An Introduction to *i*SC∀M

Steven Martell

University of British Columbia

s.martell@fisheries.ubc.ca

September 11, 2012

Outline

Obtaining *i*SC∀M source code

Source code available at:
`http://code.google.com/p/iscam-project/`

Use subversion to checkout a copy

On Mac & Linux, in Terminal:

```
svn checkout http://iscam-project.googlecode.com/svn/trunk/ iscam-project-read-only
```

On Windows use a subversion client
`http://tortoisesvn.net/`

Obtaining *i*SC∀M source code

The source code is maintained at github:
`https://github.com/smartell/iSCAM`

Prerequisites

- A C++ compiler (preferably gcc)
- AD Model Builder (version 11.0 or later)
- R (version 2.15 or later)
  - PBSmodelling package (and dependencies)
  - Hmisc package (and dependencies)

Working with Distributed Version Control (git)

Before using git, I would highly recommend spending some time learning how to use git. There are many online resources and most of them can be found at:
`http://git-scm.com/documentation`

A video tutorial:
`http://www.youtube.com/watch?v=ZDR433b0HJY`

Cheat sheet: `http://cheat.errtheblog.com/s/git/`

Read the Readme file for more instructions.

Initial checkout

Essentially want to make a clone of the repository on your computer.

```
git clone --depth 1 git://github.com/smartell/iSCAM.git
```

```
git clone --branch master --single-branch git://github.com/smartell/iSCAM.git
```

The above command will make a copy of the repository, including the directory structure, on your computer

Directory structure

- iSCAM-project
  - dist
  - docs
  - examples
  - fba
  - scripts
  - src

Directory structure

- iSCAM-project
  - dist
    - debug
    - R
    - release
  - docs
  - examples
  - fba
  - scripts
  - src

dist contains the compiled ADMB code in debug and release versions, and the R scripts for dealing with output.

Directory structure

- iSCAM-project
  - dist
  - docs
    - API
    - iSCAM-guide
    - userGuide
  - examples
  - fba
  - scripts
  - src

docs contains directories for the users guide, this presentation, and the API documentation for the source code.

The users guide and presentation is written in latex, and the API is built using doxygen.

Directory structure

- iSCAM-project
  - dist
  - docs
  - examples
    - 4VWXHerring
    - Cusk
    - ...
    - Makefile
    - makeproject
  - fba
  - scripts
  - src

examples directory contains several different examples and a Makefile for running the examples.

makeproject is a Unix script for setting up a new example directory.

Directory structure

- iSCAM-project
  - dist
  - docs
  - examples
  - fba
    - BC-herring-2011
    - makeproject
    - ReadMe.txt
  - scripts
  - src

fba is a directory for "full blown assessment"

The ReadMe.txt file documents the various projects, and makeproject is a Unix script for setting up a new assessment directory.

Directory structure

- iSCAM-project
  - dist
  - docs
  - examples
  - fba
  - scripts
    - scripts
  - src

scripts contains various scripts that are copied into assessment directories.

Directory structure

- iSCAM-project
  - dist
  - docs
  - examples
  - fba
  - scripts
  - src
    - admb-code
    - r-code

src contains directories for the ADMB source code and the R-code and source files for the R-package.

Editors

Windows

- Textpad `http://www.textpad.com/`

Mac OSX

- Textmate `http://macromates.com/`

Linux

- Vim `http://www.vim.org/`

Cross platform

- Emacs `http://www.gnu.org/s/emacs/`
- eclipse `http://www.eclipse.org/`
- sublime `http://www.sublimetext.com/`

Editors

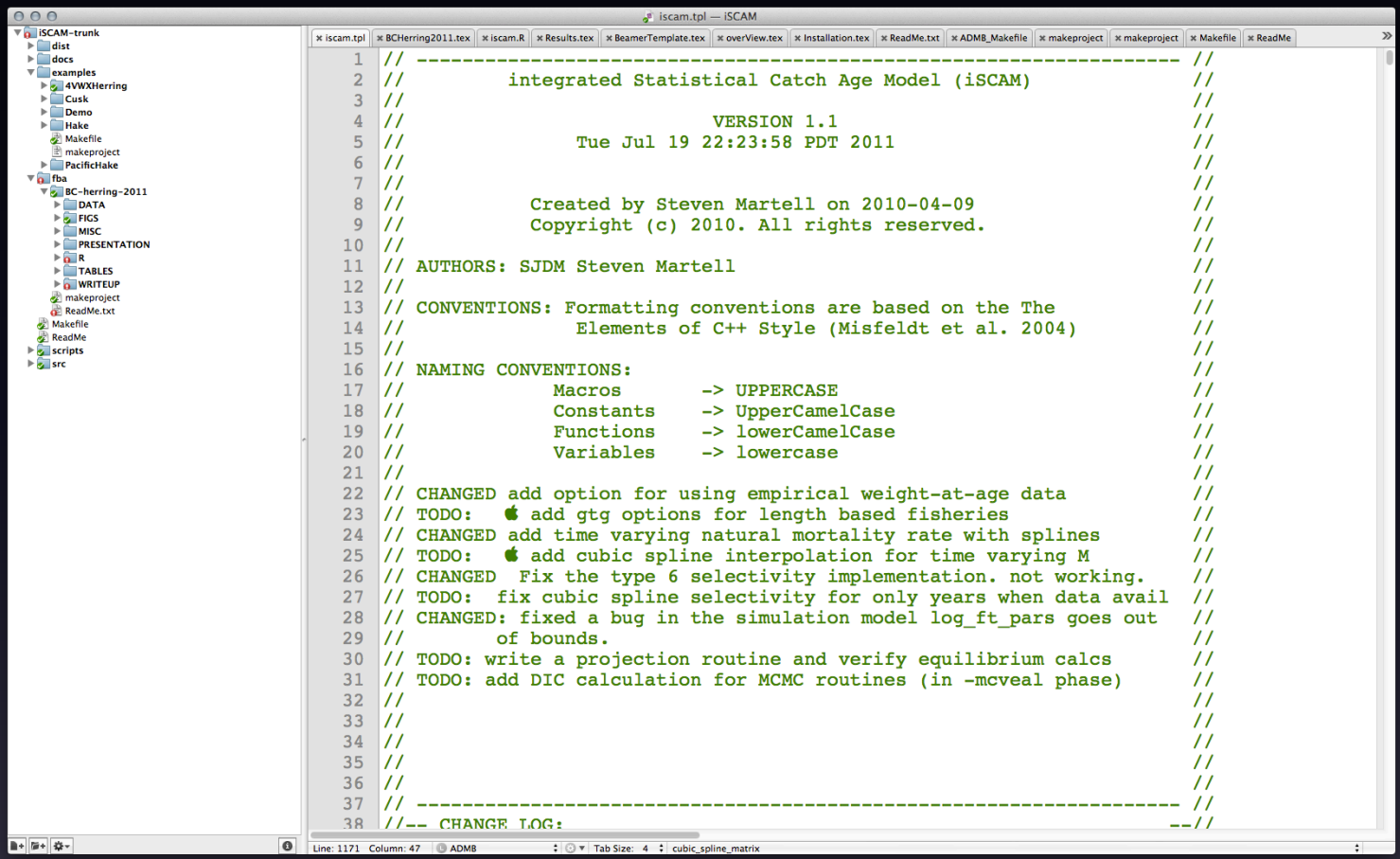


Figure: Textmate on Mac OSX

Outline

Compiling ADMB source code

What you need:

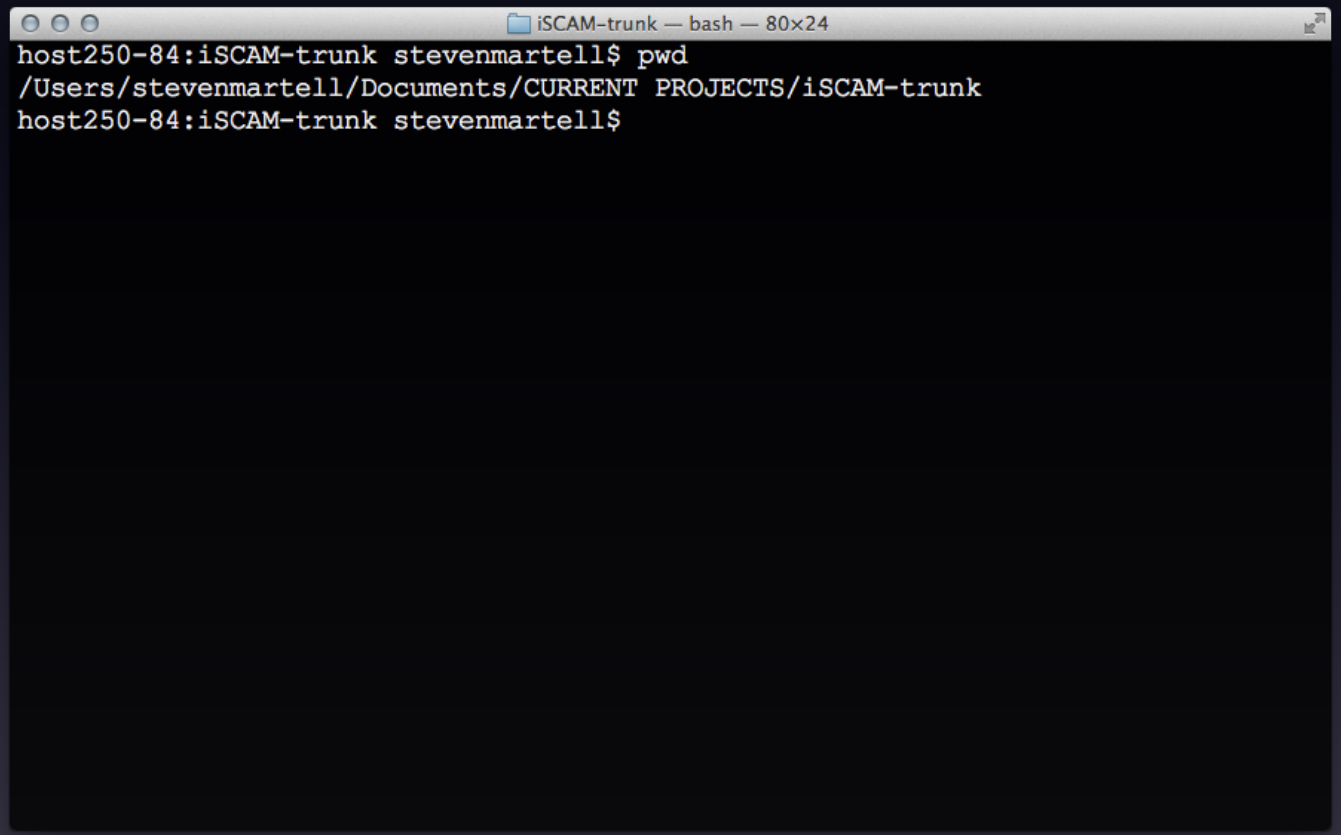
- C++ compiler (gcc recommended)
  - Mac OSX: install Xcode from appstore
  - Linux: `http://gcc.gnu.org/`
  - Windoze: `http://www.mingw.org/`
- ADMB libraries:
  `http://admb-project.org/downloads`

ADMB source code for *i*SC∀M found in:
`./iSCAM-trunk/src/admb-code/`

Compiling from the command line

At the command line:

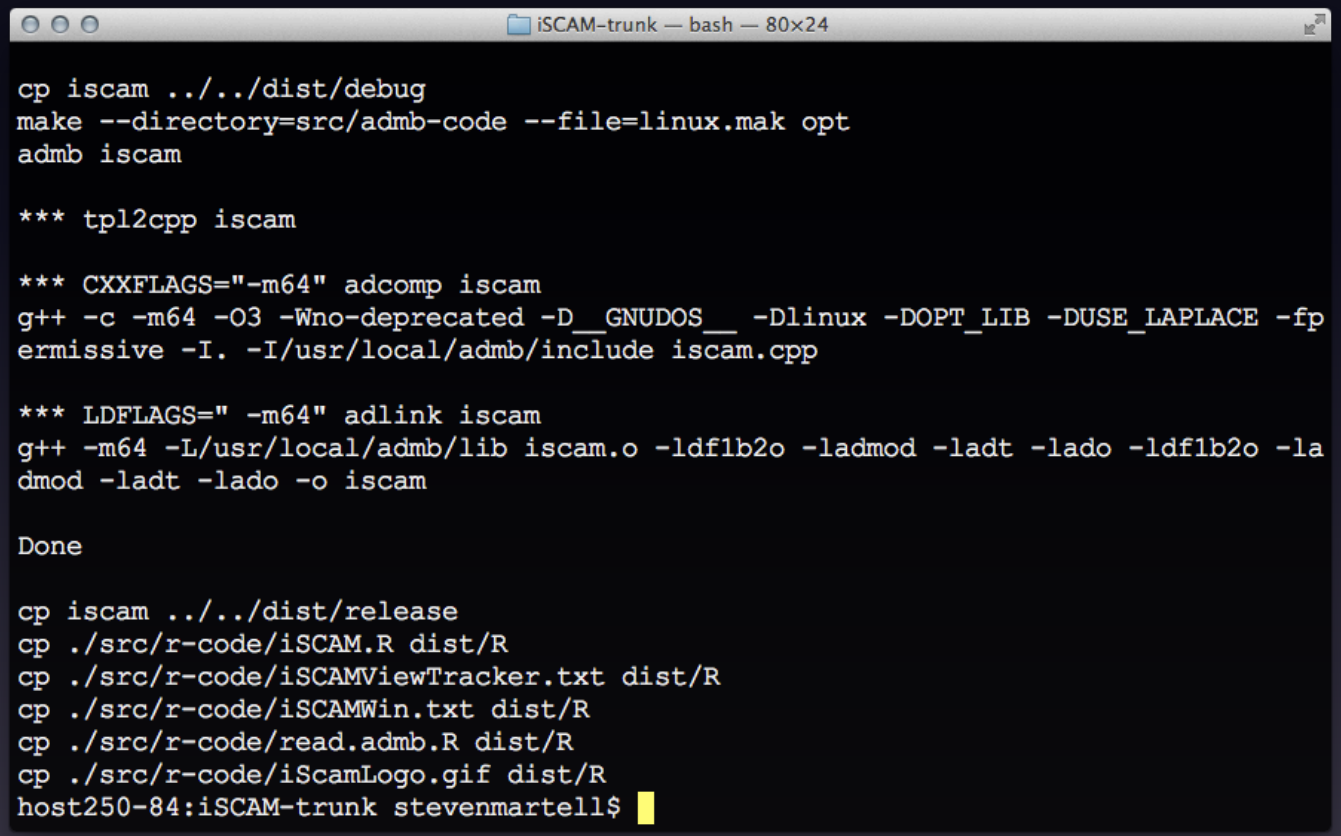
- use cd to navigate to the ./iSCAM-trunk directory



Compiling from the command line

At the command line:

- use cd to navigate to the ./iSCAM-trunk directory
- Linux or Mac OSX: type Make



Compiling from the command line

At the command line:

- use cd to navigate to the ./iSCAM-trunk directory
- Linux or Mac OSX: type Make
- Windows: see `http://gnuwin32.sourceforge.net/packages/make.htm`

Using the make file will compile the *i*SC∀M source code and place copies of the code in the distribution directory ("dist")

Using make on windoze machines

If you want to run makefiles on Windows that were written for Mac or Linux, you need to reinstall mingw. Make sure to check off "Developer tools" and "C++ libraries" and "Objective C libraries". Then run the mingw shell from the start menu and once inside that you can just type "make" as usual.

Outline

Running examples

Examples in `iSCAM-trunk/examples`

- `Demo`
- `Hake`

Demo

- The `Demo` directory is not present in the examples when you first checkout a copy of *i*SC∀M from the svn repository.
- To build the `Demo directory` cd to the examples directory and use `./makeproject Demo`

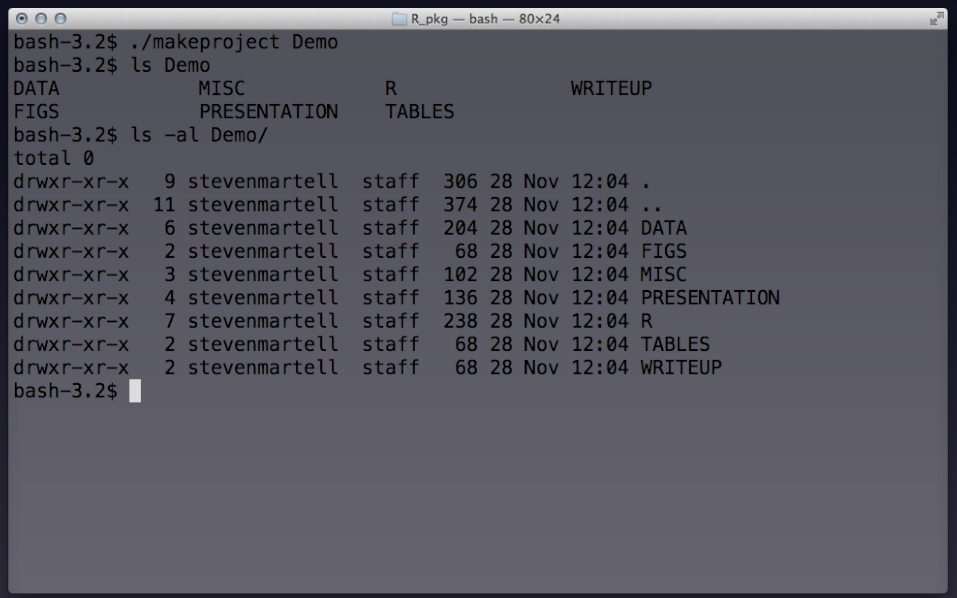


Figure: Using `makeproject` command to create `Demo`.

Running the ADMB model in Demo

- cd to the `examples/Demo/DATA` directory
- type `make` at the command line

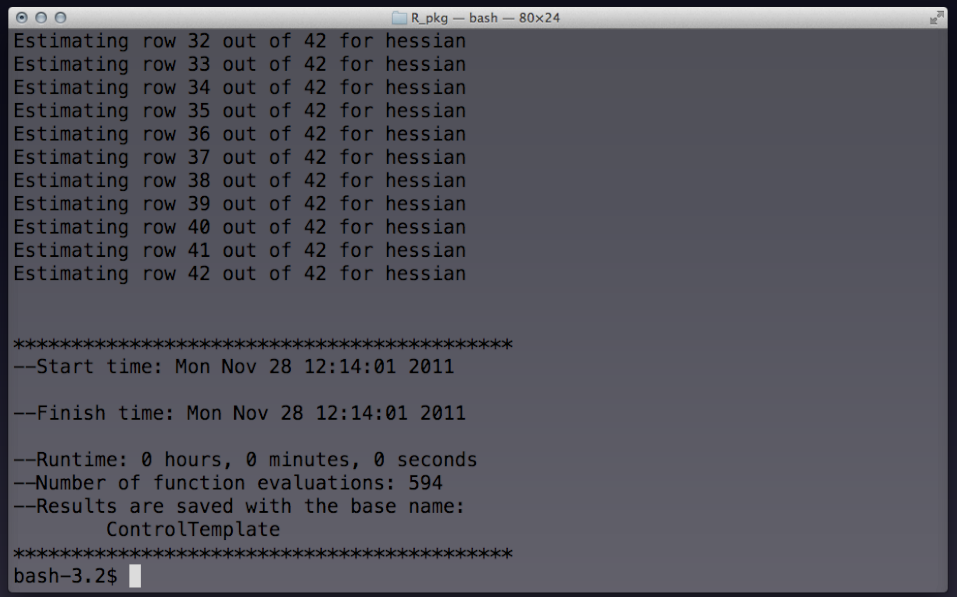


Figure: Terminal output after the Demo model has run

More on using `Makefile`

A makefile is a Unix utility that automatically executes a set of shell commands (rules). Target rules are executed based on dependencies.

Targets

- all: copy executable and run model with DAT & ARG
- run: copy executable and force a run
- mcmc: copy executable and run `mcmc` and `mceval`
- retro: copy executable and run retrospective R-script
- clean: remove executable & other ADMB output files

Dependencies

- EXEC - the name of the executable
- CTL - the name of the control file

If the dependencies change then running make will execute the target scripts, otherwise there is no need to re-run the model.

Setting up Makefile

User must supply variable Definitions in the Makefile:

```
EXEC    = iscam
prefix  = ../../../dist
DAT     = RUN.dat
CTL     = ControlTemplate
ARG     =
MCFLAG = -mcmc 10000 -mcsave 100 -nosdmcmc
NR      = 4
```

`EXEC` is the program name, `prefix` is the (relative) path to the `dist` directory, `DAT` is the data file, `CTL` is the name of the control file, `ARG` optional command line argument (e.g., make run ARG="-nohess"), `MCFLAG` is the arguments for `make mcmc`, and `NR` is number of retrospective years (e.g., `make retro`).

Using `make` at the command line

Makefiles are smart, will only execute rules if the dependencies change:

```
bash-3.2$ make
make: Nothing to be done for 'all'.
bash-3.2$
```

Using `make` at the command line

You can change the Makefile Defs at the command line:

```
bash-3.2$ make run ARG="-est -nox"
...
**********************************************************
--Start time: Tue Nov 29 11:51:10 2011

--Finish time: Tue Nov 29 11:51:11 2011

--Runtime: 0 hours, 0 minutes, 1 seconds
--Number of function evaluations: 424
--Results are saved with the base name:
        ControlTemplate
**********************************************************
bash-3.2$
```

Parallel execution with Make

Run multiple models in SUBDIR using: `make -j4`
The "-j" option specifies the number of processors to use.
`SUBDIR` is the list of subdirectories in `DATA` (one for each model)

```
## Makefile for running models
## Author: steven martell <martell.steve@gmail.com>

## Macros
SUBDIR = CC PRD QCI SOG WCVI AREA27 AREA2W
TARGET =
.PHONY: default $(SUBDIR) mcmc

## Targets
default: $(SUBDIR)
$(SUBDIR):
cd $@ && $(MAKE) $(TARGET)

.PHONY: clean
clean_files := $(foreach dir,$(SUBDIR),$(dir)/clean)

clean: $(clean_files)
$(clean_files):
cd $(@D) && $(MAKE) clean
```

Sorry does not work on WINDOZE!

Using guiView

In the R directory, source the iSCAM.R file in R >guiView()

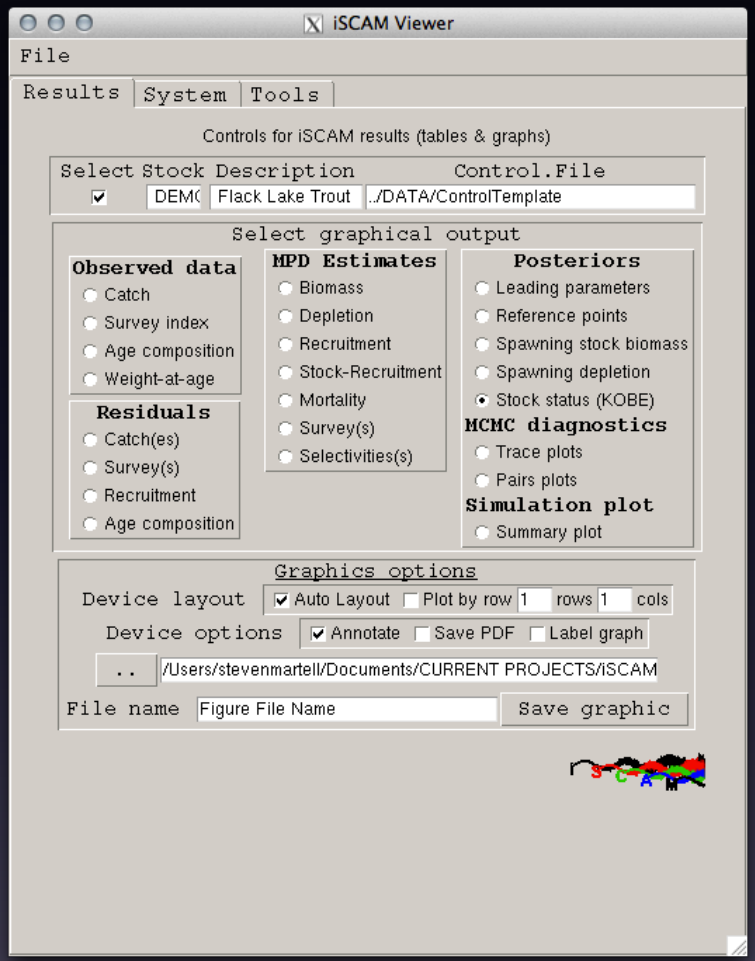


Figure: R gui for *i*SC∀M