BAYESIAN LIFE CYCLE MODEL FOR ATLANTIC SALMON STOCK ASSESSMENT

USER GUIDE for the R/Nimble codes

ICES WKBSalmon 2023

15 October 2023

Authors

Rémi LEMAIRE-PATIN [1,2,3,#] and Etienne RIVOT [1,2,*]

[1] DECOD Ecosystem Dynamics and Sustainability, Institut Agro, INRAE, Ifremer, Rennes, France

[2] MIAME, Management of Diadromous Fish in their Environment, OFB, INRAE, L'Institut Agro, E2S UPPA, Rennes, France

[3] ISPA, Bordeaux Sciences Agro, INRAE, F-33140, Villenave d'Ornon, France

*Email: etienne.rivot@institut-agro.fr

[#] Email: remi.lemaire-patin@inrae.fr

Recommended format for citation:

Lemaire-Patin, R., Rivot, E. 2023. Bayesian life cycle model for Atlantic salmon stock assessment. User guide for the R/ Nimble codes. ICES WKBSalmon 2023, Working Paper, October 2023, 10 pp.
https://doi.org/10.17895/ices.pub.24752079

Acknowledgements

This work was made possible by the investment of the numerous people from various institutions who collect and compile the data used by the ICES Working Group on North Atlantic Salmon.

The project has received funding from the Office Français de la Biodiversité under grant agreement INRAE-OFB SalmoGlob-ToolBox, and from the European Regional Development Fund through the Interreg Channel VA Programme, project SAMARCH Salmonid Management Round the Channel.

This working paper uses material from

Rivot, E., Lemaire-Patin, R., Olmos, M., Chaput, G., Hernvann, P-Y. 2023. A hierarchical Bayesian life cycle model for Atlantic salmon stock assessment and provision of catch advice at the North Atlantic basin scale. ICES WKBSalmon 2023, Working Paper XXX/XXX, October 2023, 143 pp.

Table of contents

Abstract

This WP provides guidelines to use the suite of programs written in R/Nimble for the hindcasting-forecasting streamline needed for the stock assessment and provision of catch advices.

The core of the Bayesian Life Cycle Model is written in BUGS language using the Nimble package for Bayesian estimation using MCMC algorithms. The Nimble model is run within R.n

The programs use data files formatted to be read by the Nimble code. The data are directly available from the shiny web application at https://sirs.agrocampus-ouest.fr/discardless_app/WGNAS-ToolBox/

In practice the same life cycle model is used for fitting the historical time series (hindcasting) and for forecasting. In practice, one unique life cycle model code written in Nimble is used for both the hindcasting and the forecasting phases. This ensures model consistency between the two phases and limits errors as no re-coding is required between the two phases. In addition, the posterior MCMC samples from the hindcasting phase can be easily re-used to propagate parameters uncertainty in the forecasts.

1 Structure of the folders for the R/nimble program and outputs

1.1 Base folders containing R/Nimble codes

The R/Nimble codes for the programs are structured in folders as follows:

R/Nimble codes	What's in
run.R	The master script to be run for the whole streamline: • load required packages, functions and data, MCMC configuration • generate initial values for MCMC chains • run MCMC for the hindcasting phase (could be long; ~12h in the recommended MCMC configuration using 10 chains in run in parallel) • run Mont Carlo simulation for the forecasting phase • format all results to produce figures and tables.
model	Contains the Nimble code for the LCM. Some sections that are activated/disactivated depending on the mode "hindcasting" or "forecasting".
input_data	All data and initial values used to run the hindcasting. Those data must be pre-formatted to be read by Nimble. Data read by Nimble must be structured in two different files: • *Constant_nimble* must contain all the data that are used directly as fixed values (constant) in the model (e.g. the biological characteristics) • *Data_Nimble* must contain all the data considered as observations with observation errors The *Constant_nimble* and *Data_nimble* files are automatically generated through the shiny web app. They can also be alternatively updated by hand, although this is more tedious and error-prone. This folder also contains the initial values for MCMC chains generated by the first section of the master script *run.R*.
Rscript	All R scripts called by the master script *run.R* using *source()*
Functions	All R functions called in the code.

1.2 Results automatically saved

All outputs and results (MCMC samples, tables and figures) are automatically created and saved in folders that are also automatically created.

These folders can be deleted before a new run.

Folders created during the run	What's in
output_hindcast	Contains hindcast results in two forms: MCMC samples as mcmc.list object or as tables saved as .csv files
output_forecast	Contains intermediate files of forecast results (mostly tables saved as .RDS) build from the forecast results and used to produce figures
temp_figures	Contains intermediate files (mostly tables saved as .RDS) build from the hindcast and forecasting phase and used to produce figures
temp_docx	Contains intermediate files (mostly .RDS) used to produce docx
temp_shiny	Contains intermediate files used to produce figures formatted for the shiny web app SalmoGlob Toolbox https://sirs.agrocampus-ouest.fr/discardless_app/WGNAS-ToolBox/
Figures	All individual figures saved as *.png* files. Figures are produced for: • Input data • Hindcast results • Convergence diagnostic • Forecast results
Figure_docx	Figures automatically summarised in .docx, with Hindcast and Forecast summaries. Also contains table with forecasted compliance to conservation limits (CL)
log	Saves logs message obtained during hindcast, forecast or analysis of results. These messages could be of help to identify bugs.

2 Sections of the master script "*run.R*"

2.1 Note - on the use of "sink" and "cli"

The R code has option to communicate to the user (in the R console and in the log files) and to limit the amount of information that appear in the R console during the run. R-package *cli* is used for friendly communication.

cli. When reading the code, all lines related to *cli::cli_xxx* may be ignored as they do not concern the model itself but only participate in reporting achievement of the different phases/sections.

sink. To limit the output printed to the R console, output redirection was organized with *sink()*. Lines involving *sink()* may also be safely ignored. Note that configuration steps involve adding a function on errors that should deactivate the *sink()*, but if you execute code and do not see any outputs the sink may be already active. In such situation restarting R is recommended.

2.2 Master script – "***run.R***"

The whole suite of programs for hindcasting / forecasting is run through the master script ***run.R***:

- Loading data, functions
- Set MCMC configurations
- Generate initial values of MCMC chains
- Configure, build and run the model for the hindcasting phase
- Configure, build and run the model for the forecasting phase (scenarios for catch advices)
- Produce outputs (figures) for checking model convergence and fit to the data
- Produce outputs (figures, tables) for the hindcasting and forecasting phase

Below we detail the different sections.

2.3 General settings - "*0_setup.R*"

This set of script must be run prior to any run of the model. It loads required libraries, functions, data and variables needed to run the model. It also handles the general configuration of the MCMC sampling.

2.3.1 R and Nimble configuration

The script controls the R configuration (version of R, Nimble and different packages) to be sure the appropriate software configuration is appropriate to run the model. It loads all required packages:

- If working on a Windows system, please make sure that *Rtools* is available and properly setup (see comments in *Rscript/0_setup.R*)
- R version > 4.0 is required. Script will cause an error if R version is older. If you have a very good reason to not update R, you could ignore the following line in the script

  Rscript/0_setup/0_general_checks.R :
  stop("Your ",R.version$version.string, " is too old and should be > 4.0.0")

- Nimble version > 0.12 is required. Nimble is a package in development and functionality change regularly. Current code was developed with Nimble version 0.12.2 and supports for other version (especially older one) is not guaranteed. More recent version may also cause problem in the future if the code is not regularly updated to changes in Nimble package. In case your nimble package is older than 0.12, the script will propose an automatic updating.

2.3.2 MCMC configuration

The MCMC configuration must be specified in the Script *"0_setup.R"*. The recommended default configuration is:

- 10 chains
- 275 000 iterations
- thinning of 250
- 1100 iterations kept by chains after thinning
- a burnin of 100 iterations after thinning

With this configuration, inferences will be drawn using 1000 iterations for each chain, that is a total of 10000 iterations available after collating the 10 chains.

2.3.3 Parallelization

Parallelization is strongly recommended to reduce computational time. User can configure whether hindcast is run in parallel and whether forecast is run in parallel as well. For hindcast number of cores required is the number of chains (default 10). For forecasts, the number of scenarios will be shared among cores. The ideal situation is to have as many cores as scenarios.

With such configuration, standard runtime on recent computer is ~12-18h for hindcast (depends on the hardware configuration, processor and RAM) and ~1h for a fully paralleled forecast (as many cores as scenarios). Non-parallel run for hindcast may take between 120 and 180h.

2.3.4 Variables to monitor

An important script called in this section is "*Rscript/0_setup/0_get_nimble_monitors.R"* that configures nimble monitors, i.e. all variables within the nimble model which values will be recorded for all iterations. Adding new variables in the model require an update of this script. If not, MCMC for the new variables won't be stored.

2.4 Generate initial values - "*1_generate_inits.R*"

This script generates well-chosen initial values for the MCMC algorithm which are a priori pretty close to the high posterior density regions. This is an important phase as initializing the model to random values can seriously hampers the mixing of MCMC chains and dramatically increase the computational time to reach convergence.

This generation is a two-step procedure that first generate best-guess values for key model parameters (harvest rate, post-smolt survival rate and prob. maturing as 1SW) and then sample other variables in their distribution given the fixed values for the key parameter:

- **1.** Generate best guess values for key variables, especially post-smolt survival (theta3) and prob. of maturing as 1SW (theta4) and all harvest rates. This makes use of function "*generate_fixed_values.R*" (in the functions directory);
- **2.** Those values are then used in a 2nd step to simulate all other model variables. This makes use of the function "*simul_inits.R*" that uses the nimble code of the model to simulate values from the model, with several key nodes fixed to some values (from the step 1). In that way, the simulations provide values of states variables (typically the abundance at different life stages) that are consistent with the transition rates and close to the high posterior density region.

Important note. Adding new variables or changing model structure (changing stochastic nodes) may require adjusting the two functions to accommodate for those changes.

Initial values generated through this procedure are directly saved as *.rds* file in the "*input_file*" folder.

2.5 Compiling the model and running MCMC simulations - "*2_build_and_run_hindcast.R*"

This section handles building and running the hindcast model. There are three main steps in this section:

- **1.** Building and compiling the model and the MCMC sampler (those steps are separated in Nimble, which provides some flexibility to customize the MCMC sampler);
- **2.** Running the MCMC simulation to fit the model. This produces an object of the class *mcmc.list* where all MCMC samples are stored;

- **3.** Reformatting the MCMC samples into more readily usable *data.frame* stored as .csv

This sections makes uses of the script "*2_compile_model.R*" which loads the Nimble code and compiles the model, build and compile the MCMC sampler, and run MCMC for the hindcast.

Setup for MCMC chains (number of iteration, thin, etc.) are defined in section "*0_setup.R*".

The Nimble model can be used in hindcast or forecasted mode. The option hindcast = TRUE is defined in that script "*2_build_and_run_hindcast.R*".

As nimble models are very verbose, and R console can very quickly be flooded with messages, a redirection of output to log files have been setup.

2.5.1 Parallel MCMC run

Option to run hindcast on multiple cores in parallel "*parallel_run = TRUE/FALSE*" is defined in "*0_setup.R*". If "*parallel_run = TRUE*", the script in "2_compile_model.R" will be called for each cluster (model compilation should be repeated in each cluster). Default option is 1 chain per cluster (so e.g., 10 chains will use 10 cluster). Non-parallel version is also available and will be activated if option "parallel_run = FALSE" is specified in "*0_setup.R*". Non parallel should be the preferred option to explore and test the script but is not recommended for long simulations run needed to fit the model.

2.5.2 Formatting MCMC samples

At the end of the run, MCMC samples are formatted and saved in the folder *output_hindcast*

- *mcmc_results.rds*: contains all MCMC samples (collating all chains) formatted in a table;
- *time_mcmc_results.rds*: contains simulation time;
- all posterior of all variables individually are formatted as tables and saved as .CSV files in the directory "*output_hindcast/posteriors*".

This formatting phase makes use of important functions "*get_df_varname.R*" and "*get_mcmc.R*".

2.5.3 Note on log files and errors when using parallelization

Windows system provides limited support for redirection especially when using the R package parallel so output redirection is different between Unix and Windows system:

- **Windows.** When running the cluster, most output cannot easily be directed to the R console so a log file per core is created to store the results and the progression can only be seen in each core logfile
- **Unix**. When running the cluster, output is separated so that compilation and building of models on each core are using separate logfile, with steps written directly in the R console, although

error or warning may only be seen in the logfiles. During the model run, outputs and progression can be directly seen from the console.

2.6 Format hindcast results - *"3_format_hindcast_results.R"*

In this section MCMC results saved in folder *output_hindcast* are re-loaded and then formatted into several R objects (as .RDS files) that are shaped to be easily read by ggplot to produce graphics (for instance, one calculate the median and different quantiles for all variables, that will be used to plot summary of posterior distributions).

These objects are stored as .RDS files in the folder *temp_figures*

- temp_figures/convergence ;
- temp_figures/hindcast.

These objects will be read in section 4 to produce graphics using *ggplot.*

2.7 Plot hindcast results - *"4_plot_hindcast_results.R"*

This section makes several figures to illustrate results of the hindcasting phase. Figures are based on .RDS files generated in the previous section "*3_Format_hindcast_results*" and stored in the folder *temp_figures.* Figures (as *.png* files) are stored in the folder *Figures:*

- Figures concerning data only are stored in *Figures/Input Data/;*
- Figures concerning model convergence are stored in *Figures/Model Convergence/;*
- Figures concerning hindcast are stored in *Figures/Hindcast/.*

The script also calls *RMarkDown* files that produces .docx files that contains a selection of figures.

2.8 Run forecasting under different catch options - *"5_run_forecast.R"*

In this section, the life cycle model is run in a forecasting mode under different catches options at Faroes and West Greenland and by integrating the posterior uncertainty around the parameters from the hindcasting phase.

Results of the forecasting phase are stored in the folder *output_forecast* directly as .CSV files for each scenario separately.

2.8.1 Catch options at WG and Faroes

Catch options are defined in the script "*define_forecast_scenario.R*" and in the function "*generate_forecast_const_nimble.R*"

- The script "*define_forecast_scenario.R*" setup the main configuration of the forecast scenario (catches at WG and Faroes, number of forecasted years)
- The function "*generate_forecast_const_nimble.R*" defines all other variables needed to specify the conditions during forecasting: homewater catches, all other marine catches, biological characteristics

Characteristics of each scenario are built during the simulation in a dynamic object "*Const_forecast*"

2.8.2 Forecasting horizon

Because of the demographics and the way the equations are written in the Nimble code, all variables in the model are not defined for the same time series.

The minimum number of years for the forecast horizon for all variables is defined by the variable "*year_after*" in script "*define_forecast_scenario.R*". All variables will be generated at least for "*year_after*" years after the last year of the hindcasting phase. But almost all variables will be generated for a longer time series (e.g., could be *year_after + 5* for some variables, which is not a problem).

The variable "*assess_CL*" in the script "*7_plot_forecast_results.R*" (see below) controls the forecast horizon after the last year off data for which compliance to CL is assessed. It also controls the forecast period that is plotted in all graphs.

"*year_after* " and "*assess_CL*" are typically set to 3 years as it corresponds to the time frame of catch advices required by NASCO.

2.8.1 Using the Nimble code in forecasting mode

This script makes of the Nimble code in forecasting mode. The Nimble model is called (compiled and run) by the function "*simul_forecast.R*". The option "*hindcast = FALSE*" which allows using the model in a forecasting mode is specified in the function "*simul_forecast*".

2.8.2 Integrating posterior uncertainty around parameters

Posterior uncertainty around parameters estimates are integrated in the simulations. This is done by simulating trajectories by sampling *n.keep* MCMC draws for each chain (x number of chains). This

setting of simulations is defined in the script "*define_forecast_scenario.R*" that generates the *dfsampling* object that contains this info.

2.8.3 Parallelization

Option to run forecast in parallel is defined in the "*0_setup.R*" script (option "*parallel_forecast = TRUE/FALSE*"). The number of nodes in the cluster to be used for forecasting can be controlled with the variable "*ncluster_forecast*" in the script "*0_setup.R*". The best default option is to set the number of nodes to the number of scenarios so as each scenario can be run on one node.

2.9 Format Forecast results - "*6_format_forecast_results.R*"

In this section MCMC results saved in the folder *output_forecast* as .CSV files are re-loaded and formatted into several R (as .RDS files) object that are prepared to be read by ggplot to produce graphics. Those objects are stored as .RDS files in folder *temp_figures/forecast (t*hey will be read in Section "*Plot forecast results"* below to produce graphics using ggplot.

The script makes use of the management objectives (defined in eggs for all SU) loaded from "*input_data/Data_CL.rds*".

The script essentially calculates the variables to get

- The compliance to conservation limits at different scales (SU, country, complex). This is calculated as the proportions of the Monte Carlo draws of the eggs deposition that reach the Management objectives (country, stock complex).
- The proportions of eggs spawned by 2SW fish, aggregated at different scales (country, stock complex)

2.10Plot forecast results - "*7_plot_forecast_results.R*"

This script generates figures and tables of the forecasting results. It works directly from pre-formatted .RDS files generated in the previous section "*6_Format_forecast_results.R*". It also makes use of the management objectives loaded from "*input_data/Data_CL.rds*".

Figures concerning forecasts are saved in the folder *Figures/forecast*.

The script also produces summary as *.docx* file through *RMarkDown* files saved in the *Figure_docx* folder. It also produces tables for the probability to reach management objectives also saved as .xlsx files in the *Figure_docx* folder.

Note the number of years after the last assessment year (last year of data) for which the compliance to management objectives is calculated is controlled by the variable "*assess_CL*" in

"*7_plot_forecast_results*". This variable also controls the forecast period that will be plotted in all graphs.