AD Model Builder IDE

Emacs admb-mode without the Emacs Version 10.1 (2012-03-01) Revised manual (2012-04-04)

This is the manual for AD Model Builder IDE (ADMB-IDE) version 10.1. The latest edition of the manual is available at: http://admb-project.org/tools/admb-ide

Copyright © 2009, 2010, 2011, 2012 Arni Magnusson

To cite ADMB-IDE, use the newsletter article (see [References], page 20) as a fixed reference.

AD Model Builder IDE is an aggregate of six components:

- AD Model Builder 10.1 is released under the BSD License. Source code: http://code.google.com/p/admb-project/downloads/list
- GCC 4.5.2 is released under the GPL. Source code: ftp://ftp.gnu.org/gnu/gcc/
- GDB 7.4 is released under the GPL. Source code: ftp://ftp.gnu.org/gnu/gdb/
- GNU Emacs 23.4.1 is released under the GPL. Source code: ftp://ftp.gnu.org/gnu/emacs/
- ADMB Mode 6.8 for Emacs is released under the Simplified BSD License. Source code: http://admb-project.org/tools/editors/emacs/admb.el
- ADMB-IDE .emacs 10.1 is released under the Simplified BSD License. Source code: http://admb-project.org/tools/admb-ide/core

Table of Contents

1	P	${f reamble} \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots$	1
2	Ir	ntroduction	2
	2.1	Emacs admb-mode	. 2
	2.2	ADMB-IDE for Windows	
	2.3	ADMB-IDE for Linux/Mac OS	. 4
3	${f T}$	utorial	5
	3.1	Create a working copy of simple	. 5
	3.2	Build, run, and view the results	
	3.3	Debug	
4	Ir	nterface	13
	4.1	Menu	13
	4.2	Toolbar	14
	4.3	Shortcut keys	15
5	C	onfiguration	17
6	${f T}$	roubleshooting	18
	6.1	General usage	
	6.2	Configuration	
7	\mathbf{R}	deferences	20

1 Preamble

The purpose of ADMB-IDE is to make the convenient features of Emacs admb-mode available to non-Emacs users. In other words, to disable the standard Emacs behavior.

Experienced Emacs users may prefer to ignore the ADMB-IDE '.emacs' file, and simply install and load 'admb.el' like other Emacs packages. It is a standard "major mode" that follows all Emacs mode conventions.

2 Introduction

2.1 Emacs admb-mode

The process of creating statistical models with AD Model Builder (ADMB) involves writing, compiling, and testing. An integrated development environment (IDE) allows the user to perform these tasks more efficiently than with a basic editor and a shell.

GNU Emacs is a complex and powerful editor that comes with particularly good support for R, LATEX, backup/revision control, and other useful tools for statistical computing. Its admb-mode provides syntax highlighting, compilation, file manipulation, outline code navigation, templates, and smaller tools for creating ADMB models. Emacs users can fetch admb-mode from http://admb-project.org/tools/editors/emacs/admb.el and start using it right away, after reading the commentary at the top of the file.

The problem with Emacs is that it requires considerable time to learn and configure, although for heavy-duty statistical computing this can be a rewarding investment. As the programmer Larry Wall once said: "If ease of use was the highest goal, we'd all be driving golf carts." The http://admb-project.org/tools/editors/emacs page contains some pointers for setting up and learning Emacs. There are, however, good reasons why many users may not feel like adopting Emacs as their main editor, but would still appreciate a simple IDE for ADMB.

The rest of this tutorial demonstrates how Emacs with admb-mode can be configured as a user-friendly ADMB-IDE, without learning the details of Emacs. This is achieved with an unusual '.emacs' configuration file that emulates common keybindings of basic editors, while disabling some of the most used Emacs keybindings. This '.emacs' file is therefore not intended for experienced Emacs users, although they may find it an interesting read.

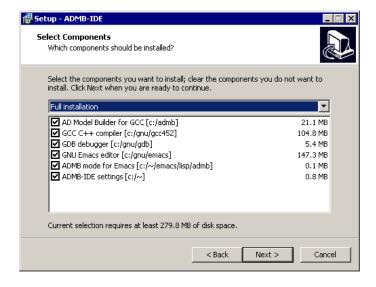
2.2 ADMB-IDE for Windows

Somebody set up us the bomb.

There are two ways to install ADMB-IDE for Windows: use an installer or set up the individual components by hand.

Installer

The 'admb-ide-101-win64.exe' installer sets up the five main components (ADMB, GCC, GDB, Emacs, admb-mode) with a customized Emacs user interface, file associations, and environment variables to glue everything together. The only catch is that the user must accept the default directory structure, and the full installation option is strongly recommended:



The director	v structure	is designed	d to accommod	ate coexisting	versions	of ADMB	and GCC:

Directory	Component	Overwrite
c:/admb/admb101-gcc452-win64	ADMB	Yes
c:/gnu/emacs	Emacs	Yes
c:/gnu/gcc452-win64	GCC	Yes
c:/gnu/gdb	GDB	Yes
c:/~/emacs/lisp/admb	admb-mode	Yes
c:/~	.emacs	No

It is perfectly safe to install ADMB-IDE on top of existing ADMB installations, which could reside in subdirectories of c:/admb or anywhere else. Compilers may also coexist, in subdirectories of c:/gnu or anywhere else. Short and shallow paths without spaces are helpful when configuring and writing customized scripts.

This is a practical setup for other free software as well. Take for example the R statistical software. By separating the main program (c:/gnu/r) from the user settings (c:/~/.Rprofile, c:/~/Rconsole) and user libraries (c:/~/r/library), the main program can be removed and upgraded without affecting the user setup.

As indicated in the table above, the installer will not overwrite an existing c:/~/.emacs file. This is because a personal '.emacs' configuration file is the only component that is irreplaceable; all other components are available and free on the internet.

One thing to keep in mind is that the installer modifies the user PATH and file associations. In rare cases, users may need to reconfigure these according to taste and needs after installing ADMB-IDE. Advanced users may choose to deselect these options during the installation for this reason.

Manual setup

Users can also set up and configure the individual components by hand, starting from the 'admb-ide-101-win64.zip' kit. The following guidelines may be useful for that:

```
http://admb-project.org/documentation
http://admb-project.org/tools/editors/emacs/install
http://admb-project.org/tools/editors/emacs/config
http://mingw.org/wiki/HOWTO_Install_the_MinGW_GCC_Compiler_Suite
```

See also the chapters on [ADMB-IDE for Linux/Mac OS], page 4 and [Troubleshooting], page 18 in this manual.

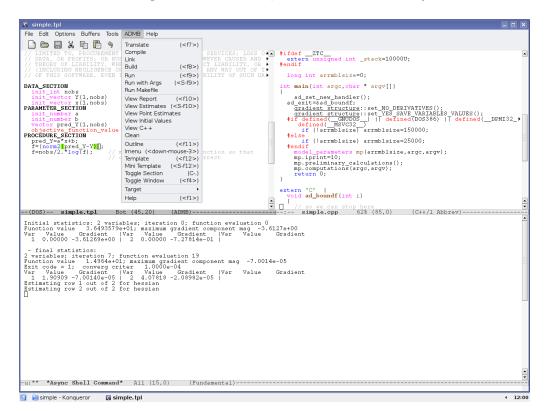
2.3 ADMB-IDE for Linux/Mac OS

Behold, the blueprints of a golf cart.

Setting up ADMB-IDE for Linux or Mac OS is equivalent to the "manual setup" described above, so the same guidelines apply. The key steps are:

- 1. Install ADMB, GNU Emacs, GCC (including the C++ component), and GDB.
- 2. Download the ADMB-IDE .emacs configuration file and place it in ~/.emacs to apply the customized Emacs user interface.
- 3. Download admb-mode, and place it in ~/emacs/lisp/admb/admb.el to provide ADMB syntax highlighting and IDE features.

The tutorial in the next chapter uses Windows, but ADMB-IDE is very similar in Linux:



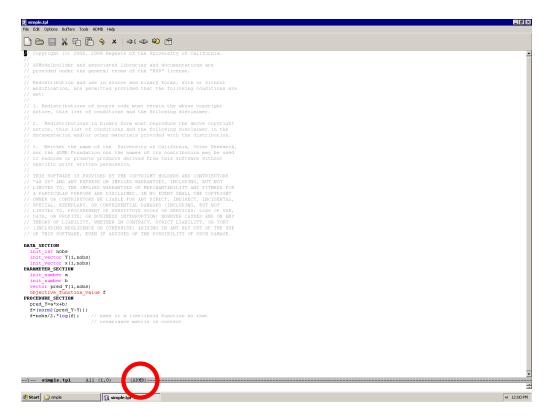
3 Tutorial

3.1 Create a working copy of simple

First open Windows Explorer and create a folder called c:/simple. Then navigate to c:/admb/admb101-gcc452-win64/examples/admb/simple and copy the model and data files, creating:

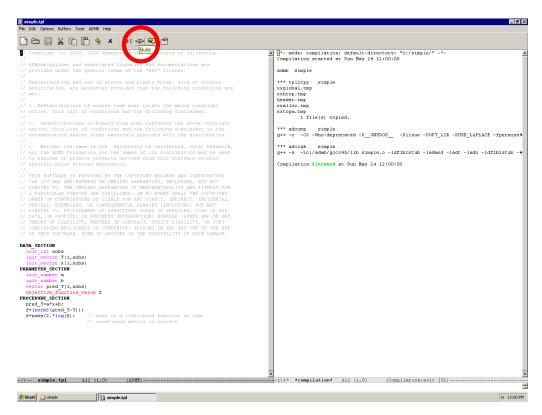
- c:/simple/simple.dat
- c:/simple/simple.tpl

Now double-click 'simple.tpl' in the c:/simple folder. The file should open in Emacs in admb-mode (see red circle) and the code should be in color:

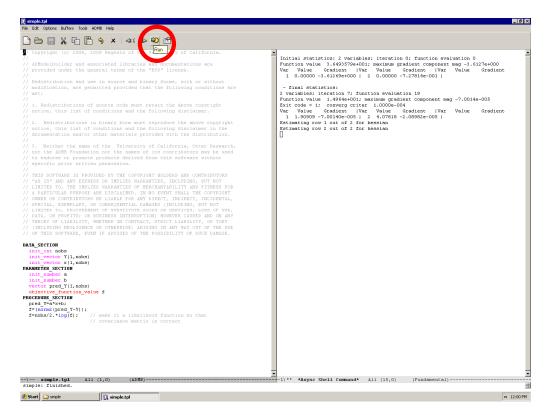


3.2 Build, run, and view the results

Build the model by clicking the \clubsuit icon, or press f8:



Run the model by clicking the icon, or press f9:

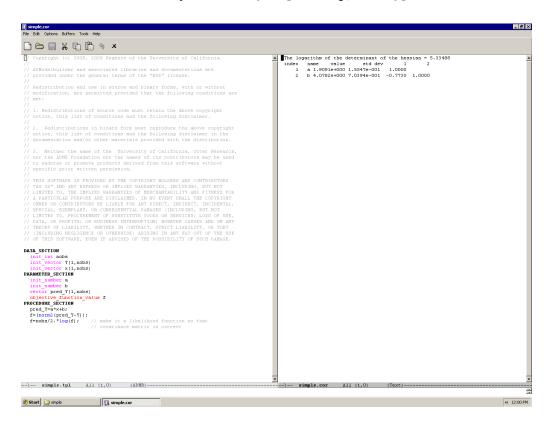


Many ADMB models output their results to a '.rep' report file, and ADMB-IDE provides the icon and f10 key to open the report file. The simple model outputs no report file, but the parameter estimates, standard errors, and correlations are found in the '.cor' file.

This is an opportunity to introduce basic buffer and window management. In Emacs, a buffer is like a page, often representing a file, but sometimes other things, like the compilation and command output buffers in the previous two screenshots. The Emacs screen is divided into one or more windows, where each window shows one buffer, while other buffers reside in the background. Explore the *Buffers* menu, as well as [Shortcut keys], page 15.

Try out different ways to open the '.cor' file:

- 1. Press escape to maximize the active window. Then click the icon or press C-o (Ctrl and o) and select c:/simple/simple.cor.
- 2. Press escape to maximize the active window, C-x 3 to split into two windows, and select the window on the right with a mouse click or f6. Click the icon or press C-o and select c:/simple/simple.cor.
- 3. Click the ADMB \rightarrow View Estimates menu entry or press S-f10.
- 4. Click the ADMB → View Any menu entry or press C-p, then type 'cor' and return.



After viewing, maximize a window by pressing escape, or close a window by clicking the \times icon or pressing C-w or C-f4.

Note how the ADMB menu and toolbar icons are only available when the active window is in admb-mode. Press f2 at any point to switch a window to admb-mode.

3.3 Debug

Types of bugs

Bugs in ADMB models can be categorized by the point of discovery:

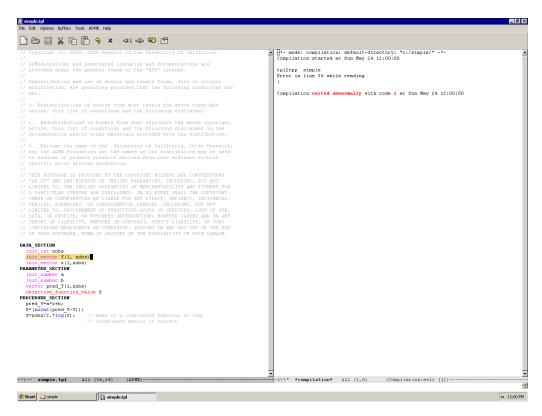
- 1. tpl2cpp reports a bug (cannot translate)
- 2. g++ reports a bug (cannot compile or link)
- 3. The model builds fine, but crashes or writes no output when run (no results)
- 4. The model runs fine, but not like it is supposed to (strange results)

Locating bugs

- Warnings or error messages indicate line number, or function/variable name
- Insert lines of code that print informative messages during runtime
- Comment out parts of the code
- Use a debugger

Example 1: tpl2cpp reports a bug

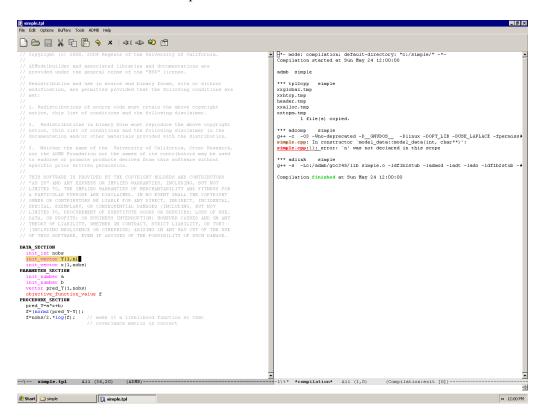
Create a bug by inserting an extra space inside a vector declaration: init_vector Y(1,nobs) → init_vector Y(1, nobs). Then click the **i icon or press f7 to translate TPL to C++:



The tpl2cpp translator reports an error in line 36 of 'simple.tpl'. Click $Edit \rightarrow Go \ To \rightarrow Goto \ Line$ or press C-g to move the cursor to that line, and then remove the unwanted space.

Example 2: g++ reports a bug

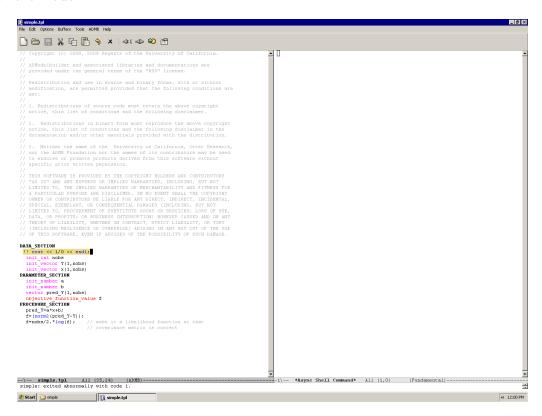
Create a bug by referring to a nonexistent variable: init_vector $Y(1,nobs) \rightarrow init_vector Y(1,n)$. Then click the icon or press f8 to build the model:



The g++ compiler reports an error in line 11 of 'simple.cpp'. Click the highlighted filename to open the C++ source file with the cursor in that line. After realizing what the problem is (with the help of the error message 'n' was not declared in this scope), go back to the ADMB code in 'simple.tpl' and change the 'n' to 'nobs'.

Example 3: No results

Create a bug by dividing by zero at the top of the DATA_SECTION: !! cout << 1/0 << endl;. Then click the ⇒ icon or press f8 to build the model. Ignoring the warning, click ⇒ or press f9 to run the model:



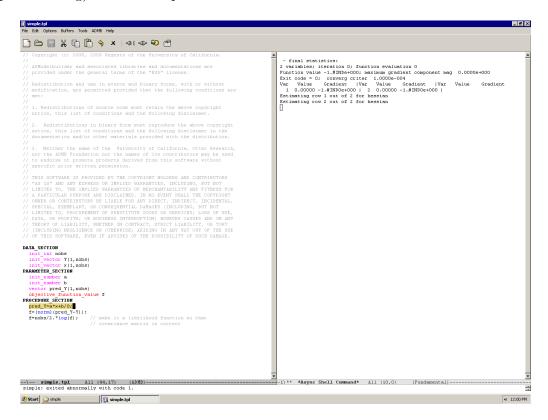
The shell command simple exits abnormally with code 1, a generic code for failure. The easiest way to search for this bug is to insert informative messages in the code, like

```
DATA_SECTION
cout << "DATA_SECTION begins" << endl;
...
cout << "DATA_SECTION ends" << endl;</pre>
```

and/or simplify the model, possibly by commenting out parts of the code. After narrowing the search step by step, the problematic line(s) can be changed or removed. To comment or uncomment large parts of code, use the M-; keystroke (see [Shortcut keys], page 15).

Example 4: Strange results

Create a bug by dividing by zero in a PROCEDURE_SECTION assignment: $pred_Y=a*x+b$; $\rightarrow pred_Y=a*x+b/0$;. Then click the \rightleftharpoons icon or press f8 to build the model (this time there is no compiler warning). Click \rightleftharpoons or press f9 to run the model:



The ADMB on-screen report indicates successful convergence (exit code 0) with an objective function value of '-1.#INDe+000', while Emacs reports failure (exit code 1). The easiest way to search for this bug is to insert informative messages in the code, like

PROCEDURE_SECTION

```
cout << "The value of a is: " << a << endl;
cout << "The value of a*x is: " << a*x << endl;
cout << "The value of b is: " << b << endl;
cout << "The value of b/0 is: " << b/>cout << "The value of f is: " << f << endl;</pre>
```

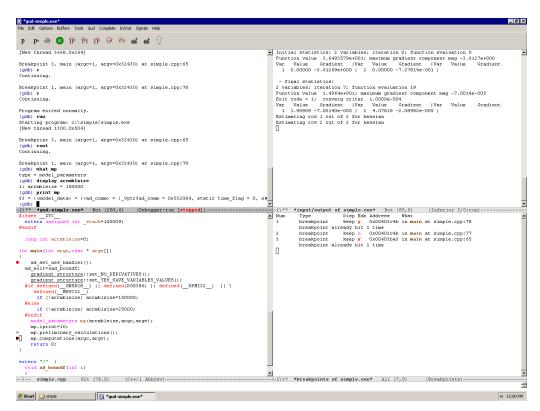
and/or simplify the model, possibly by commenting out parts of the code. A more advanced option is to use a debugger.

GDB: When the going gets tough

GNU Emacs and GCC can interact closely with the GDB debugger—these programs were all created by the same programmer Richard Stallman. A program must fulfill two conditions before debugging:

- The model executable (e.g. 'simple.exe') must build successfully, so a debugger is only helpful for bugs of type 3 and 4 (see [Types of bugs], page 8).
- The model executable must include debugging symbols. To embed debugging symbols with ADMB-IDE, either select "Debug" compilation from the ADMB → Target menu, or press C-- d (Ctrl and -, then d), followed by return.

Using GDB to debug an ADMB model is beyond the scope of this tutorial, but when simpler debugging methods fail, it is time to click $Tools \rightarrow Debugger$ (GDB):



4 Interface

4.1 Menu

Menu label	Purpose	Emacs command	
Translate	Translate TPL to C++	admb-tpl2cpp	
Compile	Compile C++ to object code	admb-compile	
Link	Link object code to exe	admb-link	
Build	Build executable from TPL	admb-make	
Run	Run executable	admb-run	
Run with Args	Run executable with args	admb-run-args	
Run Makefile	Run Makefile in current dir	admb-run-makefile	
View Report	Open .rep file	admb-rep	
View Estimates	Open .cor file	admb-cor	
View Point Estimates	Open .par file	admb-par	
View Initial Values	Open .pin file	admb-pin	
View C++	Open C++ file	admb-cpp	
View Any	Open model file	admb-open	
Clean Directory	Remove temporary files	admb-clean	
Outline	Navigate with outline	admb-outline	
Imenu	Navigate with imenu	imenu	
Template	Insert template	admb-template	
Mini Template	Insert minimal template	admb-template-mini	
Toggle Section	Toggle section indicator	admb-toggle-section	
Toggle Window	Toggle secondary window	admb-toggle-window	
Target	Choose what to build	admb-set-flags	
Help	Show help page	admb-help	
ADMB Mode Version	Show ADMB Mode version	admb-mode-version	

4.2 Toolbar

Icon	Purpose	Emacs command
<u> </u>	New buffer	new-buffer
_		11011 2 41 1 01
	Open file	find-file
	Save file	save-buffer
Ж	Cut	kill-region
	Copy	copy-region-as-kill
	Paste	cua-paste
¬	Undo	undo
×	Close	kill-this-buffer
= 0€ (Translate TPL to C++	admb-tpl2cpp
=⊕	Build executable from TPL	admb-make
	Run executable	admb-run
	Open .rep file	admb-rep

4.3 Shortcut keys

In combinations, 'S-' means Shift, 'C-' means Ctrl, and 'M-' means the Alt key.

Keystroke	Purpose	Emacs command
f1	Help	admb-help
S-f1	Show ADMB-IDE version	admb-ide-version
f2	ADMB mode	admb-mode
f3	Data mode	conf-unix-mode
f4	Toggle secondary window	admb-toggle-window
C-f4	Close	kill-buffer-maybe-window
M-f4	Quit	save-buffers-kill-emacs
f5	Reload	revert-buffer
f6	Other window	other-window
C-f6 / M-f6	Next buffer	next-buffer
f7	Translate TPL to C++	admb-tpl2cpp
f8	Build executable from TPL	admb-make
f9	Run executable	admb-run
f10	Open .rep file	admb-rep
S-f10	Open .cor file	admb-cor
f11	Navigate with outline	admb-outline
f12	Insert template	admb-template
S-f12	Insert minimal template	admb-template-mini
C	Toggle compilation flags	admb-toggle-flag
C-,	Toggle trailing whitespace	toggle-trailing-whitespace
C	Toggle section indicator	admb-toggle-section
C-a	Select all	mark-whole-buffer
C-b	Next buffer	next-buffer
C-c	Copy	cuaprefix-override-handler
C-f	Find, find next	isearch-forward
C-g	Goto line	goto-line
C-1	Recenter	recenter
C-n	New	new-buffer
C-o	Open	find-file
С-р	Open in other window	admb-open
C-q	Quit	save-buffers-kill-emacs
C-r	Replace	query-replace
C-s	Save	save-buffer
C-S	Save as	write-file
C-v	Paste	cua-paste
C-w	Close	kill-buffer-maybe-window
C-x	Cut	cuaprefix-override-handler
C-x 2	Split window above/below	split-window-vertically
C-x 3	Split window left/right	split-window-horizontally
C-z	Undo	undo
C-return	Rectangle functions	cua-set-rectangle-mark
C-space	Expand recognized words	dabbrev-expand
C-M-space	Open recent files	recentf-open-files

М-,	Delete trailing whitespace	delete-trailing-spc-tab-m
M-;	Comment/uncomment region	comment-dwim
escape	Cancel dialog, maximize window	keyboard-escape-quit

Mouse button	Purpose	Emacs command
C-left right	Switch buffers Navigate with imenu	mouse-buffer-menu imenu

ADMB-IDE does not emulate perfectly the way many editors open menus with the Alt key. To open the *Edit* menu, for example, it is not enough to press Alt and e simultaneously. ADMB-IDE provides four ways to open the *Edit* menu:

- 1. Mouse click on the menu bar
- 2. Tap Alt first and then e (Windows)
- 3. Hold Alt and tap e twice (Windows)
- 4. Hold Alt and tap f, then release Alt and tap Right arrow (Linux)

The idea behind ADMB-IDE, however, is that users can memorize intuitive keystrokes to undo, cut, copy, paste, find, replace, and goto line, without opening the <code>Edit</code> menu. Also don't forget that ADMB-IDE is open source, so users are free to modify any part of the program, including the keybindings defined in the '.emacs' file.

Longtime users of the Vi editor can turn on Emacs evil-mode or viper-mode, which are full-featured Vi emulators for Emacs.

5 Configuration

Personal .emacs file

ADMB-IDE is intended for people who don't know Emacs, are not interested in learning it, and will only use it to work with ADMB. The design goal is that ADMB-IDE should work out of the box and get the job done with minimum fuss.

It is, however, in the nature of modellers to experiment and improve. Users who modify the original '.emacs' file are no longer using ADMB-IDE, but Emacs with admb-mode and a personal '.emacs' file. One reason to modify the '.emacs' file or write a new one from scratch is to install additional packages like ESS (Emacs Speaks Statistics). Another reason is to redefine the keybindings, possibly closer to the Emacs defaults, e.g. to enable the C-h help system and to keep the fingers on the home row. Other reasons include setting fonts and colors, setting user variables, or defining new user functions. Users with a personal '.emacs' file can update ADMB, Emacs, GCC, GDB, and admb-mode independently, or salvage pieces from a recent ADMB-IDE zip bundle.

Note that it is not advisable to configure Emacs by clicking Options \rightarrow Save Options or Options \rightarrow Customize Emacs. Editing the '.emacs' file directly is a more reliable and transparent approach. See http://admb-project.org/tools/editors/emacs for guidelines.

Example

```
The f10 key in ADMB-IDE runs admb-rep to a open a report file:

(local-set-key [f10] 'admb-rep ); menu-bar-open
```

The semicolon starts a comment, reminding that the default behavior of Emacs is to run menubar-open when f10 is pressed. In ADMB-IDE, it is easy to activate the menu bar with the mouse or the Alt key, so f10 can be used for something else. The liberal spacing is to align surrounding lines of code, but neither the comment nor multiple spaces are necessary.

Some users may find it practical to open the report file in an external browser, rather than inside ADMB-IDE. The report file is often best viewed in a large window, and the ADMB-IDE windows are somewhat busy showing other things. It is easy to rebind the f10 key,

```
(local-set-key [f10] 'admb-rep-browser ); menu-bar-open
```

but as mentioned in the documentation of admb-rep-browser, the '.rep' file ending in Windows may need to be associated with the desired browser program, Firefox or the like.

Providing ADMB-IDE while retaining a personal .emacs file

It can be practical to make a canned version of ADMB-IDE available, while using a different '.emacs' file for most Emacs sessions. For example, an experienced Emacs user may want to test how ADMB-IDE works, or demonstrate it to colleagues, without constantly shuffling '.emacs' files. In Windows, one can place the ADMB-IDE '.emacs' file in c:/admb/ide and then start ADMB-IDE with the shell command:

```
c:/gnu/emacs/bin/runemacs.exe -Q -l c:/admb/ide/.emacs -f admb-mode
```

The '-Q' option tells Emacs to ignore the default startup file(s), the '-1' tells it to load a Lisp file, and the '-f' tells it to call a function. This command can be used in a start menu or desktop shortcut, with the c:/~/icons/admb64.ico decorative icon, and similar tricks can be used in Linux and Mac OS.

6 Troubleshooting

6.1 General usage

The ADMB menu and toolbar icons disappear

These only appear when the current buffer is in admb-mode. Either switch to a '.tpl' buffer that is already in admb-mode, or press f2 to switch to admb-mode in the current buffer. Other modes may have special menus and toolbar icons that are useful for that mode, see for example the [GDB screenshot], page 12.

Undo is confusing and redo is missing

The undo feature in ADMB-IDE does both undo and redo. When undo is performed repeatedly, it goes further back in the undo history. Any command other than undo will interrupt this sequence, and from that point the previous undo commands become ordinary changes that can be undone, equivalent to redo. Try, for example, copying some text and then paste it three times. Now undo three times, interrupt with a harmless key like the Up arrow, and then undo again to redo. To undo all changes since last save, it's easiest to reload using the f5 key.

The Tab key does not indent code properly

ADMB-IDE does not know the appropriate indentation of every line, so generally users indent their code manually using Space and Backspace. The Tab key is programmed to insert a number of spaces, as suggested by the previous line, which is sometimes useful.

Lines end with strange ^M characters

This is how Emacs shows Dos line endings, although in most cases Dos line endings are handled more gracefully. It could be that the file contains mixed line endings (both Dos and Unix), and the simplest solution is to delete all ^M characters. It could also be that the Emacs variable file-name-buffer-file-type-alist matches the '.tpl' file ending, and the simplest solution is to set that variable to nil.

Clicking ADMB \rightarrow Run Makefile returns an error

Makefiles are a sophisticated build automation tool, not required for general ADMB usage. This command invokes the make program that looks for a file called 'Makefile'. If the make program or the makefile is not found, an error is returned. This feature is provided for advanced users who have prepared a makefile in the working directory.

6.2 Configuration

Double-clicking a '.tpl' file in Windows Explorer does not open it in Emacs

The '.tpl' file ending needs to be associated with Emacs. This can be done with registry entries or in Windows Explorer folder options.

Emacs cannot load admb-mode

The directory containing the 'admb.el' file needs to be in the Emacs variable load-path, and the admb-mode command needs to be autoloaded in the '.emacs' configuration file.

Compilation commands are not recognized

The PATH environment variable needs to point to the directories containing the compilation programs (tpl2cpp.exe, tpl2rem.exe), scripts (admb.bat, adcomp.bat, adlink.bat), and the g++.exe program. Likewise, the ADMB_HOME environment variable needs to point to the main ADMB directory. Windows environment variables can be set using Dos scripts like c:/autoexec.bat and c:/~/bat/admb-set.bat, or by right-clicking the My Computer icon, then $Properties \rightarrow Advanced \rightarrow Environment Variables \rightarrow User variables \rightarrow New.$

Limited user (i.e. non-administrator) accounts in Windows can also prevent the ADMB-IDE installer from setting the environment variables ADMB_HOME and PATH. In those cases, the right-click-properties method described above can be used to set the variables after the installation. Many Linux distributions include only the C component of GCC, so users need to install the optional C++ component before using ADMB.

7 References

ADMB

Fournier, D.A., H.J. Skaug, J. Ancheta, J. Ianelli, A. Magnusson, M.N. Maunder, A. Nielsen, and J. Sibert. 2012. *AD Model Builder: Using automatic differentiation for statistical inference of highly parameterized complex nonlinear models.* Optimization Methods and Software 27:233–249.

Fournier, D. 2011. An introduction to AD Model Builder for use in nonlinear modeling and statistics. Version 10.0.

http://admb-project.org/documentation/manuals/admb-user-manuals

Skaug, H. and D. Fournier. 2011. Random effects in AD Model Builder: ADMB-RE user guide. Version 10.0.

http://admb-project.org/documentation/manuals/admb-user-manuals

Fournier, D. 2011. AUTODIF: A C++ array language extension with automatic differentiation for use in nonlinear modeling and statistics. Version 10.0.

http://admb-project.org/documentation/manuals/admb-user-manuals

Magnusson, A. 2009. ADMB-IDE: Easy and efficient user interface. ADMB Foundation Newsletter 1(3):1–2.

http://admb-foundation.org/wp-content/uploads/Newsletter/ADMBNewsletterJuly2009.pdf

Emacs

Stallman, R. 2011. *GNU Emacs manual*. 16th ed. Updated for Emacs version 23.3. http://www.gnu.org/software/emacs/manual/emacs.html

Lewis, B., D. LaLiberte, R. Stallman, and the GNU Manual Group. 2011. GNU Emacs Lisp reference manual for Emacs version 23.3. Rev. 3.0.

http://www.gnu.org/software/emacs/manual/elisp.html

Chassell, R. 2009. An introduction to programming in Emacs Lisp. Rev. 3.10.

http://www.gnu.org/software/emacs/emacs-lisp-intro/

GCC

Stallman, R.M. and GCC Developer Community. 2010. Using the GNU Compiler Collection. Version 4.5.2.

http://gcc.gnu.org/onlinedocs/

GDB

Stallman, R., R. Pesch, S. Shebs, et al. 2012. Debugging with GDB: The GNU source-level debugger. 10th ed. for GDB version 7.4.

http://sourceware.org/gdb/download/onlinedocs/