

Installation

Source code  
SVN commands  
Directories  
Text Editors

Compiling

Running Examples

Demo Model  
Makefile

# An Introduction to *i*SCAM

Steven Martell

University of British Columbia

*s.martell@fisheries.ubc.ca*

November 30, 2011

# Outline

## Installation

Obtaining source code

SVN commands

Directories

Text Editors

## Compiling Source Code

## Running Examples

Demo Model

Makefile

### Installation

Source code

SVN commands

Directories

Text Editors

### Compiling

### Running Examples

Demo Model

Makefile

# Obtaining *i*SCAM source code



Steven Martell

## Installation

- Source code
- SVN commands
- Directories
- Text Editors

## Compiling

## Running Examples

- Demo Model
- Makefile

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/>

# Useful SVN commands

## Installation

- Source code
- SVN commands
- Directories
- Text Editors

## Compiling

## Running Examples

- Demo Model
- Makefile

Usage: `svn command`

Command	Description
<code>checkout</code>	Check out a working copy from a repository.
<code>info</code>	Display information about a local or remote item.
<code>update</code>	Bring changes from the repository into the working copy.
<code>log</code>	Show the log messages for a set of revision(s) and/or file(s).
<code>revert</code>	Restore pristine working copy file (undo most local edits).
<code>commit</code>	Send changes from your working copy to the repository.
<code>diff</code>	Display the differences between two revisions or paths.
<code>help</code>	Display help (usage <code>svn help &lt;command&gt;</code> )

# Directory structure

## Installation

Source code  
SVN commands  
Directories  
Text Editors

## Compiling

## Running Examples

Demo Model  
Makefile

- iSCAM-trunk
  - ▶ dist
  - ▶ docs
  - ▶ examples
  - ▶ fba
  - ▶ scripts
  - ▶ src

# Directory structure

## Installation

Source code  
SVN commands  
Directories  
Text Editors

## Compiling

## Running Examples

Demo Model  
Makefile

- iSCAM-trunk

- ▶ 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

## Installation

Source code  
SVN commands  
Directories  
Text Editors

## Compiling

## Running Examples

Demo Model  
Makefile

- iSCAM-trunk

- ▶ 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

## Installation

Source code  
SVN commands  
Directories  
Text Editors

## Compiling

## Running Examples

Demo Model  
Makefile

- iSCAM-trunk

- ▶ 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

## Installation

Source code  
SVN commands  
Directories  
Text Editors

## Compiling

## Running Examples

Demo Model  
Makefile

- iSCAM-trunk

- ▶ 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

## Installation

Source code  
SVN commands  
Directories  
Text Editors

## Compiling

## Running Examples

Demo Model  
Makefile

- iSCAM-trunk

- ▶ dist
- ▶ docs
- ▶ examples
- ▶ fba
- ▶ scripts
  - ▶ scripts
- ▶ src

`scripts` contains various scripts that are copied into assessment directories.

# Directory structure

## Installation

Source code  
SVN commands  
Directories  
Text Editors

## Compiling

## Running Examples

Demo Model  
Makefile

- iSCAM-trunk

- ▶ 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

## Installation

Source code  
SVN commands  
Directories  
Text Editors

## Compiling

## Running Examples

Demo Model  
Makefile

## 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

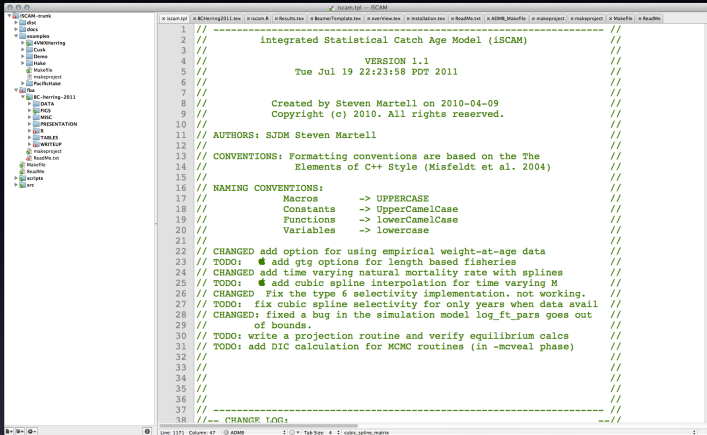
## Installation

- Source code
- SVN commands
- Directories
- Text Editors

## Compiling

## Running Examples

- Demo Model
- Makefile



```
1 //-----  
2 // integrated Statistical Catch Age Model (iSCAM) //  
3 //  
4 // VERSION 1.1 //  
5 // Tue Jul 19 22:23:58 PDT 2011 //  
6 //  
7 //  
8 // Created by Steven Martell on 2010-04-09 //  
9 // Copyright (c) 2010. All rights reserved. //  
10 //  
11 // AUTHORS: SJDM Steven Martell //  
12 //  
13 // CONVENTIONS: Formatting conventions are based on the The //  
14 // Elements of C++ Style (Misfeldt et al. 2004) //  
15 //  
16 // NAMING CONVENTIONS: //  
17 // Macros -> UPPERCASE //  
18 // Constants -> UpperCamelCase //  
19 // Functions -> lowerCamelCase //  
20 // Variables -> lowercase //  
21 //  
22 // CHANGED add option for using empirical weight-at-age data //  
23 // TODO: * add gtr options for length based fisheries //  
24 // CHANGED add time varying natural mortality rate with splines //  
25 // TODO: * add cubic spline interpolation for time varying M //  
26 // CHANGED Fix the type 6 selectivity implementation. not working. //  
27 // TODO: fix cubic spline selectivity for only years when data avail //  
28 // CHANGED: fixed a bug in the simulation model log_ft_pars goes out //  
29 // of bounds. //  
30 // TODO: write a projection routine and verify equilibrium calcs //  
31 // TODO: add DIC calculation for MCMC routines (in -mcveal phase) //  
32 //  
33 //  
34 //  
35 //  
36 //  
37 //  
38 //== CHANGE LOG: ==//
```

Figure: Textmate on Mac OS X

# Outline

## Installation

- Obtaining source code
- SVN commands
- Directories
- Text Editors

## Compiling Source Code

## Running Examples

- Demo Model
- Makefile

## Installation

- Source code
- SVN commands
- Directories
- Text Editors

## Compiling

## Running Examples

- Demo Model
- Makefile

# Compiling ADMB source code

## Installation

Source code  
SVN commands  
Directories  
Text Editors

## Compiling

## Running Examples

Demo Model  
Makefile

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 *iSCAM* 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

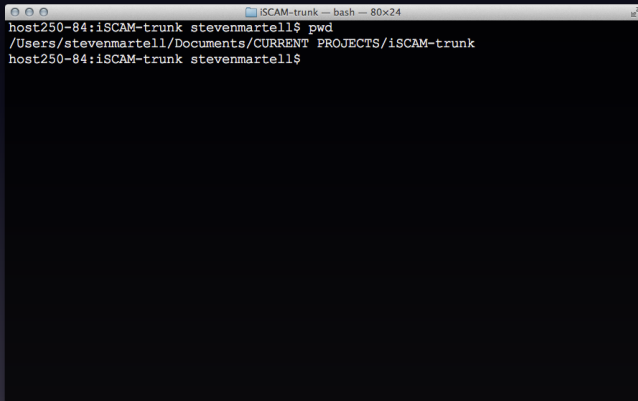
## Installation

Source code  
SVN commands  
Directories  
Text Editors

## Compiling

## Running Examples

Demo Model  
Makefile

A terminal window titled "iSCAM-trunk — bash — 80x24" is shown. The prompt is "host250-84:iSCAM-trunk stevenmartell\$". The user enters "pwd" and the terminal outputs "/Users/stevenmartell/Documents/CURRENT PROJECTS/iSCAM-trunk". The prompt returns to "host250-84:iSCAM-trunk stevenmartell\$".

```
host250-84:iSCAM-trunk stevenmartell$ pwd
/Users/stevenmartell/Documents/CURRENT PROJECTS/iSCAM-trunk
host250-84:iSCAM-trunk stevenmartell$
```



# Compiling from the command line

At the command line:

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

## Installation

Source code  
SVN commands  
Directories  
Text Editors

## Compiling

## Running Examples

Demo Model  
Makefile

```
iSCAM-trunk — bash — 80x24

cp iscam ../../dist/debug
make --directory=src/admb-code --file=linux.mak opt
admb iscam

*** tpl2cpp iscam

*** CXXFLAGS="-m64" adcomp iscam
g++ -c -m64 -O3 -Wno-deprecated -D_GNUDOS_ -Dlinux -DOPT_LIB -DUSE_LAPLACE -fpermissive -I. -I/usr/local/admb/include iscam.cpp

*** LDFLAGS=" -m64" adlink iscam
g++ -m64 -L/usr/local/admb/lib iscam.o -ldflb2o -ladmod -ladt -lado -ldflb2o -ladmod -ladt -lado -o iscam

Done

cp iscam ../../dist/release
cp ./src/r-code/iSCAM.R dist/R
cp ./src/r-code/iSCAMViewTracker.txt dist/R
cp ./src/r-code/iSCAMWin.txt dist/R
cp ./src/r-code/read.admb.R dist/R
cp ./src/r-code/iScamLogo.gif dist/R
host250-84:iSCAM-trunk stevenmartell$
```

# Compiling from the command line

## Installation

Source code  
SVN commands  
Directories  
Text Editors

## Compiling

## Running Examples

Demo Model  
Makefile

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 *iSCAM* source code and place copies of the code in the distribution directory ("dist")

# Outline

## Installation

Obtaining source code

SVN commands

Directories

Text Editors

## Compiling Source Code

## Running Examples

Demo Model

Makefile

### Installation

Source code

SVN commands

Directories

Text Editors

### Compiling

### Running Examples

Demo Model

Makefile

# Running examples

## Installation

Source code  
SVN commands  
Directories  
Text Editors

## Compiling

## Running Examples

Demo Model  
Makefile

## Examples in iSCAM-trunk/examples

- Demo
- Hake

# Demo

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

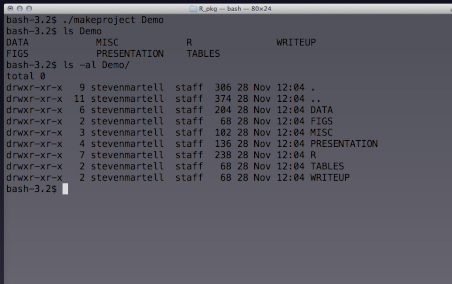
## Installation

Source code  
SVN commands  
Directories  
Text Editors

## Compiling

## Running Examples

Demo Model  
Makefile

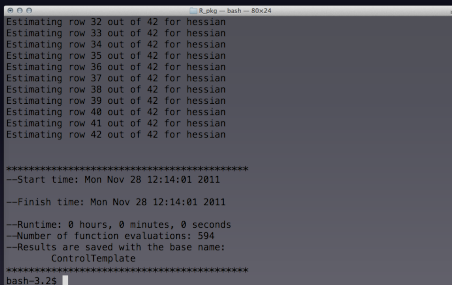


```
bash-3.2$ ./makeproject Demo
bash-3.2$ ls Demo
DATA          MISC          R              WRITEUP
FIGS          PRESENTATION  TABLES
bash-3.2$ ls -al Demo/
total 0
drwxr-xr-x  9 stevenmartell staff 306 28 Nov 12:04 .
drwxr-xr-x 11 stevenmartell staff 374 28 Nov 12:04 ..
drwxr-xr-x  6 stevenmartell staff 204 28 Nov 12:04 DATA
drwxr-xr-x  2 stevenmartell staff  68 28 Nov 12:04 FIGS
drwxr-xr-x  3 stevenmartell staff 102 28 Nov 12:04 MISC
drwxr-xr-x  4 stevenmartell staff 136 28 Nov 12:04 PRESENTATION
drwxr-xr-x  7 stevenmartell staff 238 28 Nov 12:04 R
drwxr-xr-x  2 stevenmartell staff  68 28 Nov 12:04 TABLES
drwxr-xr-x  2 stevenmartell staff  68 28 Nov 12:04 WRITEUP
bash-3.2$
```

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



```
R_pkg -- bash -- 80x24
Estimating row 32 out of 42 for hessian
Estimating row 33 out of 42 for hessian
Estimating row 34 out of 42 for hessian
Estimating row 35 out of 42 for hessian
Estimating row 36 out of 42 for hessian
Estimating row 37 out of 42 for hessian
Estimating row 38 out of 42 for hessian
Estimating row 39 out of 42 for hessian
Estimating row 40 out of 42 for hessian
Estimating row 41 out of 42 for hessian
Estimating row 42 out of 42 for hessian

*****
--Start time: Mon Nov 28 12:14:01 2011

--Finish time: Mon Nov 28 12:14:01 2011

--Runtime: 0 hours, 0 minutes, 0 seconds
--Number of function evaluations: 594
--Results are saved with the base name:
    ControlTemplate
*****
bash-3.2$
```

Figure: Terminal output after the Demo model has run

## Installation

Source code  
SVN commands  
Directories  
Text Editors

## Compiling

## Running Examples

Demo Model  
Makefile

# 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.

[Installation](#)[Source code](#)[SVN commands](#)[Directories](#)[Text Editors](#)[Compiling](#)[Running Examples](#)[Demo Model](#)[Makefile](#)

# 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).

## Installation

- Source code
- SVN commands
- Directories
- Text Editors

## Compiling

## Running Examples

- Demo Model
- Makefile



# Using make at the command line

## Installation

- Source code
- SVN commands
- Directories
- Text Editors

## Compiling

## Running Examples

- Demo Model
- Makefile

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$
```

Installation

- Source code
- SVN commands
- Directories
- Text Editors

Compiling

Running Examples

- Demo Model
- Makefile

# 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!

## Installation

- Source code
- SVN commands
- Directories
- Text Editors

## Compiling

## Running Examples

- Demo Model
- Makefile

# Using guiView

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

iSCAM

Steven Martell

Installation

Source code  
SVN commands  
Directories  
Text Editors

Compiling

Running Examples

Demo Model  
Makefile



Figure: R gui for iSCAM