

TMB-IDE

Emacs `tmb-mode` without the Emacs
Version 1.6-2 (2015-10-28)

Arni Magnusson

This is the manual for Template Model Builder IDE (TMB-IDE) version 1.6-2.

The latest edition of the manual is available at:

<ftp://ftp.hafro.is/pub/tmb/tmb-ide.pdf>

Copyright © 2015 Arni Magnusson

The TMB-IDE `.emacs` configuration file is released as free software under the GNU General Public License (GPL).

Table of Contents

| | | |
|----------|-----------------------------------|-----------|
| 1 | Preamble | 1 |
| 1.1 | Credit | 1 |
| 1.2 | A simplified Emacs | 1 |
| 1.3 | System requirements | 1 |
| 2 | Installation | 2 |
| 2.1 | Configuration file | 2 |
| 2.2 | Shortcut | 2 |
| 3 | Introduction | 3 |
| 3.1 | Emacs <code>tmb-mode</code> | 3 |
| 4 | Tutorial | 4 |
| 4.1 | Create a working directory | 4 |
| 4.2 | Create a simple model | 4 |
| 4.3 | Build and run | 4 |
| 4.4 | Manage buffers | 4 |
| 4.5 | Debug | 4 |
| 5 | Interface | 5 |
| 5.1 | The basics | 5 |
| 5.2 | Menu | 6 |
| 5.3 | Keybindings | 7 |
| 6 | Configuration | 9 |
| 6.1 | Personal <code>.emacs</code> file | 9 |
| 6.2 | Canned TMB-IDE | 9 |
| 7 | References | 10 |

1 Preamble

1.1 Credit

The author of Template Model Builder (TMB) is Kasper Kristensen at DTU Aqua, Denmark.

Casper Berg (DTU Aqua, Denmark), Jan Jaap Poos (IMARES, Netherlands), and Kasper Kristensen provided useful feedback on the underlying `tmb-mode` that improved the functionality and workflow in TMB-IDE.

1.2 A simplified Emacs

The purpose of TMB-IDE is to make the convenient features of Emacs `tmb-mode` available to non-Emacs users. TMB-IDE can be defined as “starting Emacs in `tmb-mode` with a special `.emacs` file that disables the default Emacs keybindings”.

Experienced Emacs users may prefer to ignore the TMB-IDE `.emacs` file, and simply install and load `tmb.el` like other Emacs packages. It is a standard “major mode” that follows all Emacs mode conventions.

1.3 System requirements

TMB-IDE has been tested in Linux and Windows. It should work in other operating systems, perhaps requiring some modifications by the user.

What TMB-IDE does is to bind together several software components into an easy and efficient user interface for TMB. These components need to be already installed on the computer:

- R with package TMB
- Emacs with packages `tmb-mode` and `ess`
- GCC with `g++` compiler
- GDB (optional)

2 Installation

2.1 Configuration file

To install TMB-IDE, simply place the `.emacs` file into the `HOME` directory, or use some other way of loading it into Emacs during startup. In Linux, the `HOME` directory is `~` but in Windows it may or may not be defined. To investigate, open a shell and type:

```
echo %HOME%
```

If this command returns an existing directory, place the `.emacs` file there, otherwise place it in the `C:/` root directory.

2.2 Shortcut

To make it easy to start TMB-IDE, consider creating a shortcut of some kind to start Emacs: a desktop icon, start menu entry, shell alias, and/or a keybinding.

3 Introduction

3.1 Emacs tmb-mode

The process of creating statistical models with Template Model Builder (TMB) involves writing, compiling, and testing. An integrated development environment (IDE) allows the user to perform these tasks efficiently.

GNU Emacs is a complex and powerful editor that comes with particularly good support for C++, R, \LaTeX , backup/revision control, and other useful tools for statistical computing. Its `tmb-mode` provides syntax highlighting, compilation, file manipulation, templates, and smaller tools for creating TMB models. Emacs users can download `tmb.el` from the web and start using `tmb-mode` 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 advanced 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.” 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 TMB.

The rest of this tutorial demonstrates how Emacs with `tmb-mode` can be configured as a user-friendly TMB-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.

The screenshot shows the Emacs editor interface with a menu bar (File, Edit, Options, Buffers, Tools, TMB, C++, Help) and a toolbar. The left pane displays the source code of `mini.cpp`, which includes a template for an objective function. The right pane shows the compilation output, indicating an error: 'Sigma' was not declared in this scope. The status bar at the bottom shows the current file is `mini.cpp` at line 11, column 28, and the compilation exited with code 1.

```
File Edit Options Buffers Tools TMB C++ Help
mini.cpp
#include <TMB.hpp>

template<class Type>
Type objective_function<Type>
{
  DATA_VECTOR(x);
  PARAMETER(mu);
  PARAMETER(logSigma);

  Type f;
  f = -sum(dnorm(x, mu, exp
return f;
}

-.*. mode: compilation; default-directory: "~/". -.*.
Compilation started at Wed Oct 28 17:35:26

R --quiet --vanilla -e "require(TMB); compile('mini.cpp','-fno-gnu-unique ->
> require(TMB); compile('mini.cpp','-fno-gnu-unique -O0 -Wall')
Loading required package: TMB
g++ -m64 -I/usr/include/R -DNDEBUG -I/home/arnima/r/library/TMB/include -I/
mini.cpp: In member function 'Type objective_function<Type>::operator()()':
mini.cpp:11:29: error: 'Sigma' was not declared in this scope
    f = -sum(dnorm(x, mu, exp(Sigma), true));
                           ^
In file included from /home/arnima/r/library/TMB/include/TMB.hpp:60:0,
                 from mini.cpp:1:
/home/arnima/r/library/TMB/include/tmb_core.hpp:1416:16: __required from he
mini.cpp:8:13: warning: unused variable 'logSigma' [-Wunused-variable]
    PARAMETER(logSigma);
    ^
/home/arnima/r/library/TMB/include/tmb_core.hpp:182:30: note: in definition
#define PARAMETER(name) Type name(objective_function::
^
make: *** [mini.o] Error 1
Error in compile("mini.cpp", "-fno-gnu-unique -O0 -Wall") :
  Compilation failed
Execution halted

Compilation exited abnormally with code 1 at Wed Oct 28 17:35:28

-:--- mini.cpp All (11,28) (TMB) U:~* *compilation* All (9,0) (Compilation:exit [1])
```

4 Tutorial

4.1 Create a working directory

Open TMB-IDE and create a working directory called `c:/demo`:

```
M-n
c:/demo
C-d
c:/demo
```

The `M-n` keystroke means hold `Alt` and press `n`. This creates a new directory.

The `C-d` keystroke means hold `Ctrl` and press `d`. This sets the current directory.

See [\[Keybindings\]](#), [page 7](#), for a full list of TMB-IDE keybindings.

4.2 Create a simple model

TMB-IDE has a built-in example model, which can be used as a minimal template to create a new model. To create a mini template in the current directory, find the menu above the editor window and select *TMB* → *Mini Template* or press `f12`.

The screen will now split in two parts, which is the default state of a TMB-IDE session. The TMB model code (C++) is in the main window and the compilation script (R) in a secondary window. The cursor remains in the TMB model window, which makes TMB-specific commands available via the menu and keybindings.

4.3 Build and run

The compilation script contains R code that compiles and runs the model. There are two alternative ways to build a TMB model:

- *TMB* → *Build* or `f8`: only build the model
- *TMB* → *Run* or `f9`: run R script, typically builds (if not already built) and runs the model

4.4 Manage buffers

The secondary window is now showing either compilation output or an active R session, but the R script is no longer visible. To switch between the R script, compilation output, and the R session, select *View Script* (`f11`), *View Compilation*, or *View R Session* in the TMB menu.

Note how the *TMB* menu is only available when the active window is in `tmb-mode`. Press `f2` at any point to switch a window to `tmb-mode`.

This is a good time to learn basic buffer and window management in general. In Emacs, a buffer is like a page, often representing a file, but sometimes other things, like compilation and command output. 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 buffer-related [\[Keybindings\]](#), [page 7](#).

A window can be split in two by selecting *New Window Below* (`C-x 2`) or *New Window on Right* (`C-x 3`) in the *File* menu. The `escape` key lets one window fill the Emacs screen.

4.5 Debug

To debug a model in TMB-IDE, select *TMB* → *Debug* or press `f10`. This starts an R session that invokes the GNU Debugger (GDB). To finish the debugging session, type `quit` in the (gdb) prompt.

5 Interface

5.1 The basics

Files

The keystrokes **C-n**, **C-o**, **C-s**, and **C-w** can be used to create, open, save, and close files. For convenience, **C-p** opens any file that has a similar name as the TMB model, for example, **C-p dat** to open `mymodel.dat`. Directories can be created with **M-n** and switched to with **C-d**. To quit TMB-IDE, press **C-q**.

Cut, copy, paste (CUA mode)

These three commands are essential for editing any text, and they form a major obstacle for users who try Emacs for the first time.

In almost all other editors, **C-x**, **C-c**, and **C-v** are used to cut, copy, and paste. These are the keybindings that the majority of computer users have memorized and use in a wide variety of applications. The default keybindings in Emacs to cut, copy, and paste (**C-w**, **M-w**, **C-y**) are neither intuitive nor easy to memorize at first. Rebinding keystrokes is easy enough in Emacs, except **C-x** and **C-c** are reserved keystrokes in Emacs, to access many of the most important commands. Rebinding those keystrokes is questionable, in the same way it would be questionable for an R package to redefine things like `q()` or `pi`. There is no perfect solution, but the alternatives include:

1. Bite the bullet and learn the Emacs defaults.
2. Define keybindings to cut/copy/paste that feel convenient, but are not **C-x** or **C-c**.
3. Load special `cua-mode` that overrides the reserved defaults and binds **C-x**, **C-c**, and **C-v** to cut, copy, and paste.

TMB-IDE takes the third option: CUA mode. Among the drawbacks is that text is not reliably copied between applications, not even between Emacs instances. Workarounds include:

- Copy with **C-insert** and paste with **M-insert**.
- Copy with left mouse button, paste with middle button.

Undo/redo

The undo command in TMB-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.

Comment/uncomment

The **M-;** key comments or uncomments the highlighted code region, depending on whether the region is already commented or not.

Secondary window

The default state of a TMB-IDE session is with the screen split in two parts, with the TMB model code (C++) in the main window and something else in a secondary window. This secondary window can be navigated without leaving the main window, using intuitive keybindings: **M-up**, **M-down**, **M-pgup**, **M-pgdn**, **M-home**, **M-end**.

Compilation errors

If the secondary window contains compilation output, **M-up** and **M-down** will navigate between error messages instead of lines.

Help system

Besides the main TMB-IDE help page (**f1**), the Emacs help system has a help page for every function and variable, as well as keystrokes:

C-h k Keystroke
C-h f Function
C-h a Search for command
C-h v Variable

5.2 Menu

| Menu label | Purpose | Emacs command |
|---------------------|-------------------------|-----------------------------------|
| View Script | Show R script | <code>tmb-open</code> |
| View Compilation | Show compilation output | <code>tmb-show-compilation</code> |
| View R Session | Show R sesssion | <code>tmb-show-r</code> |
| Compile | Build model | <code>tmb-compile</code> |
| Run | Run R script | <code>tmb-run</code> |
| Make | Run makefile | <code>tmb-make</code> |
| Stop | Stop current process | <code>tmb-kill-process</code> |
| Clean | Remove *.o, *.so, *.dll | <code>tmb-clean</code> |
| Debug | Debug model with GDB | <code>tmb-debug</code> |
| Toggle NaN Debug | Toggle NaN exceptions | <code>tmb-toggle-nan-debug</code> |
| Mini Template | Create minimal template | <code>tmb-template-mini</code> |
| Multi-Window Layout | Arrange three windows | <code>tmb-multi-window</code> |
| Help | Show help page | <code>tmb-help</code> |
| TMB Mode Version | Show TMB Mode version | <code>tmb-mode-version</code> |

5.3 Keybindings

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

| Keystroke | Purpose | Emacs command |
|-----------|----------------------------|------------------------------|
| f1 | Help | tmb-help |
| S-f1 | Show TMB-IDE version | tmb-ide-version |
| f2 | TMB mode | tmb-mode |
| f3 | R mode | R-mode |
| f4 | Data mode | conf-unix-mode |
| C-f4 | Close | kill-this-buffer |
| M-f4 | Quit | save-buffers-kill-emacs |
| f5 | Reload | revert-buffer |
| f6 | Other window | other-window |
| C-f6 | Next buffer | next-buffer |
| f7 | Clean | tmb-clean |
| f8 | Compile | tmb-compile |
| f9 | Run | tmb-run |
| f10 | Debug | tmb-debug |
| f11 | View script | tmb-open |
| f12 | Mini template | tmb-mini-template |
| C-, | Toggle trailing whitespace | toggle-trailing-whitespace |
| C-. | Toggle function indicator | tmb-toggle-show-function |
| C-a | Select all | mark-whole-buffer |
| C-b | Next buffer | next-buffer |
| C-c | Copy | cua--prefix-override-handler |
| C-d | Change current directory | cd |
| C-f | Find, find next | isearch-forward |
| C-g | Goto line | goto-line |
| C-h | Emacs help system | help |
| C-l | Recenter | recenter-top-bottom |
| C-n | New | new-buffer |
| C-o | Open | find-file |
| C-p | Open in other window | tmb-open-any |
| 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-this-buffer |
| C-x | Cut | cua--prefix-override-handler |
| C-z | Undo/redo | undo |
| M-, | Delete trailing whitespace | delete-trailing-spc-tab-m |
| M-; | Comment/uncomment region | comment-dwim |
| M-n | Create new directory | make-directory |
| M-o | Other window | other-window |

| | | |
|------------------------|--------------------------------|---|
| <code>escape</code> | Cancel dialog, maximize window | <code>keyboard-escape-quit</code> |
| <code>tab</code> | Indent line or region | <code>c-indent-line-or-region</code> |
| <code>C-return</code> | Rectangle functions | <code>cua-set-rectangle-mark</code> |
| <code>C-space</code> | Expand recognized words | <code>dabbrev-expand</code> |
| <code>C-M-space</code> | Open recent files | <code>recentf-open-files</code> |
| <code>C-x 2</code> | Split window below | <code>split-window-below</code> |
| <code>C-x 3</code> | Split window right | <code>split-window-right</code> |
| <code>M-up</code> | 2nd window up, previous error | <code>tmb-scroll-up</code> |
| <code>M-down</code> | 2nd window down, next error | <code>tmb-scroll-down</code> |
| <code>M-pgup</code> | 2nd window page up | <code>scroll-other-window-down</code> |
| <code>M-pgdn</code> | 2nd window page down | <code>scroll-other-window</code> |
| <code>M-home</code> | 2nd window home | <code>beginning-of-buffer-other-window</code> |
| <code>M-end</code> | 2nd window end | <code>end-of-buffer-other-window</code> |

| Mouse button | Purpose | Emacs command |
|---------------------|---------------------|--------------------------------|
| <code>C-left</code> | Switch buffers | <code>mouse-buffer-menu</code> |
| <code>right</code> | Navigate with imenu | <code>imenu</code> |

6 Configuration

6.1 Personal .emacs file

TMB-IDE is intended for people who don't know Emacs, are not interested in learning it, and will only use it to work with TMB. The design goal is that TMB-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 TMB-IDE, but Emacs with `tmb-mode` and a personal .emacs file. One reason to modify the .emacs file or write a new one from scratch is to install additional Emacs packages. Another reason is to redefine the keybindings, probably closer to the Emacs defaults. Other reasons include setting fonts and colors, setting user variables, or defining new user functions. Users with a personal .emacs file can update TMB, Emacs, GCC, GDB, and `tmb-mode` independently.

Note that it is not advisable to configure Emacs by clicking *Options* → *Save Options* or *Options* → *Customize Emacs*. Editing the .emacs file directly is a more reliable and transparent approach.

6.2 Canned TMB-IDE

It can be practical to make a canned version of TMB-IDE available, while using a different .emacs file for most Emacs sessions. For example, an experienced Emacs user may want to test how TMB-IDE works, or demonstrate it to colleagues, without shuffling .emacs files. One way to do this is to rename the TMB-IDE .emacs to something like `tmb-ide.el` and then start TMB-IDE with the command

```
emacs -Q -l ~/tmb-ide.el -f tmb-mode
```

in Linux, or

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

in Windows.

The `-Q` option tells Emacs to ignore the default startup file(s), the `-l` tells it to load a Lisp file, and the `-f` tells it to call a function. This command can be assigned to a desktop icon, start menu entry, or shell alias.

7 References

TMB

Kristensen, K., A. Nielsen, C.W. Berg, H. Skaug, and B. Bell. 2015. *TMB: Automatic differentiation and Laplace approximation*. arXiv submission 1209.5145v1.

<http://arxiv.org/pdf/1509.00660>

Kristensen, K. with contributions by B. Bell, H. Skaug, A. Magnusson, C. Berg, A. Nielsen, M. Maechler, T. Michelot, and M. Brooks. 2015. *Template Model Builder: A general random effect tool inspired by ADMB*. R package version 1.6.2.

<http://cran.r-project.org/package=TMB>

Magnusson, A. 2015. *Template Model Builder IDE: Emacs tmb-mode without the Emacs*.

<http://www.hafro.is/~arnima/tmb.html>

Magnusson, A. 2015. *Virtual TMB: Template Model Builder in a box*.

<http://www.hafro.is/~arnima/tmb.html>

TMB Wiki. 2015. *Welcome to the TMB wiki*.

<https://github.com/kaskr/adcomp/wiki>

Emacs

Stallman, R. 2015. *GNU Emacs manual*. 17th ed. Updated for Emacs version 24.5.

<http://www.gnu.org/software/emacs/manual/emacs.html>

Lewis, B., D. LaLiberte, R. Stallman, and the GNU Manual Group. 2015. *GNU Emacs Lisp reference manual for Emacs version 24.5*. Rev. 3.1.

<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/manual/eintr.html>

Rossini, A.J., R.M. Heiberger, K. Hornik, M. Maechler, R.A. Sparapani, S.J. Eglen, S.P. Luque, H. Redestig, and V. Spinu. 2015. *ESS: Emacs speaks statistics*.

<http://ess.r-project.org>

GCC

Stallman, R.M. and GCC Developer Community. 2013. *Using the GNU Compiler Collection*. Version 4.8.3.

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

GDB

Stallman, R., R. Pesch, S. Shebs, et al. 2013. *Debugging with GDB: The GNU source-level debugger*. 10th ed. for GDB version 7.6.1.

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

R

R Core Team. 2015. *R: A language and environment for statistical computing*.

<https://cran.r-project.org/manuals.html>