# Editor’s issues:

1) Statements regarding SS3 as the standard for stock assessment, re Reviewer 1. In reality SS3 is a controversial assessment tool and reference to it as a standard while others will not use it, does not convey the present reality. Explicit and implicit reference to SS3 as the standard needs to be removed and acknowledged that it is an approach/tool in a suite.

2) Following from comments of Reviewer 1 and the idea of a standard, the idea of a base OM approach should be explored further. This would have the added effect of broadening the work somewhat to consider this issue that seems to be arising more and more frequently in management procedure type approaches, i.e. an OM that is specific to the situation and fitted as such or a more general approach where many of the assumptions may not reflect the specific situation.

3) Reviewer 2 would like to see the conclusions rewritten. The reviewer suggests ways to write this which I think could slightly broaden the paper.

4) The software needs to be made easily available. Reviewer 1 suggests getting it onto or linked to the NFT website (and possibly on CRAN) which would seem to be the way to reach the relevant people.

5) Please respond to the comment from Reviewer 1 concerning zero observation error.

**Line-specific comments are included below.**

# Abstract

Simulation testing is an important approach to evaluating fishery stock assessment methods. In the last decade, the fisheries stock assessment modeling framework Stock Synthesis (SS3) has become widely-used around the world. However, there lacks a generalized and scriptable framework for SS3 simulation testing. Here, we introduce ss3sim, an R package that facilitates large-scale, rapid, and reproducible end-to-end simulation testing with SS3. ss3sim requires an existing SS3 model configuration along with plain-text control files describing alternative population dynamics, fishery properties, sampling scenarios, and assessment approaches. ss3sim then generates an underlying truth, samples from that truth, modifies and runs an estimation model, and synthesizes the results. The simulations can be run in parallel, speeding computation, and the source code is free to be modified under an open-source MIT license. ss3sim is designed to explore structural differences between the underlying truth and assumptions of an estimation model, or between multiple estimation model configurations. For example, ss3sim can be used to answer questions about model misspecification, retrospective patterns, and the relative importance of different types of fisheries data. We demonstrate the software with a simple example, discuss how ss3sim complements other simulation software, and outline specific research questions that ss3sim could address.

# Introduction

Fisheries stock assessment models are an invaluable tool to providing scientific advice and to evaluating the impact of alternative management actions on fishery resources [@gulland1983; @hilborn1992]. Although a variety of stock assessment approaches are available, it is often not straightforward to select among competing approaches that may lead to different modeling outcomes and associated scientific advice to management.

Simulation testing is a critical component to testing fishery stock assessment methods, particularly given the potential for model misspecification [@hilborn1987; @hilborn1992; @rosenberg1994; @peterman2004; @deroba2013a]. With simulation testing we can evaluate the precision and bias of alternative assessment methods in a controlled environment where we know the true dynamics of fisheries resources under exploitation. Recent simulation studies have been key to improving strategies for dealing with, for example, time-varying natural mortality () [@lee2011; @jiao2012; @deroba2013; @johnson2013], uncertainty in steepness of the stock-recruit relationship [@lee2012], and environmental variability [@schirripa2009], as well as determining the utility and influence on assessment outcomes of various fishery dependent and fishery independent data sources [@magnusson2007; @wetzel2011a; @ono2013; @yin2004].

Stock Synthesis (SS3, the third version of the software) is a widely-used fisheries stock assessment modeling framework [@methot2013]. SS3 implements statistical age-structured population modeling, using a wide range of minimally-processed data [@maunder2013; @methot2013]. By using this generalized framework, individuals conducting fisheries stock assessments and peer reviewers can focus on the underlying science, instead of the model code [@methot2013]. Owing in part to these advantages, SS3 is one of the world's most commonly-used stock assessment modelling frameworks, particularly in the United States and Australia, where it has been used in 35 and 12 stock assessments as of 2012, respectively [@methot2013]. Stock Synthesis is also commonly used as a framework for stock assessment simulation testing [@helu2000; @yin2004; @schirripa2009; @lee2011; @jiao2012; @lee2012; @crone2013a; @hurtadoferro2013].

SS3 is increasingly becoming a standard for fisheries stock assessment, but there lacks a generalized framework for simulation testing with SS3. As a result, most stock assessment simulation-testing work to date has used custom frameworks tailored to the particular needs of each study [@helu2000; @yin2004; @magnusson2007; @wetzel2011a; @jiao2012; @wilberg2006; @deroba2013a; @deroba2013; @crone2013a; @hurtadoferro2013]. The programming language R [@rcoreteam2013] is an ideal language to write such a generalized framework in because (1) R has become the standard for statistical computing and visualization and (2) the R package r4ss [@r4ss2013] facilitates reading, processing, and plotting of SS3 model output.

Here we introduce ss3sim, an R package that facilitates large-scale, rapid, and reproducible end-to-end simulation testing with the widely-used SS3 framework. We begin by outlining the general structure of ss3sim and describing its functions, and then demonstrate the software with a simple example. We conclude by discussing how ss3sim complements other simulation testing software and by outlining some research questions that our freely accessible and general SS3 simulation-testing framework could address.

# The ss3sim framework

## Design goals of ss3sim

We designed ss3sim evaluations to be reproducible, flexible, and rapid. *Reproducible*: ss3sim simulations are produced using R code, plain-text control files, and SS3 model configurations. ss3sim also allows for random seeds to be set when generating observation and process error. In combination, these features make simulations repeatable across computers and operating systems (Windows, OS X, and Linux). *Flexible*: ss3sim inherits the flexibility of SS3 and can therefore implement many available stock assessment configurations by either modifying existing SS3 model configurations or by modifying built-in generic life-history model configurations (Text S1). Furthermore, ss3sim summarizes the simulation output into plain-text comma-separated-value (.csv) files allowing the output to be processed in R or other statistical software. Finally, the ss3sim code is written under an open-source MIT license and can be freely modified. *Rapid*: ss3sim relies on SS3, which uses AD Model Builder as a back-end optimization platform @fournier2012] --- a rapid and robust non-linear optimization software [@bolker2013]. ss3sim also facilitates the deployment of simulations across multiple computers or computer cores (i.e. parallelization), thereby reducing runtime. By using the vetted SS3 framework [@methot2013] with the tested ss3sim package [@johnson2013; @ono2013], the time to develop a large-scale simulation can be reduced substantially, shifting focus to the research questions themselves.

## The general structure of an ss3sim simulation

ss3sim consists of both low-level functions that modify SS3 configuration files and high-level functions that combine these low-level functions into a complete simulation experiment (Figure 1, Table 1). In this paper we will focus on the structure and use of the high-level function run\_ss3sim; however, the low-level functions can be used on their own as part of a customized simulation (see Text S1).

An ss3sim simulation requires three types of input: (1) a base SS3 model configuration describing the underlying true population dynamics, or operating model (OM); (2) a base SS3 model configuration to assess that truth based on data generated using the OM, also known as the estimation model or method (EM); and (3) a set of plain-text files (case files) describing alternative model configurations and deviations from these base models (e.g. different fishing mortality or trajectories). We refer to each unique combination of OM, EM, and case files as a scenario. Scenarios are usually run for multiple iterations, possibly adding unique process and observation error to the OM each time. An ss3sim simulation therefore refers to the combination of all scenarios and iterations.

The run\_ss3sim function works by modifying SS3 configuration files as specified in the case-file arguments (change functions), running the OM, sampling from the time-series of true population dynamics to generate a dataset (sample functions), running the EM to get maximum-likelihood estimates of parameters and derived quantities, and synthesizing the output for easy data manipulation and visualization (get functions) (Figure 1).

# An example simulation with ss3sim

To demonstrate ss3sim, we will work through a simple example in which we examine the effect of (1) high vs. low precision of a fishery independent index of abundance and (2) fixing natural mortality ($M$) at an assumed value vs. estimating $M$. All files to run this example are included in the package data, and a more detailed description is available in the accompanying vignette (Text S1). ss3sim requires R version 3.0.0 or greater and SS3 (see Text S1 for more detailed instructions). In R, the development version of ss3sim can be installed with:

install.packages(devtools)  
devtools::install\_github("ss3sim", username = "seananderson",  
 dependencies = TRUE)

You can read the documentation and open the vignette (Text S1) with:

?ss3sim  
help(package = "ss3sim")  
vignette("ss3sim-vignette")

## Setting up the SS3 model configurations

ss3sim comes with built-in SS3 model configurations that represent three general life histories: cod-like (slow-growing and long-lived), flatfish-like (fast-growing and long-lived), and sardine-like (fast-growing and short-lived). These model configurations are based on North Sea cod (*Gadus morhua*; R. Methot, pers. comm.), yellowtail flounder (*Limanda ferruginea*; R. Methot, pers. comm.), and Pacific sardine (*Sardinops sagax caeruleus*) [@hill2012] (Text S1). We recommend modifying these built-in model configurations to match a desired scenario, although it is possible to modify an existing SS3 model configurations to work with ss3sim (Text S1). We will base our example around the built-in cod-like model setup.

## Setting up the case files

The high-level function run\_ss3sim can run all simulation steps based on a specified scenario ID and a set of semicolon-delimited plain-text files that describe alternative cases (Figure 1). These files contain argument values that will be passed to the low-level ss3sim R functions (e.g. change\_index, a function that controls how the fishery and survey indices are sampled; Table 1).

To use run\_ss3sim all case files must be named according to the type of case (e.g. E for estimation or F for fishing mortality), a numeric value representing the case number, and an alphanumeric identifier representing the species or stock (e.g. cod; Table 1, Text S1). We combine these case IDs with hyphens to create scenario IDs. For example, one of our scenarios will have the scenario ID D1-E0-F0-M0-R0-cod. This scenario ID tells run\_ss3sim to read the case files corresponding to the first data (D) case (i.e. index1-cod.txt, lcomp1-cod.txt, agecomp1-cod.txt), the zero case for estimation (E; i.e. E0-cod.txt), and so on.

To investigate the effect of different levels of precision of a fishery independent index of abundance, we will manipulate the argument sds\_obs that gets passed to the function change\_index. In data case 0, we will specify the standard deviation of the index of abundance at 0.1 and in case 1 we will increase the standard deviation to 0.4. We can do this by including the line: sds\_obs; list(0.1) in the file index0-cod.txt and the line: sds\_obs; list(0.4) in the file index1-cod.txt. We will also set up a base-case file describing fishing mortality (F0-cod.txt), a file describing a stationary trajectory (M0-cod.txt), and specify that we do not want to run a retrospective analysis in the file R0-cod.txt. We will set up the file E0-cod.txt to fix the estimation of at the true value and case E1-cod.txt to estimate a stationary (Text S1).

All of these text files are available in the package data in the folder inst/extdata/eg-cases/. As an example, here is what the complete index0-cod.txt file looks like:

fleets; 2  
years; list(seq(1974, 2012, by = 2))  
sds\_obs; list(0.1)

fleets, years, and sds\_obs refer to the arguments in the function change\_index and users can read the help for this function with ?change\_index in R.

To start, we will load the ss3sim package into an R session and locate three sets of folders within the package data: the folder with the OM, the folder with the EM, and the folder with the plain-text case files:

library(ss3sim)  
d <- system.file("extdata", package = "ss3sim")  
om <- paste0(d, "/models/cod-om")  
em <- paste0(d, "/models/cod-em")  
case\_folder <- paste0(d, "/eg-cases")

## Running the simulations

It is important to validate a simulation testing framework with minimal or no process and observation error to ensure unbiased and consistent recovery of parameters under ideal conditions [@hilborn1992; @rykiel1996]. ss3sim makes this form of model validation simple by allowing users to specify process error (i.e. recruitment deviations) and sampling error (Text S1). Since, we have already validated the cod-like model setup (Text S1), we can now run our simulation scenario. We will set bias\_adjust = TRUE to enable a procedure that aims to produce mean-unbiased estimates of recruitment and biomass despite log-normal recruitment deviations [@methot2011]. We can run 100 iterations of the simulation scenarios with the following code:

run\_ss3sim(iterations = 1:100, scenarios =  
 c("D0-E0-F0-M0-R0-cod", "D1-E0-F0-M0-R0-cod",  
 "D0-E1-F0-M0-R0-cod", "D1-E1-F0-M0-R0-cod"),  
 case\_folder = case\_folder, om\_model\_dir = om,  
 em\_model\_dir = em, bias\_adjust = TRUE)

This produces a folder structure in our working directory containing all of the SS3 output files. We can then collect the output with one function call:

get\_results\_all()

This command creates two files in our working directory: ss3sim\_scalars.csv and ss3sim\_ts.csv, which contain scalar output estimates (e.g. steepness and maximum sustainable yield) and time-series estimates (e.g. recruitment and biomass each year). These estimates come from the report files produced from each run of SS3 and are read by the r4ss R package. The .csv files contain separate columns for OM and EM values, making it simple to calculate error metrics, such as relative or absolute error. In addition to parameter estimates, the .csv files contain performance metrics, such as the maximum gradient, whether the covariance matrix was successfully calculated, and the number of parameters stuck on a bound, which in combination can be used to gauge model performance and convergence. These results are organized into "long" data format, with columns for scenario and iteration, facilitating quick analysis and plotting using common R packages such as ggplot2 [@wickham2009].

For our example simulation, the relative error in spawning stock biomass over time is, as expected, smaller when the true value of is specified rather than estimated (Figure 2, top panels E0 vs. E1). Furthermore, lower precision in the research survey index of abundance results in greater relative error in spawning stock biomass in recent years (Figure 2, top panels D0 vs. D1), and greater relative error in terminal-year depletion and fishing mortality, but not in spawning stock biomass at maximum sustainable yield, or (Figure 2, lower panels).

# How ss3sim complements other simulation software

The general purpose of ss3sim is to explore model behaviour and performance across combinations of EM configurations and alternative dynamics of fisheries resources under exploitation specified by the OM. In particular, ss3sim provides a suite of functions for dynamically creating structural differences in both OMs and EMs. This expedites testing the properties of alternative stock assessment model configurations, whether the differences are between OMs and EMs [@johnson2013], or between multiple versions of EMs [@ono2013]. However, ss3sim is less suited for quickly exploring arbitrary SS3 model setups, which may rely on SS3 configurations not yet programmed into the ss3sim package functions. Therefore, depending on the simulation study goal, other software frameworks may provide better alternatives.

One alternative framework is *Fisheries Libraries in R* (FLR) [@kell2007] --- an open-source R package developed specifically for evaluating fisheries management strategies through simulation. Compared to ss3sim, FLR is designed to explore broader questions regarding management strategies with flexible biological, economic, and management components [@hillary2009]. Thus, it is not specifically designed to explore the impact of structural differences within OMs and EMs.

Another alternative stock assessment simulation testing framework is *Fishery Simulation* (FS, <http://fisherysimulation.codeplex.com>). FS is primarily a file management tool adapted to aid in simulation testing. FS can work with stock assessment models besides SS3, make simple changes to input text files, generate random process (using a built-in random number generator) and observation errors (using the SS3 bootstrap option), run simulations in parallel, and collect results from output files. Thus, FS is closer to ss3sim in its scope than FLR in that it specifically focuses on the performance of stock assessment models.

FS differs from ss3sim mainly in that it uses user-specified text manipulation commands (e.g. change line 50 from 0 to 1) to alter model configurations rather than the approach of ss3sim, which uses modular functions tailored to specific purposes (e.g. add a particular time-varying mortality trajectory to a particular OM). FS works well for testing arbitrary assessment models and model configurations because it does not rely on pre-built manipulation functions [@lee2012; @piner2011; @lee2011]. In contrast, FS cannot make complicated structural changes to a model setup (e.g. adding time-varying parameters or changing the survey years), limiting its ability to to induce and test structural differences between OMs and EMs. In addition, the current version of FS is not an end-to-end package --- additional code is necessary to incorporate process and observation error in simulation testing. Finally, although FS is also open-source, it requires the Microsoft .NET framework and is therefore only compatible with the Windows operating system.

# Research opportunities with ss3sim

The ss3sim package has been used so far to evaluate alternative assessment approaches when is thought to vary across time [@johnson2013], the importance of length- and age-composition data [@ono2013], and the causes of retrospective patterns in stock assessment models. Along with those studies, ss3sim makes many important research opportunities easily approachable. Below we outline some examples.

*Time-varying model misspecification*: Ecological processes can vary through time in response to, for example, changes to fishing behaviour [@hilborn1992], regime shifts [@vert-pre2013], or climate change [@walther2002]. However, parameters such as , catchability, and selectivity are commonly assumed to be time invariant and the consequences of assuming time invariance of such parameters when facing true temporal changes has been a long-standing discussion in fisheries science [@royama1992; @wilberg2006; @fu2001]. Furthermore, although studies have tried to isolate the effects of single time-varying parameters, such as [@lee2011; @jiao2012; @deroba2013; @johnson2013], few have considered the effect of multiple time-varying parameters and their potential interaction.

*Patterns in recruitment deviations*: Typically, estimation methods assume independent log-normally-distributed recruitment deviations around a spawning stock recruitment function. However, recruitment deviations are frequently auto-correlated and their variability can change through time [@beamish1995; @pyper1998]. ss3sim makes it simple to incorporate different recruitment deviation structures and test how they affect model performance.

*Retrospective patterns*: Retrospective patterns, in which model estimates are systematically biased with each additional year of data, are a major problem in stock assessment science [@mohn1999; @legault2008]. Key questions are: what causes retrospective patterns and what assessment approaches reduce retrospective patterns [@legault2008]. ss3sim can run retrospective analyses as part of any simulation by adding a single argument --- the number of retrospective years to investigate.

# Conclusions

The increasing complexity of modern integrated stock assessment models and expanding computing power allows for the inclusion of multiple sources of data and estimation of complex processes [@maunder2013]. However, the combination of complex models and large quantities of data are commonly associated with model misspecification, which can be difficult to detect based on residual patterns alone [@maunder2013]. Therefore, it is important to investigate the consequences of model misspecification. Investigating the consequences of model misspecification on the ability of assessment models to accurately and precisely estimate parameters is one important role of simulation testing [@wilberg2006; @deroba2013a; @crone2013].

Most simulation testing work to date has used custom frameworks tailored to the particular needs of each study [@helu2000; @yin2004; @magnusson2007; @wetzel2011a; @jiao2012; @wilberg2006; @deroba2013a; @deroba2013; @crone2013a; @hurtadoferro2013]. Although the complexity of many studies requires a custom framework, we encourage authors to publish their simulation frameworks, as we have done here, and where possible, to develop their simulation frameworks in a generalized format that allows others to build on them. The initial release of ss3sim describes the basic structure used in recent studies [@johnson2013; @ono2013] and the current version of ss3sim could be used to address other important questions in stock assessment science. We hope that users will both benefit from ss3sim in its current form and extend it for their own needs, potentially contributing back to future versions.

# Acknowledgements

We thank the participants and mentors of the University of Washington's School of Aquatic and Fishery Sciences 2013 FISH 600 course. Discussions with these individuals were instrumental to the conceptual and technical development of ss3sim. Many participants also contributed code and are listed within specific ss3sim R functions. Participants: Curry Cunningham, Felipe Hurtado-Ferro, Roberto Licandeo, Carey McGilliard, Melissa Muradian, Cody Szuwalski, Katyana Vert-pre, and Athol Whitten. Mentors: Richard Methot, Andre Punt, Jim Ianelli, and Ian Taylor. We thank Jim Ianelli, Andre Punt, and Ian Stewart for comments that improved our manuscript.

SCA was supported by Fulbright Canada (generously hosted by Trevor A. Branch), NSERC, and a Garfield Weston Foundation/B.C. Packers Ltd. Graduate Fellowship in Marine Sciences. KFJ and KO were partially supported by NOAA grant 423 NA10OAR4320148. This research addresses the methods component of the good practices guide to stock assessment program of the Center for the Advancement of Population Assessment Methodology (CAPAM).

# Figure Legends

# Tables

**Table 1. Main ss3sim functions and a description of their purpose.** Simulations can be run through the run\_ss3sim function, which then calls the change functions. Users can control what the change functions do through a series of plain-text case files. For example, the case ID D1 corresponds to the case files lcomp1, agecomp1, and index1, as described in the table. Users can also skip setting up case files and specify arguments to ss3sim\_base directly, or use the change functions as part of their own simulation structure (Text S1).

|  |  |
| --- | --- |
| Function name | Description |
| run\_ss3sim | Main high-level function to run ss3sim simulations. |
| ss3sim\_base | Underlying base simulation function. Can also be called directly. |
| change\_rec\_devs | Substitutes recruitment deviations. |
| change\_f | Adds fishing mortality time series. (Case file and ID F) |
| change\_tv | Adds time-varying features. For example, time-varying natural mortality, growth, or selectivity. (Any case file and ID, e.g. M, starting with "function\_type; change\_tv") |
| change\_index | Controls how the fishery and survey indices are sampled. (Case file index, case ID D) |
| change\_agecomp | Controls how age composition data are sampled. (Case file agecomp, case ID D) |
| change\_lcomp | Controls how length composition data are sampled. (Case file lcomp, case ID D) |
| change\_retro | Controls the number of years to discard for a retrospective analysis. (Case file and ID R) |
| change\_e | Controls which and how parameters are estimated. (Case file and ID E) |
| run\_bias\_ss3 | Determines the level of adjustment to ensure mean-unbiased estimates of recruitment and biomass. |
| get\_results\_scenario | Extracts results for a single scenario. |
| get\_results\_all | Extracts results for a series of scenarios. |