ADMB and TMB codebase: Composition and history

Arni Magnusson

16 August 2017

ADMB

Timeline of ADMB codebase: year, version, MB and lines of code by file type.

		MB	cpp	lex	sh	mak	tpl	dat	R	conf	tex	html
2009	9.1	16	175	9	2	7	6	46	1	0	25	6
2011	10.0	27	206	9	1	2	5	47	1	0	21	6
	10.1	27	207	9	1	2	5	47	1	0	20	6
2012	11.0	22	220	11	2	2	7	48	1	0	30	6
2013	11.1	44	238	11	2	2	27	49	1	0	30	6
2014	11.2	17	238	16	2	3	13	50	1	0	30	5
2015	11.3	18	238	16	2	3	13	50	1	0	30	5
	11.4	17	238	16	2	3	13	50	1	0	30	5
	11.5	18	239	16	2	3	14	50	1	0	30	5
2016	11.6	18	251	16	2	3	14	52	1	1	21	5
2017	dev	18	262	16	2	3	14	52	1	1	21	5

MB: total repo size (download and unzip, excl. history)
cpp: C++ source and header files (thousand lines of code)
lex: Flex files
tpl: ADMB models
sh: shell scripts
mak: makefiles
R: R scripts
dat: data files
conf: editor settings
tex: manuals
html: example documentation
dev: 1c698c8 (2017-08-14)

Comments

The ADMB codebase takes 18 MB of storage space and consists of 284 thousand lines of code (262 C++, 16 Flex, 3 makefiles, 2 shell scripts).

Since 2009, the overall storage space has increased by 12% (1.4% per year), but the lines of code have increased by 47% (4.9% per year).

Caveats: The C++ files contain documentation (Doxygen) in many cases. Large intermediate Flex output files were introduced in 11.1 and removed thereafter. A somewhat large texinfo.tex system file was introduced in version 11.0 and removed in 11.6.

TMB

Timeline of TMB codebase: year, version, MB and lines of code by file type.

		MB	cpp	R	dat	conf	dox
2013	beta	5.0	140	2	0	0	0
2014	1.0.0	5.9	142	5	4	0	0
	1.0.1	6.0	142	5	4	0	0
	1.1.0	6.2	150	5	4	0	0
	1.2.0	6.2	150	5	4	0	0
	1.3.0	6.2	150	5	4	0	0
	1.4.0	6.3	153	5	4	0	0
	1.5.0	6.3	153	6	4	0	0
	1.5.1	6.4	154	6	4	0	0
	1.6.0	6.4	154	6	4	0	0
2015	1.6.2	6.4	154	6	4	0	0
	1.6.3	6.4	154	7	4	1	0
	1.6.4	6.4	154	7	4	1	0
	1.6.5	6.4	154	7	4	1	0
2016	1.6.6	6.5	154	7	4	1	0
	1.7.0	6.5	154	8	4	1	0
	1.7.1	6.5	154	8	4	1	0
	1.7.2	6.5	154	8	4	1	0
	1.7.3	6.8	164	8	4	1	0
	1.7.4	7.0	164	8	4	1	0
2017	1.7.6	7.2	165	8	4	1	4
	1.7.7	8.4	195	8	4	1	4
	1.7.8	8.4	196	8	4	1	4
	1.7.10	8.4	196	8	4	1	4
	1.7.11	7.8	195	8	1	1	4
	dev	7.9	195	8	1	1	4

MB: total repo size (download and unzip, excl. history)
cpp: C++ source and header files (thousand lines of code)
R: R scripts
dat: data files
conf: editor settings
dox: book
dev: 69ff3cc (2017-08-14)
beta: 24fed26 (2013-09-17)

Comments

The TMB codebase takes 8 MB of storage space and consists of 204 thousand lines of code (196 C++, 8 R).

Since 2013, the overall storage space has increased by 58% (12.1% per year), but the lines of code have increased by 43% (9.3% per year).

Caveats: Both C++ and R files contain documentation (Doxygen, Roxygen). A large part of C++ files are included from Eigen and CppAD. A large part of R files contain data values.

Discussion

Both ADMB and TMB are being released and improved at a healthy rate, the lines of code increasing by 5% and 9% per year, respectively. Documentation appears to be increasing at a slightly faster rate.

ADMB and TMB come with a large suite of examples, serving as documentation and as part of continuous integration tests. Both are documented using Doxygen/Roxygen markup within the source code.

The ADMB codebase is considerably larger than the TMB codebase, with 2 times more storage space and 40% more lines of code. The TMB source code is roughly 96% C++ and 4% R.

The timeline of "repometrics" presented here are noisy data, but nonetheless useful as an overview of the maintenance and development of the two software projects.

Appendix

ext.sh

Script to get an overview of filetypes

```
find | sed 's/.*\(\.[A-Za-z0-9]*$\)/\1/g' | grep -v / | sort | uniq > ext.txt
```

harvest-admb.sh

Script to aggregate file types and count lines

```
rm all.*
# /dev/null to handle zero-match cases
# when aggregating multiple file extensions, start with EXT > all.EXT

# C++
cat `find -type f -name '*.cpp'` /dev/null >  all.cpp
cat `find -type f -name '*.hpp'` /dev/null >> all.cpp
cat `find -type f -name '*.h'`   /dev/null >> all.cpp

# Flex
cat `find -type f -name '*.lex'` /dev/null >  all.lex

# Shell scripts
cat `find -type f -name '*.sh'`  /dev/null >  all.sh
cat `find -type f -name '*.bat'` /dev/null >> all.sh

# Makefiles
cat `find -type f -name '*.mak'`    /dev/null >  all.mak
cat `find -type f -name 'Makefile'` /dev/null >> all.mak

# R scripts
cat `find -type f -name '*.R'` /dev/null >  all.R
cat `find -type f -name '*.r'` /dev/null >> all.R
cat `find -type f -name '*.s'` /dev/null >> all.R

# ADMB models
cat `find -type f -name '*.tpl'` /dev/null >  all.tpl

# Data
cat `find -type f -name '*.dat'` /dev/null >  all.dat
cat `find -type f -name '*.DAT'` /dev/null >> all.dat

# Editor settings
cat `find -type f -name '*.el'`      /dev/null >  all.conf
cat `find -type f -name '*.vimconf'` /dev/null >> all.conf

# Manual
cat `find -type f -name '*.tex'` /dev/null >  all.tex
cat `find -type f -name '*.bib'` /dev/null >> all.tex

# Webpages
cat `find -type f -name '*.html'` /dev/null >  all.html
cat `find -type f -name '*.css'`  /dev/null >> all.html

# Count lines
wc -l all.*
```

harvest-tmb.sh

Script to aggregate file types and count lines

```
rm all.*
# /dev/null to handle zero-match cases
# when aggregating multiple file extensions, start with EXT > all.EXT

# C++
cat `find -type f -name '*.cpp'` /dev/null >  all.cpp
cat `find -type f -name '*.hpp'` /dev/null >> all.cpp
cat `find -type f -name '*.c'`   /dev/null >> all.cpp
cat `find -type f -name '*.h'`   /dev/null >> all.cpp

# R scripts
cat `find -type f -name '*.R'` /dev/null > all.R

# Makefiles
cat `find -type f -name 'Makefile'` /dev/null > all.mak

# Shell scripts
cat `find -type f -name '*.sh'` /dev/null > all.sh

# Manual
cat `find -type f -name '*.dox'` /dev/null >  all.dox
cat `find -type f -name '*.bib'` /dev/null >> all.dox
cat `find -type f -name '*.cls'` /dev/null >> all.dox
cat `find -type f -name '*.md'`  /dev/null >> all.dox
cat `find -type f -name '*.org'` /dev/null >> all.dox
cat `find -type f -name '*.Rmd'` /dev/null >> all.dox

# Data
cat `find -type f -name '*.dat'` /dev/null > all.dat

# Editor settings
cat `find -type f -name '*.el'` /dev/null > all.conf

# Count lines
wc -l all.*
```