ADMB Developers Workshop Report

June 4th - 7th 2013, *9 am - 5 pm*

University of Washington
*SAFS Building room 1*08
1122 NE Boat St, Seattle, WA

Contents

Introduction

This report summarises the events of the fourth ADMB developers workshop. As with the previous three workshops, this meeting was held in an informal style, allowing developers and interested users of the ADMB software to discuss relevant issues, immediate priorities, and long-term goals of the ADMB-project. The informal nature of the meeting allowed considerable time to be devoted to solving relevant ADMB source code problems and for the sharing of tips, tricks, and general knowledge related to coding, software development, and model building.

The meeting convened in Seattle on Tuesday June 4th and adopted the Agenda attached below. Jim Ianelli and Athol Whitten were appointed as rapporteurs. Participants from NOAA, SAFS at UW, DFO, IPHC and other institutions from both the USA and Canada joined the meeting for some or all of the sessions held over four days. The list of participants is attached. There was a formal ADMB Foundation meeting held on June 6, 2013. The report for that meeting is a separate document to this one.

Summary of discussions

The following topics were recurring themes and discussed at many times during the four day workshop.

ADMB Project status and overview

Johnoel provided an overview of the project hosts server and infrastructure. Namely the following description of the available services:

Web Server

- Open source Plone content management system and Apache web server software are used for the main web server.
- The URL is http://www.admb-project.org
- Website was recently reorganized by Juan Valero and minor modifications by other to improve accessibility.
- Currently 41 users have edit permissions to contribute to web development.

Version Control

- Open source Apache Subversion software is used for source code version control.
- Public read-only web access is available.
- The URL is http://www.admb-project.org/svn/
- Revisions presently exceed 1000 commits from 18 developers.
- Currently 20 developers have write permissions to the source code.

Issue Tracker

- Open source Redmine server is used to track issues.
- The URL is http://www.admb-project.org/redmine/projects/issues/

Automated Build Server

- Open source Buildbot server is used for automated builds and testing.
- The URL is http://www.admb-project.org/buildbot/waterfall
- Windows, Linux and MacOS build systems are hosted at PFRP@University of Hawaii.
- Automated systems use different combinations of C++ compilers and computing platforms such as Windows, Linux and MacOS to build and test the software.

Searchable Mailing Lists

- Open source Listserv server is used for ADMB mailing lists.
- The primary lists are developers@admb-project.org, users@admb-project.org and announce@admb-project.org. They are available online and searchable.
- The users list URL is http://lists.admb-project.org/pipermail/users/
  The developers list URL is http://lists.admb-project.org/pipermail/developers/
  The list currently has 174 subscribers.

The ADMB project and direction

The following discussions helped guide priority issues detailed in a summary section below.

A 64-bit compatible version of gcc is available but Chris Grandin says he has tested it and for whatever reason, it will not work when attempting to compile admb from source. This might be able to work with a cross-compiled version of the library? Matthew suggests using Clang as an alternative .cpp compiler would solve this problem.

The issue of accessing the gradient and derivative value information generated by ADMB during a model run was discussed. Such information is normally displayed at runtime during the execution of an ADMB model: but users might benefit from being able to access this information at a later time. Therefore, the issue of how best to save the these values to some file was discussed. The phase, function evaluation, parameter number, value, gradient. This was the approach suggested to resolve issue number 109.

The file newfmin.cpp contains the latest version of the function minimizer for the ADMB code (as exists to the foundation). Alternatives have been developed and there is more than one function minimiser available in ADMB: an option to choose among minimizers is available at the command line. The default option ‘newfmin’ is a quasi-newton-like method, and apparently, there are other gradient and non-gradient methods available (e.g., Nelder Mead simplex algorithm is also available “-neldmead”).

It was suggested that an entity relationship diagram (or UML diagram) would be a good addition to the documentation (perhaps an advance, or developers recommended, version of the documentation). Such a diagram would enable other developers to better understand the source code and therefore further develop the code/software. It was noted (and demonstrated) that a call-graph can be created for the ADMB source code using the Doxygen documentation system. The resulting call graph for ADMB seemed too large and unwieldy however to be of much use; however, the non-graphical class hierarchy is of some utility for understanding class inheritance. An alternative to these call graphs is provided automatically by

Doxygen. The detailed documentation for function contains a "Referenced by" paragraph containing a hyper-referenced list of other functions that call the function being described.

Regarding the double_and_int class, it was noted that the memory definition of this class could be changed so that at each point, both the derivative and value of a function are part of the active memory allocation, rather than having the values overwritten with the derivatives, as is the case at the moment.

In ADMB when estimating parameters that have user-specified bounds the transformation of each parameter (as part of the total minimisation problem) may cause issues if they are scaled differently. For example, if mapping an active parameter to a model parameter with different bounds it can create a more complex likelihood 'surface', which in-turn may make it harder for the minimiser to find the minimum (i.e., it is useful to have the derivatives scaled so they are independent of the bounds). Using a rescaling option may improve the situation and provide simpler surface over which to run the minimisation problem. The issue of whether the –rs option (at commandline) helps this problem was unclear.

Debugging and code testing

Johnoel Ancheta provided an overview of ongoing developments for the ADMB project: Including the Redmine issue tracker and the Buildbot service which automatically tests well-understood ADMB code on each new iteration of ADMB software across multiple operating systems/platforms. He also demonstrated ways to conduct profiling and debugging of code using Valgrind. Valgrind can automatically detect memory management issues (memory leak detection [mld]) and threading bugs, and perform detailed profiling to help speed up programs. Johnoel demonstrated the use of Valgrind for debugging, memory leak detection, and profiling during the workshop and also created an online Valgrind tutorial that has been added to the ADMB website.

Note: Valgrind is available for Linux and Mac OSX, but not for Windows. Unfortunately there is nothing compatible available for Windows but there are several projects underway to create a version of Valgrind (or something equivalent) for the Windows OS. Valgrind seems to be the only currently available open-source software that can effectively perform both mld and profiling. There are commercial tools such as Windows Visual Studio and MacOS XCode.

Dave Fournier demonstrated use of the GNU Debugger [gdb] at the terminal (or command prompt), in a manner which allowed for any object which had been created in an ADMB program (declared in a .tpl file) to be called live at the terminal (in an active script). Dave described how such functionality can be added to ADMB: He has written some 'print' functions for his own use that can be used to view the contents of regularly defined objects such as vectors, matrices, or arrays. This allows a user to 'see' their internal model structures and therefore check that calculations are as expected. Such convenience (as is available in active scripting languages such as R) can greatly aid model development for the end user.

Dave and Chris discussed further developing print functions that can be used together with gdb to allow

easy display of values for complex model objects (e.g., high dimensional dvariables). It was suggested that such functionality would work best if a user was able to link the required functions as a library whenever -s (safe mode) was used when compiling a .tpl file. Chris Grandin displayed his approach to doing this (entirely with scripts within GDB) and thought that Dave's alternative approach was simpler and more likely to work on different platforms. Also, since it would be compiled and linked as a library it would be much faster than the GDB scripts. The group **added this task to the priority list for ADMB**.

Other tools such as DDD, Netbeans, MS Visual C++, and other IDEs (including the ADMB-IDE) for debugging were discussed. Minimally, the ADMB project considers supporting GDB on all platforms as a priority (and others are welcome to contribute to some platform-specific alternatives as they are able).

Overview of MCMC options and studies

A discussion of the MCMC options in ADMB was led by Cole Monnahan, Ian Taylor, and Jim Thorson. Together, they reviewed and presented their previous work on this topic and the ensuing conversations provided ideas for a number of areas of development.

Cole provided an expose on a suite of different options available for generating posterior multivariate distributions using MCMC options within ADMB. He provided the workshop with a document of these findings and agreed to add them into the main ADMB documentation (perhaps as an appendix).

Ian illustrated tests using the metropolis-hastings algorithm and hybrid MCMC option and how well it performed on a particularly pathological surface (especially compared with the standard MCMC options).

Jim Thorson discussed and idea that he and Hans Skaug were working on; specificially, the possibility of subsetting the Hessian matrix of a regular ADMB model, computing the determinant and using the Laplace approximation on those as a means to approximating random effects. He suggested the steps would be:

1. Propose theta (FE and RE) and tau (var(RE));
2. Estimate theta given tau;
3. Extract Hess(theta);
4. Calculate Laplace approx to the marginal likelihood (ML);
5. Repeat steps 1-4 to maximize ML given tau;
6. Run a final time given an estimated tau.

Dave suggested that all models can be specified in a state-space formulation (i.e., where integrating across states, not random interventions), and that specifying deterministic transitions is equivalent to eliminating some of the random effects, and instead efficiency may be gained by updating states using a deterministic function.

Jim Thorson clarified (somewhat) that the proper method for changing the jump function as determined from this discussion is to read in the Hessian, invert to the covariance, convert to the correlation and variances, and then convert the correlation to the cholesky decomposition. The correlations can then be thought of as a system of n(n-1)/2 linear equations of the elements in the cholesky, and specifying a fixed value for an element in the correlation matrix is equivalent to fitting a linear model to that system of linear equations given estimated quantities in the Cholesky matrix.

Dave discussed the use of copulas (used to describe the dependence between two random variables; a kind of distribution in probability theory): he advised that they help avoid the problem of some types of distributions not being additive. For example, when a number n, of distributions of type y, are added together, the resulting distribution N is not necessarily of type y. If the distribution type y is a copula, then the resulting additive distribution will also be of type y. This type of approach could be used in a multi-step process to first estimate the covariance then apply alternative (say fat-tailed or less correlated) multivariate distributions as jumping functions.

Adding MCEVAL_SECTION

The issue of how to introduce a new section (e.g., MCEVAL_SECTION) was examined and discussed. During the week Teresa worked on developing the framework for adding this option to address Bug #55, a feature suggested by Mark Maunder (http://admb-project.org/redmine/issues/55). As Mark Maunder could not recall why he made this request or its additional functionality for the user that would differ from using "if (mceval()) {...}", development was halted. The parsing of the new section was implemented in tpl2cpp, but further changes to other ADMB areas are still needed.

Multi-threading and parallelization

John Sibert provided a trivial example "msimple.tpl" to provide an illustration on how to use recently developed pthread manager class within ADMB. Regarding the "msimple.tpl" illustration, a break-out group worked through a way to further simplify the code. This group also established that pthreads was able to work in the MacOS and windows environment in addition to Linux. Steve Martell was able to use this example to apply it to a separate application.

The following steps were identified to add parallelization to the code:

1. Identify part of a model suitable for parallelization (e.g., an extensive likelihood calculation);
2. Create a new class and add the function (it should be a class member function);
3. Compile and get "...not defined" errors;
4. Get the list of missing variables and separate them constants and dvariables (they use different sends and gets in multi threadings);
5. For the constants determine where and how often they are defined in order to use "cwrite_buffer" command to exchange the constant objects;
6. For the dvariable objects, put them in the newly created class (declare them) so that they will exist but be unallocated;
7. Test the model by running it (it will crash) but then look for unallocated objects (using a debugger). If the unallocated objects are identified as being part of the input, create a list of write lock buffer. If part of the output, look at the constructor, see how it's built to determine the correct shape;
8. Continue step 7 until the model runs.

Participants worked on examples on different platforms in order to test these developments (using the "simple model" for pthreading).

Sibert reported preliminary timing improvements achieved applying the phtread manager class to a model that estimates movement and mortality parameters in a 90x40 spatial grid. The time required for 100 function evaluations decreased from approximately 36 to 3.5 minutes. The model tracks 28 individual tag cohorts for 24 months and attempts to run each cohort on a separate thread. Fifty threads were created. Hardware: honmaguro workstation at JIMAR (purchased for ADMB project) with 12 dual core Intel Xeon CPU X5690 processors running at 3.47GHz. (John thinks these timings are overoptimistic, but they may be OK.)

Matthew noted that pthreads have options to execute threads via "round-robin" versus a "first-in and first out" policy for queueing them up. For windows machines the pthreads manager requires a pthread library (downloadable) and some minor edits to the makefile (linking pthreadwin32 library instead of the -pthreads compile option) to get it working for Windows.

Summary of concurrency discussions

The group noted that this approach appears to have promise for speeding up estimation and model execution substantially. Presently the steps to implementing multi-threading would be onerous to regular ADMB users. However, the group noted that the best way to make real progress is to continue to develop more examples along the lines of what was done at this workshop and as they are tested, find where commonalities exist.

Documentation Issues

Jon Brodziak expressed the importance of good documentation to ensure continuity and maintainability of

software projects. The ADMB foundation should consider which documentation types are most important and which should be completed with the highest priority. Currently, there is documentation available online in two formats: (1) a documentation API (application programming interface) built with the Doxygen documentation system, and (2) a set of PDF documents available for download from the website (and available in .tex source format with the source code distribution.

Jim Ianelli demonstrated the use of Doxygen to document ADMB source code and contributed library functions, and then how to commit that documentation to the ADMB documentation API using SVN. During the workshop, PDF documentation manuals were updated to the latest versions following the release of ADMB 11.1. This included updates for the ADMB, ADMB-RE, and AUTODIF manuals. The specialized dvariable constructor with kkludge_object argument is now fully documented (after extended discussions).

Website work

Juan Valero has worked during 2013 to improve the organisation and accessibility of ADMB websites. The ADMB Project website was reorganized, cleaned-up and updated. The number of pages available in the tabs menu was reduced to improve navigation by merging related pages. Additional examples to illustrate the ADMB capabilities were added to the website and allowed to fall under multiple categories. Examples from the ADMB and ADMB-re manual, the NCEAS project website and the otter-research website were moved to the ADMB project website. Users are allowed now to view the collection of all available examples and categorization was added to examples. A number of outdated and deprecated pages, such as those describing installs of deprecated compilers and ADMB versions, were removed or moved to storage. Material from past ADMB courses was collated and reorganized in the website to provide easy access to new users. Other changes included updating the list of ADMB related publications and reorganizing them by first author. When available, a short description of how ADMB benefited each of those reported projects has been added. Links to electronic copies of publications were also added when publicly available. A new interactive map was made to illustrate the worldwide location of institutions using ADMB along with places where ADMB courses and workshops have been held.

The ADMB Foundation website was upgraded to the latest WordPress version (3.5.1) from the previous version (1.5.1). The older version had a number of issues related to lack of functionality of outdated themes and plugins that were resolved by the upgrade. The website content was updated and fixes were made for broken links and various pieces of text. Deprecated content was eliminated, and pages categorized to improve navigation.

Jim Ianelli and Steve Martell suggested we adopt Markdown for writing some components of online documentation so that any edit we create in a readme file (a basic text file that is human-readable) can more easily be converted to PDF format and HTML format for the website. Use of Markdown might also permit examples and worked sections to be more easily and more frequently added to the website and associated documentation by avoiding the need for double handling and reproduction of text.

Chris Grandin noted that the most common piece of feedback he receives from the user base (or potential user base) is that the website is difficult to navigate and that the installation process if complicated and off-putting. Ian Taylor sketched out a simplified approach that should best guide the installation process.

Updated priorities for ADMB:

The group listed the technical development priorities and updated them from previous workshops:

1. Parallelization Internal ADMB part, and user control part, with a "howto"
   Work progressed prior to and during the Seattle 2013 workshop. A simple model was refined and was committed to the branch threaded2. Goal to merge current branch to the trunk so that those interested could test and aid in further development.
2. Hybrid MCMC Fully implement and document this capability
   Cole Monnahan along with Ian Taylor and Jim Thorson presented their work done on this in the past year. The idea of using multiple "heated" chains together with the multithreading was discussed. The group noted that some work coordinating documentation is needed.
3. Matrix library improvement Linking external (large) matrix library to improve random effects library
   This topic was discussed and Anders Nielsen together with Kasper Kristiansen will likely pursue/present further at the Icleandic workshop
4. MCMC streamlining Model conversion to constant objects
   Dave F. noted that this work would be relatively easy to implement but first should be tested by simply converting an existing model to use constant objects to test if the improvements are real (i.e., overhead maybe improved (speed-wise) with newer compilers)
5. Improve the ability to test/debug code
   A valgrind tutorial was made and posted on the website. Reviews of using GDB (plain and with DDD, in addition to others such as the netbeans GUI) was demonstrated.
6. Design matrices Add capabilities for creating design matrices
   Unsure if this has been added, Anders Nielsen had done some of this.
7. Data alternatives XML may be a useful approach to adding robustness between datafiles and DATA_SECTION, Netcdf capabilities?
   John Sibert reported on his draft XML library that could fit as an alternative run-time parameter dimensions, bound, phases etc. Progress has waned on building this to the extent needed to become part of the contributed library.
8. Shared library object
   To add flexibility ADMB was compiled as a shared object library in the standard distribution and also by Matthew Supernaw in CLang. Johnoel notes that this is complete (revision 1064) hence such libraries are available.
9. Flag/switch class Simplify the use of control and MCMC files, perhaps MFCL-like object capability
   This work is underway. [what priority should this be given?]

10. Code tuning, debugger Profiling tool to improve code, e.g., where adjoint code may best be applied. Gcc profiler, memory usage. Develop example howto
    See valgrind example.
11. Developers workshop To train more people on how to write and contribute code to Autodif/ADMB
    Remains a priority, limited progress at June 2013 workshop

It was noted that the ADMB should encourage student support (e.g., improving ADMB with whatever fisheries problems might arise). Trevor Branch (UW professor) might be keen to take on this type of student with one idea to make use of Dave's multithreading work.

Workshop agenda

June 4th-7th 2013, 9 am - 5 pm
University of Washington
*SAFS Building room 1*08
1122 NE Boat St

Topics and discussion items

1. Introductions, scheduling, Agenda additions — Maunder
    a. History of developers' workshops
    b. Goals of this workshop
    c. Work sessions and breakouts
    d. Next steps
2. ADMB Project update
    a. ADMB Project status within JIMAR (Univ. Hawaii) — Sibert
    b. Project Infrastructure — Ancheta
        i. Webserver
        ii. Version control
        iii. Issue tracker
        iv. Automated Build Server
    c. Releases — Ancheta
3. Website updates — Valero
4. ADMB capacity building courses — Various
    a. Locales
    b. Ways to improve
    c. Future plans?
5. Survey of ADMB model applications
    a. Fisheries
    b. Ecological — TBD
    c. Spatial models
    d. Other?
6. MCMC studies — Monnahan/Taylor
7. Publications
8. Breakout session topics
    a. Parallelization — Fournier
    b. Alternative tools tutorial — Supernaw
    c. Debugging approaches
    d. Wishlist
    e. Examples development
    f. 64 bit code and platform update
    g. OTHERS
9. Provision of improvements and documentation — Brodziak
10. Other business
11. Report writing — All

Goals

- Usage

- Concurrency- Multi threading
- Improving documentation
- Subversion
- Tracking issues
- Foundation website

Valgrind tool

- Explain usage
- Other tools - XCode profiler and Microsoft profiler

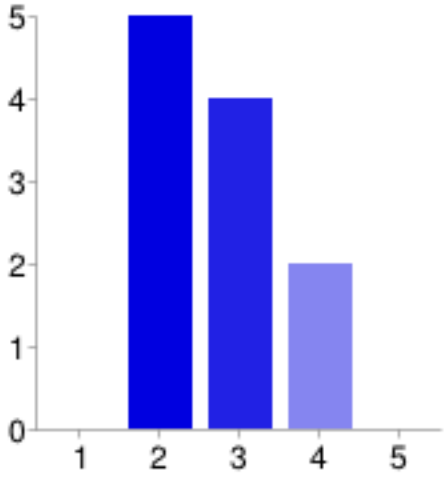
Participant list

Athol Whitten	whittena@uw.edu
Carey McGilliard	carey.mcgilliard@noaa.gov
Chris Grandin	chris.grandin@dfo-mpo.gc.ca
Cole Monnahan	monnahc@uw.edu
Cody Szuwalski	c.s.szuwalski@gmail.com
David Fournier	davef@otter-rsch.com
Ian Taylor	ian.taylor@noaa.gov
Jim Thorson	james.thorson@noaa.gov
Jim Ianelli	Jim.Ianelli@noaa.gov
Johnoel Ancheta	johnoel@hawaii.edu
John Sibert	sibert@hawaii.edu
Jon Brodziak	Jon.Brodziak@noaa.gov
Juan Valero	jvalero@iattc.org
Matthew Supernaw	matthew.supernaw@noaa.gov
Steve Martell	SteveM@iphc.int
Teresa A'mar	teresa.amar@noaa.gov

Results from post-workshop survey

Overall satisfaction of the workshop

(1: Very satisfied; 5: unsatisfied)



How could the workshop be better?

Next time, I hope we have scheduled break-out sessions, a fixed agenda, agreed upon before the

workshop, and a chance for all people (even the quiet ones) to be heard if they wish to speak about anything, I am more of a "newbie" rather than a developer, so I'm not sure how useful my comments are here. It seems like having a separate workshop for people who have not been developers, but are interested in learning about development could be useful if one goal is to get more people involved. I had a hard time following the jargon and what was happening due to my lack of development experience, though I learned some things and people were helpful when I asked questions. Organization and preparation Encourage more participation. More organization and direction and timing of topics and working groups Have a more set schedule and presenters ready before arriving. Have the first activity be determining the objectives of the workshop, the priorities of those objectives, who will be working on which ones, and what the milestones will be, after the usual agenda items have been addressed. The second activity should be reviewing all of the open items in the issues database to see if additional items should be added to the workshop "to-do/will-do" list. These activities should take place at a reasonably high level and not involve any code or methods discussion at this point. Talk with Matthew about transitioning to a more formal software development lifecycle process. And perhaps have someone volunteer to be "task master" to check on progress, perhaps before lunch each day...Attend all days! Assigned breakout groups. Not sure Organization and preparation Setting up objectives and work plan for each day

What parts of the workshop were successful?

Open discussions and example coding or problem solving on the projector seemed to work well. I only attended one day of the workshop...and the bbq. The bbq was very successful! Doing active computing activities was helpful as well. However, again, there were many parts of the workshop that were probably very useful that I either missed or didn't understand, so someone else could provide better comments! Barbecue Presenting pthreads, documentation issues, and MCMC methods. Pthreads progress is good, and discussion of documentation is headed in the right direction, i.e. unified non-redundant markdown files. The MCMC stuff is interesting to think about but there was no real resolution as to what will happen with it. Meeting people and putting faces to names. Group discussions on valgrind and specific sections of functionality (e.g., pthreads, MCMC). And, of course, the barbeque. It is always nice to hang out with you and your family, and your nephew makes very tasty food. Doxygen documentation (Jim's now doing it!). pthreads, I was able to get this working on an independent model. Happy to continue with testing/developing. The MCMC presentations Learning stuff from Dave, discussing things that matter in person rather than by email. Barbecue Hand's on demonstrations of tools

Comments? Suggestions? Feedback? Jokes?

A joke for Dave: Hand over the computer, friends don't let friends derive drunk. ;) Thanks for organizing and letting interested, but inexperienced people sit in and participate. As an outsider, it seems to me that there are some activities that need to be done that probably won't happen unless someone is hired specifically to do them, like providing documentation of various functions for the common user and more guidelines for troubleshooting - not fun, but would be very helpful to have and would make ADMB a lot more accessible to a wider audience. Try to prepare more in advance of the workshop with specific tasks by participants I couple of conference calls a year between developers may help unify everyone outside

these workshops. I was only able to attend parts so my opinion shouldn't count that much. Did Cole actually finish his presentation? I didn't recall seeing a summary of results and/or ideas in progress. Thanks for all of your hard work, Jim. Integrate the MCMC presentation stuff, where applicable with examples, into the ADMB.pdf documentation. Thanks Jim for putting in all the work to make this happen. There were a number of UW students present on the first day, but their numbers dropped by the second day. the agenda and having some approximate timing of activities/presentations could help new people attend areas of particular interest when they can't attend to all sessions due to other commitments.

The end, with thanks to Statler and Waldorf.