

Getting Started with Stock Synthesis (SS)

SS Development Team

Last Updated: September 09, 2019

1 Scope

Stock Synthesis (SS) is a statistical fisheries population dynamics modeling framework built in [AD Model Builder](#). The purpose of this document is to introduce users to running SS. We assume these users have some experience with population dynamics models and that they have a basic understanding of how to use the command line, but are new to SS.

If you have never used the command line, please see the resources in [Appendix 1](#).

To follow along, we recommend downloading the simple example from the [SS Examples Folder](#) within the [SS Document Library](#). Simple is a working SS model that runs quickly and should allow you to experiment with SS workflows without having to worry about the model's contents.

By the end of using this guide, you should:

1. Understand the input and output file structure for Stock Synthesis
2. Be able to run an SS model via the command line
3. Understand which tools are available to work with SS models
4. Troubleshoot basic problems associated with running an SS model

2 Stock Synthesis file structures and tools

SS uses text input files and produces text output files. In this section, the SS input and output files are described, as are tools that can be used with these files. See how they relate in Figure 1.

2.1 SS files: Required inputs

Four required input files are read by the SS executable. Throughout this document, we will refer to the SS executable as `ss.exe`. Keep in mind that the Linux and Mac versions of SS have no file extension, and the executable could be renamed to something else. These input files are:

1. **starter.ss:** Required file containing file names of the data file and the control file plus other run controls. Must be named `starter.ss`.
2. **data file:** File containing model dimensions and the data. The data file can have any name, as specified in the starter file, but typically ends in `.ss` or `.dat`.
3. **control file:** File containing set-up for the parameters. The control file can have any name, as specified in the starter file, but typically ends in `.ss` or `.ctl`.
4. **forecast.ss:** File containing specifications for reference points and forecasts. Must be named `forecast.ss`.



Figure 1: SS input and output files, and associated tools. Note that some tools are used with both input and output model files.

2.2 SS files: Optional inputs

The two optional input files for SS are:

1. **ss.par:** A text file with one line per parameter where order matters that could be created from a previous model run. This file is read in to overwrite the initial parameter values in the control file. The option to use the ss.par file or the control file for initial parameter values is selected in starter.ss.
2. **wtatage.ss:** File containing empirical input of body weight by fleet and population and empirical fecundity-at-age. The option to use the wtatage.ss file is selected in a line of the control file.

2.3 SS output files

After a model run, there are many output text files created. The most useful output files can be divided into:

1. **Files containing results.** These files include ss_summary.sso and report.sso that contain model results summarized in different ways.
2. **.ss_new files.** These files echo the SS input files, but include standardized comments. The values should be the same as the input files, except for the data.ss_new, which can also contain expected values from the model and bootstrapped data sets, if the user specifies so in the starter.ss file. The .ss_new files can be useful to standardize the comments in the input files and can be checked to make sure that SS interpreted the inputs as the user intended.
3. **Files used for debugging.** These files include warnings.sso and echoinput.sso.

2.4 Tool definitions

- **Any text editor** can be used to view and edit SS input and output files. Defining a custom syntax highlighting scheme for SS files may improve SS file readability. Some commonly used text editors include Atom, Emacs, Notepad++, and Sublime Text.
- The **Helper Excel Sheets** and **Excel Viewer (SS-OUTPUT)** are available in the [Helper Spreadsheets folder](#) of the NOAA Virtual Lab (Vlab) Stock Synthesis community document library. The helper excel sheets (also called helper spreadsheets) document inputs required given SS options selected and can be used as a reference while setting up model input files. The excel viewer is a tool for visualizing model results.
- **r4ss:** an R package to plot SS model results and manipulate SS input and output files. Available at: <https://github.com/r4ss/r4ss>
- **SSI:** Stock Synthesis Interface (i.e., the SS GUI). The latest version can be downloaded from [within the document library on Vlab](#). SSI can be used to edit, save, run, and visualize model inputs and outputs.

3 Running SS

SS is typically run through the command line. We will introduce three common approaches:

1. The one folder approach
2. The two folder approach
3. The path approach

Other possible approaches to running SS include:

1. Run SS via the command line from within R (discussed briefly)
2. Use SSI to set up model files and run (not discussed here; find the SSI and its documentation [within the Latest Executables folder](#) of the [Vlab SS community document library](#))

3.1 The one folder approach and demonstration of an SS model run

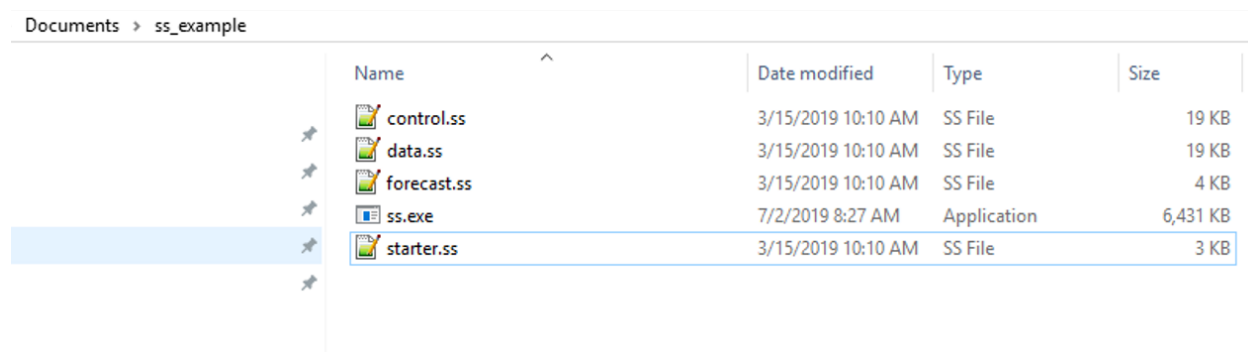
The one folder approach is so named because the model input files and SS executable are in the same folder. The one folder approach is the simplest way to run SS via the command line, so we will walk through an example run using this approach.

3.1.1 Setup for the 1 folder approach

Create a folder and add:

- Control File (Must match name in starter.ss)
- Data File (Must match name in starter.ss)
- forecast.ss
- ss.exe (or ss_opt.exe for running faster without internal checks)
- starter.ss
- Conditional files: wtatage.ss (if doing empirical wt-at-age approach) and/or ss.par (to continue from a previous run)

For example, here is what should be included for a model with no conditional files (where the control and data file names specified in starter.ss are control.ss and data.ss, respectively):



The screenshot shows a Windows File Explorer window with the address bar set to 'Documents > ss_example'. The main pane displays a list of files with columns for Name, Date modified, Type, and Size. The files listed are control.ss, data.ss, forecast.ss, ss.exe, and starter.ss. The 'ss.exe' file is highlighted with a blue selection bar.

Name	Date modified	Type	Size
control.ss	3/15/2019 10:10 AM	SS File	19 KB
data.ss	3/15/2019 10:10 AM	SS File	19 KB
forecast.ss	3/15/2019 10:10 AM	SS File	4 KB
ss.exe	7/2/2019 8:27 AM	Application	6,431 KB
starter.ss	3/15/2019 10:10 AM	SS File	3 KB

3.1.2 Run SS

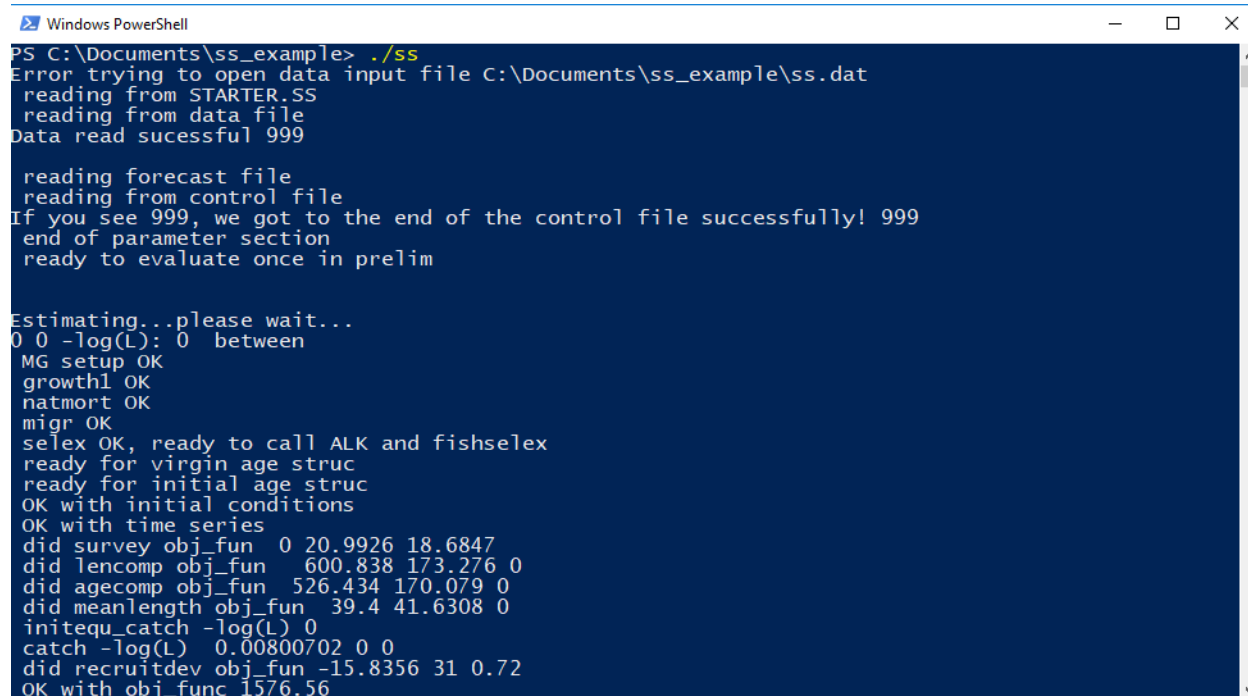
Once all of the model files and the SS executable are in the same folder, you can open your command window of choice at the location of the model files.

To do this, you can typically click to highlight the folder the model files are in, then shift + right click on the same folder and select the option from the menu to open the command line of choice (e.g., Windows Powershell). Then, type `ss` (or other name of the ss exe) into the command prompt and hit enter. Note that if you are using Windows Powershell, you will need to type `./ss`.

The exact instructions for running SS can differ depending on the command window used. If you have trouble, search for resources that describe running an executable for your specific command line.

If you still have issues, double-check that you are calling the correct exe name (e.g., you would need to type `ss_opt` to run an executable called `ss_opt.exe`.) and that the command line has the correct directory.

After starting the run, you should see output to the command line similar to:



```
Windows PowerShell
PS C:\Documents\ss_example> ./ss
Error trying to open data input file C:\Documents\ss_example\ss.dat
reading from STARTER.SS
reading from data file
Data read successful 999

reading forecast file
reading from control file
If you see 999, we got to the end of the control file successfully! 999
end of parameter section
ready to evaluate once in prelim

Estimating...please wait...
0 0 -log(L): 0 between
MG setup OK
growth1 OK
natmort OK
migr OK
selex OK, ready to call ALK and fishselex
ready for virgin age struc
ready for initial age struc
OK with initial conditions
OK with time series
did survey obj_fun 0 20.9926 18.6847
did lencomp obj_fun 600.838 173.276 0
did agecomp obj_fun 526.434 170.079 0
did meanlength obj_fun 39.4 41.6308 0
initequ_catch -log(L) 0
catch -log(L) 0.00800702 0 0
did recruitdev obj_fun -15.8356 31 0.72
OK with obj_func 1576.56
```

Note that the message `Error trying to open data input file` can be safely ignored. If you get past `Estimating...please wait...`, then the structure of the SS inputs is OK. The volume of information displayed during the run within the command window is controlled in `starter.ss` file on the line with the comment `# run display detail (0,1,2)` (1 is the most typically used value).

SS always opens and reads files in the same order (`starter.ss`, data file, `forecast.ss`, control, then `wtatage.ss` if using, and finally `ss.par` if using), writing to `echoinput.sso` as it reads. SS inputs are read in order by value, without reference to comments (i.e., anything after `#` on a line). Depending on which options are selected, SS will expect a particular number of values in a certain order, so failing to provide them will result in errors or at least a different model specification than intended.

After reading, SS proceeds immediately to pre-processing the data and creating internal parameter labels. As SS executes, it writes checks to `echoinput.sso` and warnings to `warnings.sso`.

SS next goes to the procedure section where iterative parameter changes are made by ADMB to minimize the negative log likelihood. When ADMB achieves convergence, control passes to the `sd_phase` for calculation of parameter variance (unless `-nohess` option is specified; See [command line options](#)). After the `sd_phase`, SS proceeds to the benchmark and forecast code section and then to final reporting. Output files containing results are written to the directory.

When the run completes, SS shows `!! Run has completed !!` and the number of warnings:

```
Windows PowerShell
Estimating row 47 out of 54 for hessian
Estimating row 48 out of 54 for hessian
Estimating row 49 out of 54 for hessian
Estimating row 50 out of 54 for hessian
Estimating row 51 out of 54 for hessian
Estimating row 52 out of 54 for hessian
Estimating row 53 out of 54 for hessian
Estimating row 54 out of 54 for hessian
do benchmark and forecast if requested in sdphase

In final section
Finish time: Wed Jul 10 10:39:37 2019
Elapsed time: 0 hours, 0 minutes, 57 seconds.
Final gradient: 9.30516e-005

finished COVAR.SSO
finished forecast for reporting
finished STD quantities for reporting
mceval counter: 0
finished posteriors reporting
finished SS_summary.sso
finished rebuild.sso
finished SStable.sso
writing big output now
finished report.sso
Write new starter file
Write new forecast file
Write new control file

!! Run has completed !! See warning.sso for N warnings: 1
PS C:\Documents\ss_example>
```

Examine warnings.sso for issues and suggestions, regardless of if the run completed successfully or not. If SS exits on error, you will not see the message `!! Run has completed !!`. Here is an example of a run that exited on error due to invalid inputs:

```
Windows PowerShell
PS C:\Documents\ss_example> ./ss
Error trying to open data input file C:\Documents\ss_example\ss.dat
  reading from STARTER.SS
  reading from data file
Error: Used invalid i = 0 for dmatrix rows bounded by [1, 3].
Assertion failed!

Program: C:\Documents\ss_example\ss.exe
File: linad99\dmatrix.cpp, Line 56

Expression: (index_min <= i && i <= index_max) || is_valid_row(i)

This application has requested the Runtime to terminate it in an unusual way.
Please contact the application's support team for more information.
PS C:\Documents\ss_example>
```

3.1.3 Examine the output

.ss_new files are generated if a model run completes. For more information on the .ss_new files, see the “Output Files” section of the SS user manual.

ss_summary.sso files are included with the examples on vlab so that you can compare it with the **ss_summary.sso** file generated by your model run. You should see similar if not exactly the same values for the likelihood and parameter values if using the same version of SS.

Output from SS can be read into r4ss or the excel viewer.

3.2 The two folder approach

With this approach, you can have one (or more) folders with various versions of SS and many model user folders each with one (or more) batch file(s) that point to various versions of SS. This eliminates the need

to have multiple copies of the same ss executable on your computer, which is necessary with the two folder approach.

In this approach, the ss.exe is no longer in the same folder as the model; instead, a batch file with a path to the SS executable is included in the folder with the model file.

The example shown in this handout uses batch files, which can be used on Windows only. However, there are similar workflows that can be used on OSX and Linux.

3.2.1 Example

Put the model files in one folder and the desired executable in the other:



Within the ss_exes folder, any number of ss exe versions could be added, but for now, we only have one:



The other folder contains the model files, but also a batch file, which is a text file with the extension .bat:



The batch file, a text file containing commands, can have any name, but must end in the extension .bat. In this case, we have called the batch file run_ss.bat. The first line of the batch file has the full path to the SS executable. For example, our batch file contains only the line:



Open the command window in the model user folder (in this example, the ss_example folder), then type

the name of the batch file (e.g., in this case, type `run_ss` or `./run_ss`). This will call the batch file, which will then call the `ss` executable to run in the folder with the model files. If the model runs, you will see output to the console as with the one folder approach.

This basic approach using a batch file can also be used to do multiple model runs, all called from the same batch file.

3.3 The PATH approach

The PATH is where your computer searches for files by default. This means that if an `SS` executable called `ss.exe` is in your path, when you type `ss` into the command window, regardless of the directory, your computer will be able to find `ss.exe` and use it with the model files in the current directory.

For more information on how to add Stock Synthesis to the computer's PATH, see [Appendix 2](#).

3.4 Run SS from within R

When the executable `ss.exe` is in the same folder as model input files, use `system("ss")` from the R console to run. When using `ss.exe` that is in your PATH in Windows, use:

```
get_bin <- Sys.which("ss.exe")[[1]] # get ss exe that is in your path
system(get_bin) # to run SS in current working directory
```

Running `SS` from within `R` may be desirable for setting up simulations where many runs of `SS` models are required (e.g., `ss3sim`) or if `r4ss` is being used to read model output.

4 Options for running SS

4.1 Command line options

ADMB options can be added to the run when calling the `SS` executable from the command line. The most commonly used options are:

- Skip standard errors (for quicker results or to get `Report.sso` if the hessian does not invert): `ss -nohess`
- To run without estimating anything (in theory): `ss -noest`. This will speed up your run by not optimizing, but some estimation (i.e., some parameters move away from their starting values) does seem to occur even with this option in `SS`. If you really want to run `SS` with no estimation, you should change the max phase in the `starter.ss` file to 0 and run the exe with the `-nohess` option.

To list all command line options, use one of these calls: `SS -?` or `SS -help`. More info about the `ADMB` command line options is available in the [ADMB Manual](#) (Chapter 12: Command line options).

4.2 Using `ss.par` for initial values

Typically, `SS` uses initial parameter values as specified in the parameter lines of the control file. However, initial values can be read from the `ss.par` file instead. To use `ss.par`, in `starter.ss` below the names of the data and control files, set the value to 1 rather than 0 on the line with comment

```
# 0=use init values in control file; 1=use ss.par.
```


Using the `ss.par` file comes in handy when you want to use different initial values without modifying the control file. An `ss.par` file is created during each model run and contains the ending parameter values (note that when you run a model, `ss.par` is one of the output files created). Using this `ss.par` file from a previous model run with a similar model (that still has the same parameters) will speed up run time because the initial values are closer to the maximum likelihood estimation (MLE) parameter estimates.

4.3 Using `wtatage.ss`

As stated in the manual, SS has the capability to read empirical body weight at age for the population and each fleet, in lieu of generating these weights internally from the growth parameters, weight-at-length, and size-selectivity. Please see the Empirical Weight-at-Age (`wtatage.ss`) section of the SS user manual for more information on using `wtatage.ss` with a model. There is also a weight at age SS example model available on `vlab` if you would like to try using this feature without having to build your own model.

5 Plotting results and basic troubleshooting

5.1 Using `r4ss` to organize and plot SS output

To plot the results of an `ss` model after it has run, use:

```
library(r4ss) # use r4ss package on github
# create list of quantities for the model in the directory mydir
replist <- SS_output(mydir)
SS_plots(replist) # create plots for SS run
```

See <https://github.com/r4ss/r4ss> for `r4ss` installation instructions. The master branch of `r4ss` is the most stable version, but the development branch contains the most recent fixes and developments. The `r4ss` version on CRAN was extremely out of date when this document was created, but it is probably best to check github to see if there is a recent CRAN release that could be used.

5.2 What to do when SS doesn't run

Here are some basic checks for when SS does not run:

- Make sure that all file names and directories are named correctly.
- Check that `starter.ss` references the correct names of the control and data files.
- If SS starts to read files and then crashes, check `warnings.sso` and `echoinput.sso`. The `warnings.sso` will reveal potential issues with the model, while `echoinput.sso` will show how far SS was able to run. Work backwards from the bottom of `echoinput.sso`, looking for where what SS reads does not match your inputs or where SS stopped.

For further information on troubleshooting, please refer to SS User Manual “Running SS” subsections, especially “Re-Starting a Run” and “Debugging Tips.”

6 Where to get additional help

- The [SS vlab website](#) resources, including the SS user manual and the SSI user guide.
- Email questions to nmfs.stock.synthesis@noaa.gov

- [Fisheries Research special issue \(vol. 142\) on Stock Synthesis](#)
- [Center for the Advancement of Population Assessment Methodology \(CAPAM\)](#) provides information about good practices in stock assessment

7 Appendix 1: An introduction to the command line

If you have never used the command line before, it will be helpful to learn a few basics. Some potential resources include:

- [Appendix A](#) to the Happy Git and GitHub for the useR book, which introduces the command line on Linux, OSX, and Windows systems.
- [Introduction to the Command Line by Launch School](#) has even more detail about using the command line specifically on Linux (which should also apply to OSX systems)
- A web search on “Command Line” will reveal plenty of other resources

8 Appendix 2: Putting SS in your PATH

This appendix is a slightly modified version of the ss3sim vignette section “Putting SS3 in your path.” The original version can be found in the [ss3sim repository](#).

Instead of copying the SS executable to each model folder, SS can be put in your system path, which is a list of folders that your operating system looks in whenever you type the name of a program on the command line. This approach saves on storage space since the SS binary (i.e., the SS executable or exe) is about 2.2 MB and having it located in each folder can be prohibitive in a large-scale simulation testing study. Even if you are not running a large simulation study, putting SS in your path may still be convenient, as you can use the same executable on many models, there is no need to specify a full file path to the executable each time you run a model, and no need to create a batch file that refers to the executable’s location.

8.1 For Unix (OS X and Linux)

To check if SS is in your path, assuming the binary is named SS: open a Terminal window and type `which SS` and hit enter. If you get nothing returned, then SS is not in your path. The easiest way to fix this is to move the SS binary to a folder that’s already in your path. To find existing path folders type `echo $PATH` in the terminal and hit enter. Now move the SS binary to one of these folders. For example, in a Terminal window type:

```
sudo cp ~/Downloads/SS /usr/bin/
```

You will need to use `sudo` and enter your password after to have permission to move a file to a folder like `/usr/bin/`.

Also note that you may need to add executable permissions to the SS binary after downloading it. You can do that by switching to the folder where you placed the binary (`cd /usr/bin/` if you followed the instructions above), and running the command:

```
sudo chmod +x SS
```

Check that SS is now executable and in your path:

```
which SS
```

If you followed the instructions above, you will see the following line returned:

```
/usr/bin/SS
```

If you have previously modified your path to add a non-standard location for the SS binary, you may need to also tell R about the new path. The path that R sees may not include additional paths that you have added through a configuration file like `.bash_profile`. If needed, you can add to the path that R sees by including a line like this in your `.Rprofile` file. (This is an invisible file in your home directory.)

```
Sys.setenv(PATH=paste(Sys.getenv("PATH"), "/my/folder", sep=":"))
```

8.2 For Windows

To check if SS is in your path for Windows: open a DOS prompt (either Command Prompt or Powershell should work) and type `SS -?` and hit enter. If you get a line like `SS is not recognized...` then SS is not in your path (assuming the SS executable is called `SS.exe`). To add the SS binary file to your path, follow these steps:

1. Find the correct version of the `SS.exe` binary on your computer (or download from the [SS vlab site](#)).
2. Move to and note the folder location. E.g. `C:/SS/`
3. Click on the start menu and type `environment`
4. Choose `Edit environment variables for your account` under Control Panel
5. Click on `PATH` if it exists, create it if does not exist
6. Choose `PATH` and click edit
7. In the `Edit User Variable` window add to the **end** of the `Variable value` section a semicolon and the SS folder location you recorded earlier. E.g. `;C:/SS`. **Do not overwrite what was previously in the PATH variable.**
8. Restart your computer
9. Go back to the DOS prompt and try typing `SS -?` and hitting return again.