



Getting started with CASAL2

C. Marsh

Citation: C. Marsh (2016). Getting started with CASAL2National Institute of Water & Atmospheric Research Ltd.. 15 p.

Contents

1	Introduction	1
2	How it works	2
3	Syntax of a CASAL2 file	3
4	Components of a model	4
5	Simple Example	5
6	Extended example	7
7	Analyses of output	8
8	References	9

1. Introduction

This document is a help guide for CASAL2 age-structured population modelling software package. This short document is aimed at users who are new to CASAL2. As the name suggests CASAL2 is a generalised tool for carrying out age-structured population dynamics models, including fisheries assessments and other population dynamics problems.

CASAL2 is very generalised, highly flexible, and therefore can be a bit daunting at first sight. It has a large number of run modes, settings, and user defined population dynamics choices that can be turned on and off, depending on the circumstances and the user requirements. While there is no requirement for a user to see or understand the underlying code base, it has been written so that it is well tested and great effort has been put into developing a code base that can be easily interpreted and understood by even novice programmers.

CASAL2 is open source, and is covered under the GNU GPL 2.0 licence. See the terms and conditions in the user manual (Rasmussen et al., 2016), or type `casal2 -v` into the command prompt. There is also supplementary information that may be useful to have access while reading this. This includes a more comprehensive and detailed user manual which goes into CASAL2 in more depth.

Also coming soon will be a document on how CASAL2 has inbuilt testing of code, and another validation document comparing CASAL2 predecessor with CASAL2. These can all be found in the CASAL2 file that is installed with the installer. By this stage we are assuming you have installed CASAL2 using the `casal2_setup.exe`. This will install CASAL2 into your path and create a folder in a location of your choice containing, source code, examples, a manual, an R library, this document, an executable and auxiliary libraries.

2. How it works

CASAL2 is run from a command prompt (windows) or shell (linux), CASAL2 is executed by typing `casal2 -parameter`, where `parameter` defines the run type of CASAL2. Once a CASAL2 has been executed with a certain parameter CASAL2 reads in text files. These text files define the model structure and the output wanted. For help on the parameters available and there descriptions type `casal2 --help`, this will print a help screen. There are multiple modes that CASAL2 can be run in, these are specified using the following command statement the program name followed by the mode parameter (e.g. CASAL2 -parameter). The modes and corresponding parameters include deterministic run `-r`, parameter estimation `-e`, parameter profiling `-p`, mcmc runs `-m`, and projections `-f`. There are two ways of printing output, the default is to print all output to screen, the second is to print output to a file. The second is usually the preferred if you intend on post processing output i.e. create plots. The following example shows how to read in text file that out model is configured in (`My_model.txt`) and run an estimation on some parameters in that model, then print the output to a file named `output.txt`.

```
CASAL2 -e -c My_model.txt > output.txt
```

CASAL2 calls the program, `-e` tells the program it is going to do an estimation. `-c` is the parameter that gives the name of the text file with the configured model is, and `>` is the command to specify the file name where the output is printed.

3. Syntax of a CASAL2 file

A general structure of CASAL2 files are that they are split into blocks of subcommands. A block always starts with @ symbol. Blocks describe different aspects of the model, fundamental blocks to have in the model are @model, @initialisation_phase, @categories, @time_step, and @process. Within each block there will be subcommands some will be optional and important subcommands will be mandatory. An example of subcommand is shown for the @model block,

```
@model
type age ## is the model age or length based?
min_age 1 ## minimum age in model
max_age 17 ## maximum age in model
age_plus true ## is the last age group a plus group?
start_year 1972 ## the first year of the model
final_year 2013 ## the first year of the model
initialisation_phases phase1
## The label for the block @intialisation_phase
time_steps step1 step2
## Labels for the block @time_step
```

The subcommands are all the options that follow @model, then there is a space which is where the value for the subcommand goes, i.e. min_age specifies the minimum age in the model and we have set that equal to one but could be any integer. This brings up a useful concept to understand. Different subcommands can take different types of parameters, they can be of type int, double, string and vector. For information about which parameter type a subcommand takes, you should read the syntax section of the manual, there is a field labelled type. If you use the wrong type for a subcommand, for example min_age 1.5, you will get an error. A line beginning with # is a comment and that line is ignored by CASAL2. To comment out multi-lines the user can use the curly braces {}, everything between these braces will be ignored by CASAL2. It is a useful tool for annotating models.

4. Components of a model

Components of a model that are important to know before setting up a CASAL2 model are, How many categories are in the partition, what processes occur to which categories in which order, where observations fit in to the model, and what the assumed state of the partition is before the model years run. CASAL2 runs in yearly cycles each year is split up by time steps, So processes such as fishing and spawning seasons will have an effect on how to specify time steps and so will observations such as annual surveys. The next section runs through a very simple example,

5. Simple Example

In the following example describe a situation then go on to configure a CASAL2 file to run. In this example we have a single area, single stock that has one fishery associated with it. We assume that the partition is made up of a single category (no sex or maturity in the partition). Processes and observations that occur in a typical year in the following order.

1. Recruitment
2. Fishing mortality with natural mortality
3. A survey takes place out of the fishing season and in the spawning season
4. More natural mortality
5. At the end of the year all the fish are aged.

The following model would have the following structure.

```
@model
type age ## is the model age or length based?
min_age 1 ## minimum age in model
max_age 17 ## maximum age in model
age_plus true ## is the last age group a plus group?
start_year 1972 ## the first year of the model
final_year 2013 ## the first year of the model
initialisation_phases phase1
## The label for the block @initialisation_phase
time_steps step1 step2
## Labels for the block @time_step

@categories
format Stock ## format of the category labels
names CHAT4 ## category labels
age_lengths CHAT4_AL ## Lables of age-length relationship for each category
```

The `@categories` command defines the label, number and age-length relationship of categories that make up the partition. A category is a group of individuals that have the same attributes, some examples of such attributes are, life history and growth paths. Characters in a populations that cause differing attributes can be, sex, maturity, multiple area, multiple stock's and tagging information. An example of the `@categories` block for a simple two area model with male and female in the partition.

```
@time_step step1 ## The label from the @model subcommand
processes Recruitment Mortality ## Labels for @process block

@time_step step2
processes Mortality Ageing
```

The `@time_step` command describes which processes are implemented and in what order. We will continue on from the `@model` block example, where we defined two time steps in the annual cycle (`time_steps step1 step2`). In each year we have two time steps, within each time step we have processes each process must be derined in `@process` block the following processes are described.

5 Simple Example

```
@process Recruitment ## label of process form @time_step
type recruitment_constant ## keyword relates to a specific process
## The following are specific subcommands for this type of process
r0 4E7 ## Number of average recruits if no fishing were to occur
age 1 ## age of recruits when entering the partition
categories CHAT4 ## label of categories that recruits join
proportions 1 ## proportion of recruits to each category

@process Mortality
type mortality_instantaneous
categories CHAT4 ## category labels
M 0.19 ## natural mortality rate
selectivities One ## label to a @selectivity block
## this selectivity allows for age varying mortality
time_step_ratio 0.4 0.6 ## If this process is in multiple @time_step blocks
## then this is the proportion of M that occurs in each time step.
table catches
year Fishing
1975 80000
1976 152000
1977 74000
1978 28000
1979 103000
1980 481000
1981 914000
end_table

table fisheries
fishery category selectivity u_max time_step penalty
Fishing CHAT4 FSel 0.7 step1 Catchmustbetaken
end_table

@process Ageing
type ageing
categories CHAT4_AL
```

The above defines all the processes that occur to the partition. In the process Mortality we associate a selectivity to natural mortality and in the fisheries table FSel, this would be defined as follows.

```
@selectivity One
type constant
c 1

@selectivity FSel
type double_normal
mu 3.82578
sigma_l 1.63038
sigma_r 17
```

If a age-length relationship is specified in the @categories block then the @age_length block needs to be defined, this block is used to convert age to length which is then used to convert length to weight in an age based model, it is specified as follows,

```
@age_length CHAT4_AL
type von_bertalanffy
length_weight CHAT4_LW ## label for @length_weight block
```

```

k 0.164
t0 -2.16
linf 100.8

@length_weight CHAT4_LW ## label from @age_lenght block
type basic
units tonnes
a 4.79e-09
b 2.89

```

The last important block to complete the population text file, is the @initialisation_phase. This block of commands specifies how you initialise your partition. This describes the state of the partition before start_year of the model, usually this is an equilibrium state. The subcommands available for this block are as follows,

```

@initialisation_phase phase1
type iterative ## Type of initialisation method see manual for more
years 100 ## How many years to run for

```

In the above example we have an iterative initialisation type. This will default to iterating your annual cycle for 100 years, which may or may not cause your partition to hit an equilibrium state. **N.B.** when using this initialisation method you as the user must check if the partition has reached an acceptable equilibrium state.

The next section we are defining is the observation section. We have a survey that occurs in the second time step, which is of relative abundance, this would be defined as follows.

```

@observation Survey ## label of observation
type biomass ## tyoe of observation
time_step step2 ## which time step the observation occurs
time_step_proportion 0.5 ## the observation occurs half way through the time step
categories CHAT4
selectivities One
catchability q ## The label for @catchability block
years 1992 1993 1994 1995
obs 191000 613000 597000 411000
error_value 0.41 0.52 0.91 0.61
likelihood lognormal ## likelihood to use for the objective function

@catchability q ## label from @observation
q 0.001 ## The value

```

To run the simple example which is located in CASAL2/Examples/Simple. [shift] + right click – > open command window in the above directory. Type in the command window `casal2 -r` and output should print to screen.

6. Extended example

Add a spawning stock biomass catch at age data multiple categories

7. Analyses of output

CASAL2 has an **R** library **CASAL2** which imports CASAL2 output files into **R** as a list. This library can be found in the directory where you installed the program. There is also another library that helps pull out compress useful information such SSB's and Objective scores for datasets.

```
library(CASAL2)
output = extract(file = "Output_file.txt", path = "Directory_of_file")
```

8. References

S. Rasmussen, I. Doonan, A. Dunn, C. Marsh, K. Large, and S. Mormede. Casal2 user manual. Technical Report 139, National Institute of Water and Atmospheric Research Ltd (NIWA), 2016.