=1.2, col.points="black", lty.lines=1, lwd.lines=4,
  col.lines="dimgrey", lty.bar=1, plot=TRUE, ...)
```

**Arguments**

model	fitted scape model containing element CPUE and/or Survey.
what	what to plot: "c"[ommercial] or "s"[urvey] abundance index.
series	vector of strings indicating which gears or surveys to plot (all by default).
axes	whether to plot axis values.
same.limits	whether panels should have same y-axis limits.
between	list with x and y indicating panel spacing.
ylim	vector with lower and upper y-axis limits.
q	denominator to scale the y axis, e.g. to vulnerable biomass. Similar to the div argument in plotN and plotB.
bar	extent of error bars relative to standard error.
log	whether to log-transform values.
base	logarithm base.
main	main title.
xlab	x-axis label.
ylab	y-axis label.
cex.main	size of main title.
cex.lab	size of axis labels.
cex.strip	size of strip labels.
cex.axis	size of tick labels.
las	orientation of tick labels: 0=parallel, 1=horizontal, 2=perpendicular, 3=vertical.
tck	tick mark length.

<code>tick.number</code>	number of tick marks.
<code>lty.grid</code>	line type of gridlines.
<code>col.grid</code>	color of gridlines.
<code>pch</code>	symbol for points.
<code>cex.points</code>	size of points.
<code>col.points</code>	color of points and error bars.
<code>lty.lines</code>	line type of main lines.
<code>lwd.lines</code>	line width of main lines.
<code>col.lines</code>	color of main lines.
<code>lty.bar</code>	line type of error bars.
<code>plot</code>	whether to draw plot.
<code>...</code>	passed to <code>xyplot</code> , <code>panel.xyplot</code> , and <code>panel.xYplot</code> .

**Value**

When `plot=TRUE`, a trellis plot is drawn and a data frame is returned, containing the data used for plotting. When `plot=FALSE`, a trellis object is returned.

**Note**

The `Args` function from the **gdata** package is recommended for reviewing the arguments, instead of `args`.

**See Also**

`xyplot`, `panel.xyplot`, and `panel.xYplot` are the underlying drawing functions.

`plotCA`, `plotCL`, `plotIndex`, and `plotLA` plot model fit and data.

`plotB`, `plotB2`, `plotN`, and `plotSel` plot derived quantities.

`scape-package` gives an overview of the package.

---

```
plotIndexNotLattice
```

*Plot Survey Indices*

---

**Description**

Plot index series with error bars. Create postscript files automatically.

**Usage**

```
plotIndexNotLattice(obj, objCPUE, main="", save=NULL, bar=1.96, ...)
```

**Arguments**

<code>obj</code>	data frame of survey indices from Awatea's results file ( <i>e.g.</i> , <code>currentRes\$Survey</code> ).
<code>objCPUE</code>	data frame of CPUE indices from Awatea's results file ( <i>e.g.</i> , <code>currentRes\$CPUE</code> ).
<code>main</code>	title for figure
<code>save</code>	<b>not used</b>
<code>bar</code>	standard deviation of the normal distribution (1.96 is the approximate value of the 97.5 percentile point).
<code>...</code>	<b>not used</b>

**Value**

Four postscript files:

`survIndSer.eps` . . . each survey panel focuses on the years of the survey;  
`survIndSer2.eps` . . . each panel uses a fixed set of years that span all surveys;  
`survIndSer3.eps` . . . one panel showing all series normalised to their means;  
`survIndSer4.eps` . . . compares normalised GIG series with CPUE series.

**Author(s)**

Andrew Edwards, Pacific Biological Station, Fisheries and Oceans Canada, Nanaimo BC

**See Also**

[plt.idx](#)

---

plotRmcmcPOP

*Plot Recruitment Posterior Quantiles*

---

**Description**

Plot recruitment posteriors quantiles as one graph over time.

**Usage**

```
plotRmcmcPOP( obj,
               p=c(0.025,0.25,0.5,0.75,0.975),
               xyType="quantBox",
               lineType=c(3,2,1,2,3),
               refLines=NULL, xLim=NULL, yLim=NULL,
               userPrompt=FALSE, save=T, tcl.val=-0.2,
               yaxis.by=10000, yLab="Recruitment", ...)
```

**Arguments**

obj	MCMC data frame of $R$ (currentMCMC\$R).
p	quantiles to use in quantBox.
xyType	type of plot (currently only uses quantBox).
lineType	line types to use in quantBox.
refLines	reference lines to add to plot.
xLim	limits of the x-axis.
yLim	limits of the y-axis.
userPrompt	<b>not used</b>
save	<b>not used</b>
tcl.val	tick length.
yaxis.by	increments along the y-axis to place tick marks.
yLab	label for the y-axis.
...	additional arguments passed to the function rect.

**Note**

AME: Plot recruitment posteriors quantiles as one graph over time.  
 Already have the full posterior densities done.  
 Using plotBmcmcPOP as template, but will be simpler as no extra stuff. Probably not simplifying down as much as could due to time constraints.  
 Adding yLab and then using for exploitation plot also.

**Author(s)**

Andrew Edwards, Pacific Biological Station, Fisheries and Oceans Canada, Nanaimo BC

**See Also**

[plotBmcmcPOP](#), [plotB2](#)

---

plotSnail

---

*Plot Snail Trails of Exploitation vs. Biomass*


---

**Description**

Plot the historical progression of the ratio  $u_t/u_{MSY}$  against  $B_t/B_{MSY}$ .

**Usage**

```
plotSnail(BoverBmsy, UoverUmsy, p=c(0.1, 0.9),
          xLim=NULL, yLim=NULL, Lwd=2)
```

**Arguments**

BoverBmsy	numeric matrix of $B_t$ over $B_{MSY}$ .
UoverUmsy	numeric matrix of $u_t$ over $u_{MSY}$ .
p	quantiles to show the bulk of the distribution.
xLim	limits of the x-axis.
yLim	limits of the y-axis.
Lwd	line width of the snail trail.

**Details**

The graph attempts to show the time history of the exploitation rate compared to the spawning biomass using a precautionary framework recast in Bayesian terms.

**Note**

The term *snail trail* comes from PJS.

**Author(s)**

Andrew Edwards, Pacific Biological Station, Fisheries and Oceans Canada, Nanaimo BC

**See Also**

[plotBVBnorm](#)

---

plotTracePOP	<i>Plot MCMC Traces (AME Version)</i>
--------------	---------------------------------------

---

**Description**

Plot Markov-chain Monte Carlo traces. This is a diagnostic plot for deciding whether a chain shows unwanted trends by showing the trace of moving quantiles. (Modified from the **scapeMCMC** function `plotTrace`.)

**Usage**

```
plotTracePOP(mcmc, axes=FALSE, same.limits=FALSE,
  between=list(x=axes,y=axes), div=1, span=1/4, log=FALSE,
  base=10, main=NULL, xlab=NULL, ylab=NULL, cex.main=1.2,
  cex.lab=1, cex.strip=0.8, cex.axis=0.8,
  las=0, tck=0.5, tick.number=5, lty.trace=1, lwd.trace=1,
  col.trace="grey", lty.median=1, lwd.median=1,
  col.median="black", lty.quant=2, lwd.quant=1,
  col.quant="black", plot=TRUE, probs=c(0.025, 0.5, 0.975), ...)
```

**Arguments**

<code>mcmc</code>	MCMC chain(s) as a vector, data frame or <code>mcmc</code> object.
<code>axes</code>	whether axis values should be plotted.
<code>same.limits</code>	whether panels should have same x-axis limits.
<code>between</code>	list with <code>x</code> and <code>y</code> indicating panel spacing.
<code>div</code>	denominator to shorten values on the y axis.
<code>span</code>	smoothness parameter ( <b>not used</b> ).
<code>log</code>	whether values should be log-transformed.
<code>base</code>	logarithm base.
<code>main</code>	main title.
<code>xlab</code>	x-axis title.
<code>ylab</code>	y-axis title.
<code>cex.main</code>	size of main title.
<code>cex.lab</code>	size of axis labels.
<code>cex.strip</code>	size of strip labels.
<code>cex.axis</code>	size of tick labels.
<code>las</code>	orientation of tick labels: 0=parallel, 1=horizontal, 2=perpendicular, 3=vertical.
<code>tck</code>	tick mark length.
<code>tick.number</code>	number of tick marks.
<code>lty.trace</code>	line type of trace.
<code>lwd.trace</code>	line width of trace.
<code>col.trace</code>	colour of trace.
<code>lty.median</code>	line type of median.
<code>lwd.median</code>	line width of median.
<code>col.median</code>	colour of median.
<code>lty.quant</code>	line type of quantile trace.
<code>lwd.quant</code>	line width of quantile trace.
<code>col.quant</code>	colour of quantile trace.
<code>plot</code>	whether to draw plot.
<code>probs</code>	quantile values for quantile trace.
<code>...</code>	passed to <code>panel.trace</code> ( <b>not used</b> ).

**Value**

When `plot=TRUE`, a trellis plot is drawn and a data frame is returned, containing the data used for plotting. When `plot=FALSE`, a trellis object is returned.

**Note**

The `Args` function from the **gdata** package is recommended for reviewing the arguments, instead of `args`.

**See Also**

`xyplot` and `panel.loess` are the underlying drawing functions, and `traceplot` is a similar non-trellis plot.

`plotTracePOP`, `plotAuto`, `plotCumu`, and `plotSplom` are diagnostic plots.

`plotDensPOP`, `plotDens`, `plotQuant`, and `plotChains` are posterior plots.

`scapeMCMC-package` gives an overview of the **scapeMCMC** package.

---

plotVBcatch

*Plot Vulnerable Biomass*


---

**Description**

Plot vulnerable biomass from posterior as boxplots, and add catch bars on same graph.

**Usage**

```
plotVBcatch( obj, currentRes1=currentRes,
             p=c(0.025,0.25,0.5,0.75,0.975),
             xyType="quantBox",
             lineType=c(3,2,1,2,3),
             refLines=NULL, xLim=NULL, yLim=NULL,
             userPrompt=FALSE, save=T, xLab = c(1939,1939),
             yLab=c(10000,220000),
             textLab=c("catch","vulnerable"),
             yaxis.by=10000, tcl.val=-0.2, ...)
```

**Arguments**

<code>obj</code>	MCMC data frame of $B$ ( <code>currentMCMC\$B</code> ).
<code>currentRes1</code>	list/scape object created by <code>importCol2</code> .
<code>p</code>	quantiles to use in <code>quantBox</code> .
<code>xyType</code>	type of plot (currently only uses <code>quantBox</code> ).
<code>lineType</code>	line types to use in <code>quantBox</code> .
<code>refLines</code>	reference lines to add to plot.
<code>xLim</code>	limits of the x-axis.
<code>yLim</code>	limits of the y-axis.
<code>userPrompt</code>	<b>not used</b>
<code>save</code>	<b>not used</b>
<code>xLab</code>	x-coordinates for labels.
<code>yLab</code>	y-coordinates for labels.
<code>textLab</code>	text labels to display on plot.
<code>yaxis.by</code>	increments along the y-axis to place tick marks.
<code>tcl.val</code>	tick length.
<code>...</code>	additional arguments passed to the function <code>rect</code> .



**Note**

AME: This function is essentially a tweak of plotBmcmcPOP.

**Author(s)**

Andrew Edwards, Pacific Biological Station, Fisheries and Oceans Canada, Nanaimo BC

**See Also**

[plotBmcmcPOP](#), [plotB2](#)

---

plotVBnorm

---

*Plot Spawning and Vulnerable Biomass Relative to Virgin*


---

**Description**

Plot spawning and vulnerable biomass boxplots relative to virgin levels  $B_0$  and  $V_0$ , respectively.

**Usage**

```
plotVBnorm( mcmcObj,
            p=c(0.025,0.25,0.5,0.75,0.975),
            xyType="quantBox",
            lineType=c(3,2,1,2,3),
            refLines=NULL, xLim=NULL, yLim=NULL,
            userPrompt=FALSE, save=T, xLeg=0.7, yLeg=0.9,
            yaxis.by=0.02, tcl.val=-0.2,
            B.col="black", VB.col="black", ...)

plotBVBnorm(mcmcObj,
            p=c(0.025,0.25,0.5,0.75,0.975),
            xyType="quantBox",
            lineType=c(3,2,1,2,3),
            refLines=NULL, xLim=NULL, yLim=NULL,
            userPrompt=FALSE, save=T, xLeg=0.7, yLeg=0.9,
            yaxis.by=0.02, tcl.val=-0.2,
            B.col="black", VB.col="black", ...)
```

**Arguments**

mcmcObj	MCMC list object (currentMCMC).
p	quantiles to use in quantBox.
xyType	type of plot (currently only uses quantBox).
lineType	line types to use in quantBox.
refLines	reference lines to add to plot.

xLim	limits of the x-axis.
yLim	limits of the y-axis.
userPrompt	<b>not used</b>
save	<b>not used</b>
xLeg	x-coordinate for legend.
yLeg	y-coordinate for legend.
yaxis.by	increments along the y-axis to place tick marks.
tcl.val	tick length.
B.col	colour for spawning biomass.
VB.col	colour for vulnerable biomass.
...	<b>not used</b>

**Note**

AME: tried in separate file, but then changed that to lattice and wouldn't be good format for Arni's boxplots.

Based on plotVBcatch (tweaking some).

currentRes1 is local currentRes.

**Author(s)**

Andrew Edwards, Pacific Biological Station, Fisheries and Oceans Canada, Nanaimo BC

**See Also**

[plotVBcatch](#), [plotBmcmcPOP](#)

---

plt.ageResidsPOP     *Plot Model Residuals*

---

**Description**

Plot model residuals as boxplots or qq-plots.

**Usage**

```
plt.ageResidsPOP (obj, ages=c(2,60), pct=c(5,25,50,75,95), main=NULL)
plt.ageResidsqqPOP (obj, ages=c(2,60), pct=c(5,25,50,75,95), main=NULL)
plt.yearResidsPOP (obj, ages=c(2,60), pct=c(5,25,50,75,95),
                  main=NULL, fill.in=TRUE, ... )
plt.cohortResids (obj, ages=c(2,59), pct=c(5,25,50,75,95), main=NULL)
```

**Arguments**

obj	output from <code>stdRes.CA</code> .
ages	age classes to plot.
pct	quantiles to show in boxplot or qq-plot.
main	title for plot if desired.
fill.in	logical: if TRUE, add missing years to boxplot.
...	additional arguments for <code>boxplot</code> .

**Details**

`plt.ageResidsPOP...` plot age class residuals as boxplots.  
`plt.ageResidsqqPOP...` plot age class residuals as qq-plot.  
`plt.yearResidsPOP...` plot age residuals by year as boxplots.  
`plt.cohortResids...` plot age residuals by cohort as boxplots.

**Note**

Some trouble noted adding text and legend.

**Author(s)**

Andrew Edwards, Pacific Biological Station, Fisheries and Oceans Canada, Nanaimo BC

**See Also**

[plt.allTraces](#), [plt.expRate](#), [plt.idx](#), [plt.mcmcGraphs](#), [plotIndexNotLattice](#),  
[plotChains](#), [plotCPUE](#)

---

plt.allTraces	<i>Plot MCMC Traces</i>
---------------	-------------------------

---

**Description**

Plot traces from MCMC samples.

**Usage**

```
plt.allTraces(obj, bioYrList=NULL, recYrList=NULL, save=TRUE)
```

**Arguments**

obj	vector of MCMC samples.
bioYrList	years to plot spawning biomass traces.
recYrList	years to plot recruitment traces.
save	logical: if TRUE, save figure to a raster file (.jpg).

**Note**

Appears to be some figure requested by PJS.

**Author(s)**

Andrew Edwards, Pacific Biological Station, Fisheries and Oceans Canada, Nanaimo BC

**See Also**

[plotChains](#)

---

plt.expRate

*Plot Exploitation Rate*

---

**Description**

Plot exploitation rate against year.

**Usage**

```
plt.expRate(obj, yLim=c(0,0.5), xLim=c(1954,2005))
```

**Arguments**

obj	an object from <code>load.allResFiles</code> .
yLim	limits of the y-axis.
xLim	limits of the x-axis.

**Details**

Simple points and lines plot.

**Author(s)**

Andrew Edwards, Pacific Biological Station, Fisheries and Oceans Canada, Nanaimo BC

**See Also**

[plt.idx](#)

---

plt.idx	<i>Plot Survey Index Residuals</i>
---------	------------------------------------

---

**Description**

Plot the survey index residuals as a quantile-quantile plot (see [qqnorm](#)).

**Usage**

```
plt.idx(obj, main="Residuals", save=NULL, ...)
```

**Arguments**

obj	a data frame with columns Year, stdRes, and Fit.
main	title for the plot.
save	logical: if TRUE, save the figure to a raster file (.png).
...	<b>not used</b>

**Details**

QQ-plots show sample quantiles vs. theoretical quantiles.

**Note**

The save option has been disabled for some reason.

**Author(s)**

Andrew Edwards, Pacific Biological Station, Fisheries and Oceans Canada, Nanaimo BC

**See Also**

[plt.stdResids](#), [plotIndexNotLattice](#)

---

plt.mpdGraphs	<i>Plot a Set of Figures for MPD and MCMC</i>
---------------	---

---

**Description**

Plot a set of postscript figures .eps for the MPD (mode of the posterior distribution) and MCMC (Monte Carlo Markoff Chain) results.

**Usage**

```
plt.mpdGraphs(obj, save = FALSE)
plt.mcmcGraphs(mcmcObj, projObj, save=FALSE, xlimrec=c(0, 2e+05))
```

**Arguments**

obj	an Awatea results object (e.g., currentRes).
mcmcObj	an Awatea MCMC object (e.g., currentMCMC).
projObj	an Awatea projected biomass object (e.g., currentProj).
save	<b>not used</b>
xlimrec	range for recruitments ( <b>not used</b> ).

**Details**

Creates a whole heap o postscript files.

**Value**

**plt.mpdGraphs** creates the following postscript files:

exploit.eps.....annual exploitation rate,  
 recruits.eps.....annual recruitment at age 1,  
 selectivity.eps.....selectivity curves for commercial gear(s) and survey(s),  
 ageComm.eps.....fits to annual commercial age composition (panel plots),  
 ageSurv.eps.....fits to annual survey age composition (panel plots),  
 survIndSer.eps.....four figures of survey indices (calls [plotIndexNotLattice](#)),  
 CPUEser.eps.....CPUE indices with error bars (calls [plotCPUE](#)),  
 commAgeResids.eps.....standardised residuals for commercial gear,  
 survAgeResidsSer.eps...standardised residuals for surveys,  
 meanAge.eps.....mean age for catch and surveys,  
 stockRecruit.eps.....stock recruitment function.

**plt.mcmcGraphs** creates the following postscript files:

recruitsMCMC.eps.....boxplots of annual MCMC recruitment,  
 exploitMCMC.eps.....boxplots of annual MCMC exploitation rate,  
 pdfBiomass.eps.....density panel plots of annual female spawning biomass,  
 pdfRecruitment.eps.....density panel plots of annual recruitment,  
 selectivityMCMC.eps....**not currently implemented**,  
 traceRecruits.eps.....panel plots of annual recruitment traces with running quantiles,  
 traceBiomass.eps.....panel plots of annual spawning biomass with running quantiles,  
 traceParams.eps.....panel plots of parameter traces with running quantiles,  
 splitChain.eps.....panel plots of cumulative parameter estimate chains,  
 VBcatch.eps.....boxplots of annual vulnerable biomass and barplots of catch,  
 BVBnorm.eps.....spawning and vulnerable biomass relative to their virgin levels,  
 Bproj.eps.....boxplots of spawning biomass – MCMCs and projections,  
 snail.eps.....time series of  $u_t/u_{MSY}$  vs.  $B_t/B_{MSY}$ ,  
 pairs[1,2,3].eps.....pairs plot of parameter MCMC samples.

**Note**

The function `plt.mpdGraphs` needs some cleaning up. For example, there appears to be extensive reliance on the object `currentRes` whereas the function should be using `obj`, which is the actual results object within the function.

The same is true for `plt.mcmcGraphs` where the global objects `currentMCMC` and `currentProj` are use within the function rather than using the local objects `mcmcObj` and `projObj`.

### Author(s)

Andrew Edwards, Pacific Biological Station, Fisheries and Oceans Canada, Nanaimo BC

### See Also

[plt.mcmcGraphs](#), [plotIndexNotLattice](#), [plotCPUE](#), [plotChains](#), [plotSnail](#)

---

plt.numR

*Plot Numbers at Age at Equilibrium*

---

### Description

Plot numbers at age at equilibrium. Plot recruitment (age 1).

### Usage

```
plt.numR(obj, minYr = NULL)
```

### Arguments

<code>obj</code>	an Awatea results object from <code>load.allResFiles</code> .
<code>minYr</code>	minimum year to display in plot.

### Note

Not sure what this plot is used for (RH).

### Author(s)

Andrew Edwards, Pacific Biological Station, Fisheries and Oceans Canada, Nanaimo BC

### See Also

[plotRmcmcPOP](#)

---

plt.quantBio      *Plot Quantile Boxes of MCMC and Projected Biomass*


---

### Description

Plots MCMC and projected biomass as quantile boxes, the former in black, the latter in red.

### Usage

```
plt.quantBio(obj, projObj=NULL, policy=NULL,
             p=c(0.025,0.25,0.5,0.75,0.975), xyType="lines",
             lineType=c(3,2,1,2,3), refLines=NULL,
             xLim=NULL, yLim=NULL, userPrompt=FALSE, save=T)
```

```
plt.quantBioBB0(obj, projObj=NULL, policy=NULL,
                p=c(0.025,0.25,0.5,0.75,0.975), xyType="lines",
                lineType=c(3,2,1,2,3), refLines=NULL,
                xLim=NULL, yLim=NULL, userPrompt=FALSE, save=T,
                main="", cex.main="", tcl.val=-0.2,
                xaxis.by=1, yaxis.by=10000, xaxis.lab="Year",
                yaxis.lab="Spawning biomass")
```

### Arguments

obj	an Awatea MCMC object (e.g., currentMCMC).
projObj	an Awatea projected biomass object (e.g., currentProj).
policy	numeric vector specifying catch policy.
p	quantiles to use from the biomass samples.
xyType	string specifying type of plot.
lineType	line types for the quantiles if xyType="lines".
refLines	reference points.
xLim	limits of the x-axis.
yLim	limits of the y-axis.
userPrompt	logical: if TRUE prompts user before figure is drawn.
save	logical: if TRUE save figure as a raster file .png.
main	character string specifying a title for the plot.
cex.main	font size for figure title.
tcl.val	tick length.
xaxis.by	tick mark intervals for x-axis.
yaxis.by	tick mark intervals for y-axis.
xaxis.lab	label for x-axis.
yaxis.lab	label for y-axis.



**Value**

List of the reconstructed (MCMC) and projected results.

**Note**

`plt.quantBioBB0` performs similarly as for `plt.quantBio` but uses  $B_t/B_0$  instead of  $B_t$ .

**Author(s)**

Andrew Edwards, Pacific Biological Station, Fisheries and Oceans Canada, Nanaimo BC

**See Also**

[plotBmcmcPOP](#), [plotVBcatch](#), [plotBVBnorm](#), [plotRmcmcPOP](#)

---

<code>plt.ssbVbCatch</code>	<i>Plot Annual Spawning and Vulnerable Biomass</i>
-----------------------------	--

---

**Description**

Plot MPD values of spawning biomass (SB) and vulnerable biomass(VB), as well as catch, against year.

**Usage**

```
plt.ssbVbCatch(obj, x1=1966, xLim=c(1954,2005), yLim=c(0,25000))
```

**Arguments**

<code>obj</code>	an Awatea results object from <code>load.allResFiles</code> .
<code>x1</code>	year to start plotting SB and VB lines.
<code>xLim</code>	limits of the x-axis.
<code>yLim</code>	limits of the y-axis.

**Note**

This analysis uses the MPD (mode of the posterior distribution) values for  $B$  and  $V$ .

**Author(s)**

Andrew Edwards, Pacific Biological Station, Fisheries and Oceans Canada, Nanaimo BC

**See Also**

[load.allResFiles](#), [get.resFile](#), [plt.expRate](#)

---

`plt.stdResids`
*Plot Diagnostics for Standardised Residuals*


---

### Description

Plot standardised residuals against year, fitted value, and theoretical residuals.

### Usage

```
plt.stdResids(obj, pct=c(5, 25, 50, 75, 95),
              main=NULL, yLim=NULL, xLim=xLim)
```

### Arguments

<code>obj</code>	a data frame with columns <code>Year</code> , <code>stdRes</code> , and <code>Fit</code> .
<code>pct</code>	percentiles to display as horizontal lines on the quantile-quantile plot.
<code>main</code>	title for the figure.
<code>yLim</code>	limits of the y-axis.
<code>xLim</code>	limits of the x-axis.

### Details

Figure provides three panels of standardised residuals vs.  
(i) years, (ii) fitted or predicted values, and (iii) theoretical quantiles.

### Author(s)

Andrew Edwards, Pacific Biological Station, Fisheries and Oceans Canada, Nanaimo BC

### See Also

[`plt.ssbVbCatch`](#)

---

`readAD`
*Read ADMB Input and Create AWATEA Class Object*


---

### Description

Read the ADMB input file and create an AWATEA class object.

### Usage

```
readAD(txt)
```

**Arguments**

`txt`                      string name of an Awatea input file.

**Details**

The Awatea input file contains headers (lines prefixed with "#") and data that are read sequentially into the model by the binary executable `Awatea.exe`.

**Value**

An AWATEA class cobject with the slots:

`txtnam` . . . . . character: name of the input file,  
`input` . . . . . character: vector of strings that are the lines of the input file,  
`vlst` . . . . . list: each line of the input file with a label specifying line number and  
. . . . . indicating whether the line is a Comment or Data,  
`dnam` . . . . . character: vector of strings specifying data contents labelled by line number,  
`nvars` . . . . . numeric: number of data variables,  
`vdesc` . . . . . character: vector of strings specifying data contents labelled by  
. . . . . variable number (e.g., `v001`),  
`vars` . . . . . list: numeric values of data labelled by variable number,  
`gcomm` . . . . . character: vector of comments labelled by line number,  
`vcomm` . . . . . character: vector of variable names labelled by line number,  
`output` . . . . . list: Awatea results file imported by function `importRes`,  
`reweight` . . . list: empty (later populated by function `reweight`).

**Author(s)**

Rowan Haigh, Pacific Biological Station, Fisheries and Oceans Canada, Nanaimo BC.

**References**

Hilborn, R., Maunder, M., Parma, A., Ernst, B. Payne, J., and Starr, P. (2003) Coleraine: a generalized age-structured stock assessment model. School of Aquatic and Fishery Sciences, University of Washington, 54 p.

**See Also**

`runADMB`, `reweight`

---

`refPoints`


---

*Calculate Reference Points*


---

**Description**

Calculate reference points relative to either  $B_{MSY}$  or  $B_0$ .

**Usage**

```
refPoints(mcmcObj=currentMCMC, projObj=currentProj,
          msyObj=currentMSY, refLevels=c(0.4,0.8,1))

refPointsB0(mcmcObj=currentMCMC, projObj=currentProj,
            B0Obj=B0.MCMC, refLevels=B0refLevels, refNames=B0refNames)
```

**Arguments**

mcmcObj        MCMC list object (*e.g.*, currentMCMC).  
projObj        projected biomass list object (*e.g.*, currentProj).  
msyObj        MSY list object (*e.g.*, currentMSY).  
refLevels      reference levels relative to  $B_{MSY}$  (or  $B_0$ ).  
B0Obj        vector of  $B_0$  MCMC values (*e.g.*, B0.MCMC).  
refNames      names of the  $B_0$  reference levels refLevels.

**Value**

List of reference points relative to either  $B_{MSY}$  or  $B_0$ .

**Note**

Call from Sweave as `refPoints()` or, in full:

```
refPoints(currentMCMC, currentProj, currentMSY, refLevels=c(0.4,0.8,1))
```

**Author(s)**

Andrew Edwards, Pacific Biological Station, Fisheries and Oceans Canada, Nanaimo BC

**See Also**

[calc.refVal](#)

---

reweight

---

*Reweight Abundance and Composition Data*


---

**Description**

Weight the abundance data by adjusting survey and CPUE CVs, and weight the composition data by adjusting the effective sample size  $N$ .

**Usage**

```
reweight(obj, cvpro=FALSE, mean.age=TRUE, ...)
```

**Arguments**

<code>obj</code>	an AWATEA class object created initially by <code>readAD</code> .
<code>cvpro</code>	CV process error added to CV observation error: $c_t = \sqrt{c_o^2 + c_p^2}$ ; if <code>FALSE</code> index CVs are reweighted using the standard deviation of normalized residuals.
<code>mean.age</code>	logical: if <code>TRUE</code> , use mean-age residuals to reweight the effective $N$ for the age composition data (see Francis 2011); if <code>FALSE</code> , reweight $N$ using $\Sigma(P(1 - P))/\Sigma(O - P)^2$ , where $O$ = observed proportions-at age and $P$ = predicted/fitted proportions-at-age.
<code>...</code>	additional arguments to <code>reweight</code> .

**Details**

For the reweight to work, a corresponding Awatea results file (`.res`) with the same prefix as the input file must be available in the working directory before calling `readAD`. This will populate the output slot with fitted data that the reweight needs.

**Value**

An AWATEA class object with the slots outlined in `readAD` with the following slot populated by this function:

```
reweight...list of reweight results:
..nrwt....the number of the current reweighting,
..survey...survey indices with CV values (observed, fitted, normalised residuals, reweighted),
..cpue....if used in the model, CPUE indices with CV values (as above),
..wNcpa....reweighted effective  $N$  for commercial compositions (proportions-at-age),
..wNspa....reweighted effective  $N$  for survey compositions,
..SDNR....standard deviation of normalised residuals for abundance and composition data,
..wj.....weights for composition data from a mean-age weighted calculation (Francis 2011).
```

**Author(s)**

Rowan Haigh, Pacific Biological Station, Fisheries and Oceans Canada, Nanaimo BC.

**References**

Francis, R.I.C.C. (2011, in press) Data weighting in statistical fisheries stock assessment models. *Canadian Journal of Fisheries and Aquatic Sciences*.

**See Also**

[runADMB](#), [readAD](#), [stdRes.CA](#), [stdRes.index](#)

runADMB

*Run AD Model Builder Code for Awatea***Description**

Run compiled AD Model Builder code called *Awatea* to reconstruct a population trajectory for a marine fish stock.

**Usage**

```
runADMB(filename.ext, wd=getwd(), strSpp="YMR", runNo=25, rwtNo=0,
        doMPD=FALSE, N.reweight=0, cvpro=FALSE, mean.age=TRUE,
        doMCMC=FALSE, mcmc=1e+06, mcsave=1000, ADargs=NULL, verbose=FALSE,
        doMSY=FALSE, msyMaxIter=15000, msyTolConv=0.01,
        endStrat=0.301, stepStrat=0.001, ...)
```

**Arguments**

<code>filename.ext</code>	character file name including its extension.
<code>wd</code>	character string specifying the working directory, if different from the current working directory.
<code>strSpp</code>	string 3-letter code of the species.
<code>runNo</code>	the model run number.
<code>rwtNo</code>	the reweight number.
<code>doMPD</code>	logical: if TRUE, perform an MPD analysis.
<code>N.reweight</code>	the number of reweights to perform in the MPD analysis.
<code>cvpro</code>	CV process error added to CV observation error: $c_t = \sqrt{c_o^2 + c_p^2}$ ; if FALSE index CVs are reweighted using the standard deviation of normalized residuals.
<code>mean.age</code>	logical: if TRUE, use mean-age residuals to reweight the effective $N$ for the age composition data (see Francis 2011); if FALSE, reweight $N$ using $\Sigma(P(1 - P))/\Sigma(O - P)^2$ , where $O$ = observed proportions-at age and $P$ = predicted/fitted proportions-at-age.
<code>doMCMC</code>	logical: if TRUE, perform an MCMC analysis.
<code>mcmc</code>	number of MCMC iterations to perform.
<code>mcsave</code>	frequency of MCMC iterations to save.
<code>ADargs</code>	additional arguments for a call to <i>Awatea</i> .
<code>verbose</code>	logical: if TRUE, spew <i>Awatea</i> messages to the R console.
<code>doMSY</code>	logical: if TRUE, perform an MSY analysis.
<code>msyMaxIter</code>	maximum iterations for the MSY calculations.
<code>msyTolConv</code>	tolerance for convergence in the MSY calculations.
<code>endStrat</code>	maximum fishing mortality for the MSY analysis.
<code>stepStrat</code>	fishing mortality step size for the MSY analysis.
<code>...</code>	additional arguments (not currently used for any purpose).

## Details

This function is primarily used to automate MPD reweightings and to perform the MSY calculations. The MCMCs are better run from a command line console on a super computer as they generally require > 12 h to complete.

Note that once an MCMC has been created, the user can run various projections separately using `awatea -ind filename.ext -mceval` on the command line.

## Author(s)

Rowan Haigh, Pacific Biological Station, Fisheries and Oceans Canada, Nanaimo BC.

## References

Edwards, A.M., Starr, P.J., and Haigh, R. (2010, in revision). Stock assessment for Pacific Ocean Perch (*Sebastes alutus*) in Queen Charlotte Sound, British Columbia. *Canadian Science Advisory Secretariat, Research Document*.

Edwards, A.M., Haigh, R., and Starr, P.J. (2011, in revision). Stock assessment and recovery potential assessment for Yellowmouth Rockfish (*Sebastes reedi*) along the Pacific coast of Canada. *Canadian Science Advisory Secretariat, Research Document*.

Francis, R.I.C.C. (2011, in press) Data weighting in statistical fisheries stock assessment models. *Canadian Journal of Fisheries and Aquatic Sciences*.

Hilborn, R., Maunder, M., Parma, A., Ernst, B. Payne, J., and Starr, P. (2003) Coleraine: a generalized age-structured stock assessment model. School of Aquatic and Fishery Sciences, University of Washington, 54 p.

## See Also

`importRes`, `readAD`, `reweight`, `runSweave`

---

runMCMC

*Wrapper to Function <runSweaveMCMC>*

---

## Description

This small utility function simply provides a wrapper to the function `runSweaveMCMC` so that multiple documents can be built at once.

## Usage

```
runMCMC(runs=7, rewt=0:6, cpue=FALSE, estM=TRUE)
```

**Arguments**

runs	the run number(s).
rewts	the reweight number(s).
cpue	logical: if TRUE, retain the CPUE figures in the Sweave file.
estM	logical: if TRUE the routine will assume natural mortality $M$ was estimated in the run, and the Sweave code uses the parameters "M_1" and "M_2"; if FALSE, the function removes these parameters from the Sweave file.

**Details**

Essentially loops through `runSweaveMCMC` using `(i in runs)` and `(j in rewt)`.

**Value**

Produces multiple Sweave documents and PDF files for MCMCs.

**Note**

Sweave files can be found in the library directory:

```
.../R/.../library/PBSawatea/snw/ymrrun-masterMCMC.Snw
.../R/.../library/PBSawatea/snw/ymrrun-masterMCMC20yrProjs.Snw
```

**Author(s)**

Rowan Haigh, Pacific Biological Station, Fisheries and Oceans Canada, Nanaimo BC

**See Also**

`runSweaveMCMC`, `runSweave`, `runMPD`

---

runMPD

*Wrapper to Function <runSweave>*

---

**Description**

This small utility function simply provides a wrapper to the function `runSweave` so that multiple documents can be built at once.

**Usage**

```
runMPD(runs=1, rewt=0:6, cpue=FALSE)
```

**Arguments**

runs	the run number(s).
rewts	the reweight number(s).
cpue	logical: if TRUE, retain the CPUE figures in the Sweave file.



**Details**

Essentially loops through `runSweave` using `(i in runs)` and `(j in rewtS)`.

**Value**

Produces multiple Sweave documents and PDF files.

**Note**

A recent Seave file called `ymrrun-master.Snw` can be found in the library directory:  
`.../R/.../library/PBSawatea/snw`

**Author(s)**

Rowan Haigh, Pacific Biological Station, Fisheries and Oceans Canada, Nanaimo BC

**See Also**

`runSweave`, `runSweaveMCMC`, `runMCMC`

---

`runSweave`


---

*Run Customised Sweave Files for Awatea MPD Runs*


---

**Description**

Create and run customised Sweave files for Awatea MPD runs.

**Usage**

```
runSweave(wd=getwd(), cpue=FALSE, strSpp="YMR",
          filename="input25-ymr.txt", runNo=25, rwtNo=0,
          running.awatea=0, Nsurvey=5)
```

**Arguments**

<code>wd</code>	working directory in which Awatea input and master Sweave files occur.
<code>cpue</code>	logical: if <code>TRUE</code> the routine will leave calls to CPUE figures in the Sweave file, otherwise, they are removed.
<code>strSpp</code>	three-letter code that identifies the species.
<code>filename</code>	name of Awatea input file.
<code>runNo</code>	the run number that identifies a unique combination of input values.
<code>rwtNo</code>	the reweight number of the MPD run to build and collate figures.
<code>running.awatea</code>	numeric: if 0 load a previous <code>.rep</code> file, if 1 re-run Awatea
<code>Nsurvey</code>	number of surveys specified in the input file; controls image creation and place-holders.

### Details

The values specified by the arguments (or derived variables) are directly substituted into the Sweave file wherever similarly named variables preceded by the @ symbol occur.

The `cpue` switch signals the removal of pieces of Sweave code if `cpue=TRUE`.

The `Nsurvey` argument essentially copies one Sweave line into `Nsurvey` lines for a set of lines identified by unique Sweave snippets.

### Value

A customised Sweave file for `runNo` and `rwtNo` is created in a subdirectory `./strSpprun/MPD.runNo.rwtNo` from where it is run.

### Note

A fairly recent Seave file called `ymrrun-master.Snw` can be found in the library directory: `.../R/.../library/PBSawatea/snw`

### Author(s)

Rowan Haigh, Pacific Biological Station, Fisheries and Oceans Canada, Nanaimo BC

### See Also

[runMPD](#), [runSweaveMCMC](#), [runADMB](#)

---

runSweaveMCMC

*Run Customised Sweave Files for Awatea MCMC Runs*

---

### Description

Create and run customised Sweave files for Awatea MCMC runs.

### Usage

```
runSweaveMCMC( wd=getwd(), cpue=FALSE, estM=TRUE,
               strSpp="YMR", filename="input25-ymr.txt",
               runNo=25, rwtNo=0, running.awatea=0 )
```

### Arguments

<code>wd</code>	working directory in which Awatea input and master Sweave files occur.
<code>cpue</code>	logical: if <code>TRUE</code> the routine will leave calls to CPUE figures in the Sweave file, otherwise, they are removed.
<code>estM</code>	logical: if <code>TRUE</code> the routine will assume natural mortality $M$ was estimated in the run, and the Sweave code uses the parameters "M_1" and "M_2"; if <code>FALSE</code> , the function removes these parameters from the Sweave file.

<code>strSpp</code>	three-letter code that identifies the species.
<code>filename</code>	name of Awatea input file.
<code>runNo</code>	the run number that identifies a unique combination of input values.
<code>rwtNo</code>	the reweight number of the MPD run to build and collate figures.
<code>running.awatea</code>	numeric: if 0 load a previous <code>.rep</code> file, if 1 re-run Awatea

### Details

The values specified by the arguments (or derived variables) are directly substituted into the Sweave file wherever similarly named variables preceded by the @ symbol occur.

The `cpue` switch signals the removal of pieces of Sweave code if `cpue=TRUE`.

The `estM` switch signals the removal of "M\_1 " and "M\_2 " from the Sweave code if `estM=FALSE`.

### Value

A customised Sweave file for `runNo` and `rwtNo` is created in a subdirectory `./strSpprun/MCMC.runNo.rwtNo` from where it is run.

### Note

Sweave files can be found in the library directory:

```
.../R/.../library/PBSawatea/snw/ymrrun-masterMCMC.Snw
.../R/.../library/PBSawatea/snw/ymrrun-masterMCMC20yrProjs.Snw
```

### Author(s)

Rowan Haigh, Pacific Biological Station, Fisheries and Oceans Canada, Nanaimo BC

### See Also

[runMCMC](#), [runSweave](#), [runADMB](#)

---

srFun

*Stock Recruitment Function*


---

### Description

Take a vector of spawners in year  $t-1$  and calculate recruits in year  $t$ .

### Usage

```
srFun(spawners, h = h.mpd, R0 = R0.mpd, B0 = B0.mpd)
```

**Arguments**

spawners	a vector of spawners where either : each element corresponds to spawners in year $t-1$ or the vector calculates a single year but multiple MCMCs.
h	steepness parameter value.
R0	recruitment at $t = 0$ (virgin conditions).
B0	spawning biomass at virgin conditions.

**Details**

(AME wording) To input a vector of spawners in year  $t-1$  and calculate recruits in year  $t$ . Output for recruits is vector, each element corresponds to spawners the the year before, so will usually want to shift the output by 1 so that recruits in year  $t$  are based on spawners in year  $t-1$ .

Can also have each input as a vector (used when calculating a single year but multiple MCMCs, as in first year of projections is based on penultimate year of MCMC calculations).

**Value**

A vector of recruitments in year  $t$ .

**Note**

This function was originally a subfunction in `plt.mpdGraphs`.

**Author(s)**

Andrew Edwards, Pacific Biological Station, Fisheries and Oceans Canada, Nanaimo BC

**See Also**

[plt.mpdGraphs](#)

---

stdRes.CA

---

*Calculate Standardised Residuals for Robust Normal Likelihood*


---

**Description**

Calculate the standardised residuals for Awatea's implementation of the Fournier *robustified* normal likelihood for proportions-at-length.

Based on PJS's summary of the CASAL document and ACH's change to length.

**Usage**

```
stdRes.CA(obj, trunc=3, myLab="Age Residuals", prt=TRUE)
```

**Arguments**

<code>obj</code>	scape/list object of Awatea's results file ( <code>.res</code> ).
<code>trunc</code>	maximum standardised residual; values greater than this are set to <code>trunc</code> .
<code>myLab</code>	general label for the output.
<code>prt</code>	logical: if <code>TRUE</code> , print the results.

**Value**

List object of standardised residuals.

**Author(s)**

Andrew Edwards, Pacific Biological Station, Fisheries and Oceans Canada, Nanaimo BC

**See Also**

[stdRes.index](#), [importCol2](#), [reweight](#)

---

`stdRes.index`*Calculate Standardised Residuals for Abundance Indices*

---

**Description**

Calculate the standardised residuals for commercial and survey indices.

**Usage**

```
stdRes.index(obj, label=NULL, prt=TRUE)
```

**Arguments**

<code>obj</code>	data frame of observed and fitted index values from Awatea's results file ( <code>.res</code> ).
<code>label</code>	general label for the output.
<code>prt</code>	logical: if <code>TRUE</code> , print the results.

**Value**

Input data frame with additional column of standardised residuals.

**Author(s)**

Andrew Edwards, Pacific Biological Station, Fisheries and Oceans Canada, Nanaimo BC

**See Also**

[stdRes.CA](#), [importCol2](#), [reweight](#)

tabSAR

*Make Files of Reference Point Tables***Description**

Generate comma-delimited, two-dimensional output tables (.csv) from objects reporting probabilities and times for DFO reference points and COSEWIC criteria.

**Usage**

```
tabSAR(models=paste("input-ymr", pad0(c(29, 30), 2), pad0(1, 2), sep="."),
        pnam="refProbs3Gen90", tnam=c("Ttab0.5", "Ttab0.8", "Ttab0.95"),
        cats=seq(0, 2500, 500), digits=2)
```

**Arguments**

models	names of binary system files that store the decision tables.
pnam	name of list object containing matrices of reference probabilities.
tnam	names of matrices reporting times to reach reference points/criteria.
cats	catch strategies (subset) to report in output tables.
digits	number of digits to retain after the decimal.

**Details**

At the moment, the reference point objects are a list of matrices for the probabilities and individual matrices for the times to reach the references. The latter should also be a list of matrices to parallel the probability object.

**Value**

No value is returned in R; however, comma-delimited text files (.csv) are created in the user's working directory. The probability tables have the suffix `_prob.csv` while the time-to-target tables have the suffix `_targ.csv`.

**Note**

The subset of the master decision tables in the stock assessment are primarily used in the SAR (Science Advisory Report) for managers.

**Author(s)**

Rowan Haigh, Pacific Biological Station, Fisheries and Oceans Canada, Nanimo BC

**See Also**

`refPoints`, `findTarget`,  
`cat`, `round`, `pad0`, `show0`

# Index

## \*Topic **IO**

get.resFile, 8  
out.pmTables, 20  
tabSAR, 62

## \*Topic **arith**

cquantile, 6  
refPoints, 51

## \*Topic **array**

calc.projExpect, 3  
makeErrMat, 18

## \*Topic **character**

runMCMC, 55  
runMPD, 56  
runSweave, 57  
runSweaveMCMC, 58

## \*Topic **connection**

plt.mpdGraphs, 45

## \*Topic **device**

close.allWin, 4  
graphics, 10

## \*Topic **distribution**

plotChains, 29  
stdRes.CA, 60  
stdRes.index, 61

## \*Topic **file**

get.resFile, 8  
importCol2, 10  
importMCMC.ddiff, 12  
importRes, 14  
out.pmTables, 20  
tabSAR, 62

## \*Topic **hplot**

compB0, 5  
plotB2, 23  
plotBmcmcPOP, 25  
plotBox, 26  
plotChains, 29  
plotCPUE, 30  
plotDensPOP, 31

plotIndex2, 34  
plotIndexNotLattice, 35  
plotRmcmcPOP, 36  
plotSnail, 37  
plotTracePOP, 38  
plotVBcatch, 40  
plotVBnorm, 41  
plt.ageResidsPOP, 42  
plt.allTraces, 43  
plt.expRate, 44  
plt.idx, 45  
plt.mpdGraphs, 45  
plt.numR, 47  
plt.quantBio, 48  
plt.ssbVbCatch, 49  
plt.stdResids, 50

## \*Topic **interaction**

msyCalc, 19

## \*Topic **interface**

importCol2, 10  
importMCMC.ddiff, 12  
importRes, 14  
mainMenu, 17

## \*Topic **iteration**

runADMB, 54

## \*Topic **list**

calc.projExpect, 3  
load.allResFiles, 15  
MAfun2, 16

## \*Topic **logic**

allEqual, 2  
findTarget, 7

## \*Topic **manip**

calc.refVal, 4  
findTarget, 7  
runMCMC, 55  
runMPD, 56  
runSweave, 57  
runSweaveMCMC, 58

- \*Topic **methods**
  - readAD, 50
  - reweight, 52
- \*Topic **models**
  - readAD, 50
  - reweight, 52
  - runADMB, 54
- \*Topic **package**
  - PBSawatea, 22
- \*Topic **robust**
  - stdRes.CA, 60
- \*Topic **ts**
  - cquantile, 6
  - importProjRec, 13
  - plotDensPOP, 31
  - plotTracePOP, 38
  - srFun, 59
- \*Topic **univar**
  - MAfun2, 16
- \*Topic **utilities**
  - getYrIdx, 9
  - panLab, 20
  - panLegend, 21
- addLabel, 21
- addLegend, 21
- all, 3
- allEqual, 2
- boxplot, 6, 26
- boxplot.stats, 27, 28
- bxp, 6, 27, 28
- calc.projExpect, 3, 4
- calc.projExpect2
  - (calc.projExpect), 3
- calc.projProbs (calc.projExpect), 3
- calc.projProbs2
  - (calc.projExpect), 3
- calc.refProbs (calc.projExpect), 3
- calc.refVal, 3, 4, 52
- cat, 62
- clearAll, 3
- clipVector, 3
- close.allWin, 4
- closeWin, 4
- compB0, 5
- cquantile, 6
- cumuplot, 7
- densplot, 33
- expression, 27
- factor, 28
- findPat, 9
- findTarget, 4, 7, 19, 62
- get.resFile, 8, 18, 49
- getYrIdx, 9
- graphical parameters, 27
- graphics, 10
- importCol2, 9, 10, 15–18, 25, 40, 61
- importMCMC, 6, 12
- importMCMC.ddiff, 12
- importProj, 12, 13
- importProj.ddiff
  - (importMCMC.ddiff), 12
- importProjRec, 13
- importRes, 11, 14, 51, 55
- load.allResFiles, 15, 49
- loadMenu (mainMenu), 17
- MAfun2, 16
- mainMenu, 9, 17
- makeErrMat, 18
- mcmc, 13
- mcmcMenu (mainMenu), 17
- mpdMenu (mainMenu), 17
- msyCalc, 6, 19
- NA, 27
- out.pmTables, 20
- pad0, 9, 62
- panel.barchart, 24
- panel.densityplot, 33
- panel.loess, 40
- panel.superpose, 24
- panel.xYplot, 35
- panel.xyplot, 35
- panLab, 20
- panLegend, 21
- PBSawatea, 22
- PBSawatea-package (PBSawatea), 22



- plotAuto, 33, 40
- plotB, 24, 35
- plotB2, 23, 26, 35, 37, 41
- plotBmcmcPOP, 25, 37, 41, 42, 49
- plotBox, 6, 26
- plotBVBnorm, 38, 49
- plotBVBnorm (plotVBnorm), 41
- plotCA, 24, 35
- plotChains, 29, 40, 43, 44, 47
- plotCL, 24, 35
- plotCPUE, 30, 43, 46, 47
- plotCumu, 33, 40
- plotDens, 33, 40
- plotDensPOP, 30, 31, 40
- plotDensPOPpars (plotDensPOP), 31
- plotIndex, 24, 35
- plotIndex2, 24, 34
- plotIndexNotLattice, 31, 35, 43, 45–47
- plotLA, 24, 35
- plotmath, 6, 27
- plotN, 24, 35
- plotProp, 19
- plotQuant, 33, 40
- plotRmcmcPOP, 36, 47, 49
- plotSel, 24, 35
- plotSnail, 37, 47
- plotSplom, 33, 40
- plotTrace, 33
- plotTracePOP, 7, 30, 38, 40
- plotVBcatch, 26, 40, 42, 49
- plotVBnorm, 41
- plt.ageResidsPOP, 42
- plt.ageResidsqqPOP  
(plt.ageResidsPOP), 42
- plt.allTraces, 43, 43
- plt.cohortResids  
(plt.ageResidsPOP), 42
- plt.expRate, 43, 44, 49
- plt.idx, 36, 43, 44, 45
- plt.mcmcGraphs, 43, 47
- plt.mcmcGraphs (plt.mpdGraphs), 45
- plt.mpdGraphs, 45, 60
- plt.numR, 47
- plt.quantBio, 48
- plt.quantBioBB0 (plt.quantBio), 48
- plt.ssbVbCatch, 49, 50
- plt.stdResids, 45, 50
- plt.yearResidsPOP  
(plt.ageResidsPOP), 42
- qqnorm, 45
- read.table, 11, 12, 15
- readAD, 15, 50, 53, 55
- readLines, 11, 15
- refPoints, 51, 62
- refPointsB0 (refPoints), 51
- resetGraph, 10
- reweight, 15, 51, 52, 55, 61
- round, 62
- runADMB, 15, 17, 51, 53, 54, 58, 59
- runMCMC, 55, 57, 59
- runMPD, 56, 56, 58
- runSweave, 55–57, 57, 59
- runSweaveMCMC, 8, 56–58, 58
- scan, 11, 15
- scape-package, 11, 15, 24, 35
- scapeMCMC-package, 33, 40
- show0, 62
- srFun, 59
- stdRes.CA, 43, 53, 60, 61
- stdRes.index, 53, 61, 61
- stripchart, 28
- tabSAR, 62
- traceplot, 40
- utilMenu (mainMenu), 17
- weightBio, 19
- write.table, 20
- writeList, 20
- xyplot, 24, 33, 35, 40