



AD Model Builder

Arni Magnusson

Vigo, 7 Apr 2010



AD Model Builder

Powerful program to fit models

Fast, reliable, flexible

Free software

Somewhat hard to learn

Number cruncher

Use alongside Excel, R, etc.

Fitting nonlinear models

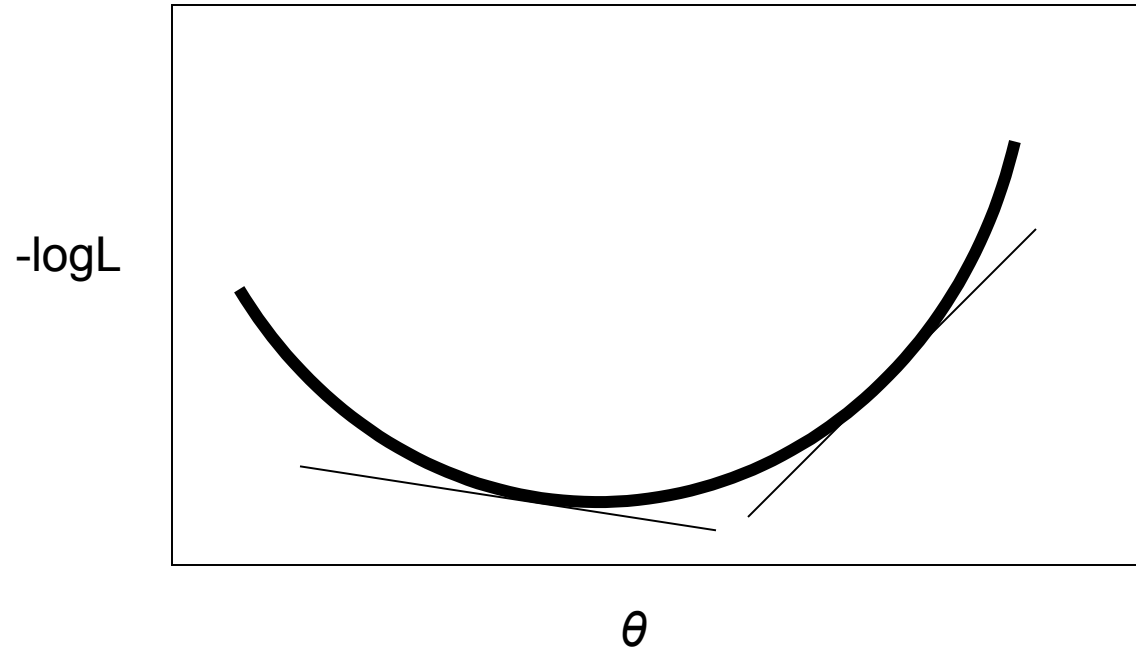
- Even simple models (<10 parameters) can have complex likelihood space
- Multiple minima, ridges and canyons
- Excel solver is easy to use, but cannot handle complex models or evaluate uncertainty
- R `optim()` evaluates uncertainty, but cannot handle complex models

Performance

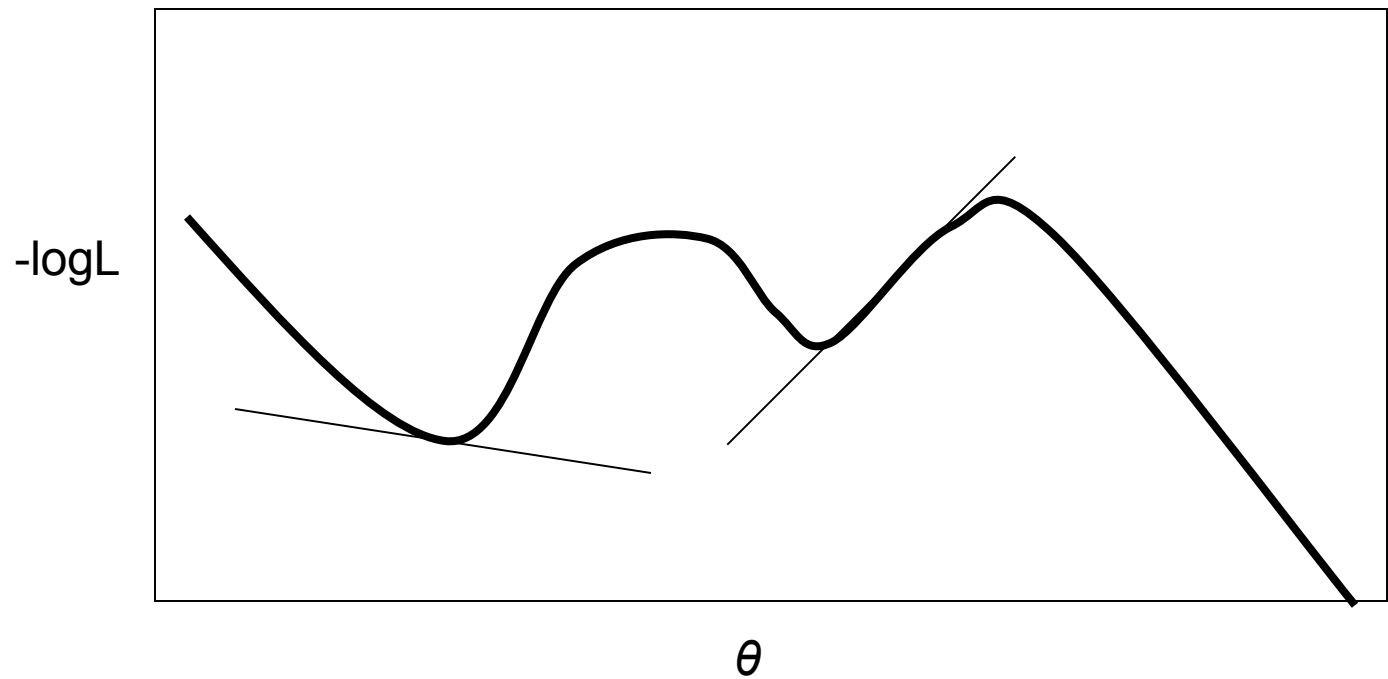
Modeling Package	msec/function call	number of function calls	time to converge
AD Model Builder	131	291	38 seconds
Gauss	167	23,365	1.08 hours
Matlab	639	18,360	3.25 hours
S-plus	n/a	n/a	n/a

Schnute et al. (1998)
fisheries stock assessment model

Derivatives

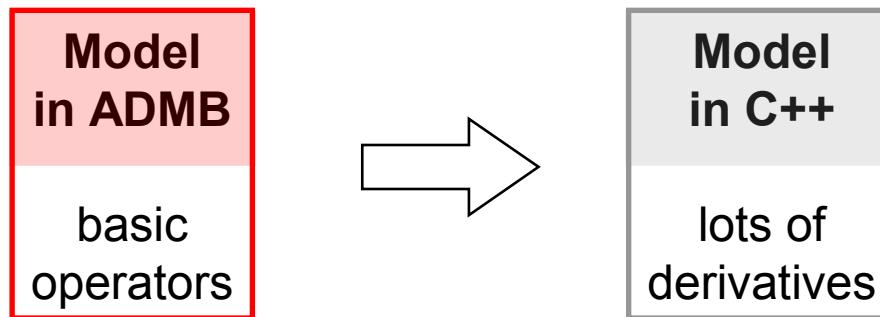


Derivatives



Automatic differentiation

- The "AD" in AD Model Builder
- Models and loops can be rolled out to basic operators like $+$ $-$ $*$ $/$ $^$ $\sqrt{}$ \log \exp
- All those operators are overloaded to perform differentiation (chain rule, etc.)



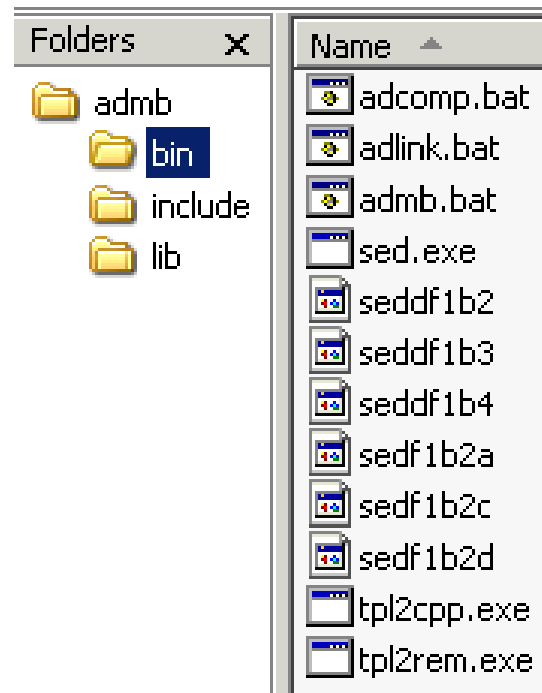
History

- 1990 MULTIFAN (Fournier et al.)
- 1991 AD book (Griewank and Corliss, eds.)
- 1993 ADMB 1.0
- 2007 ADMB 8.0.2
- 2007 ADMB Foundation (Sibert, Ancheta, and Maunder)
- 2008 ADMB 9.0 (freeware)
- 2009 ADMB 9.1 (free software)
- 2010 ADMB Meeting (short-term and long-term goals)

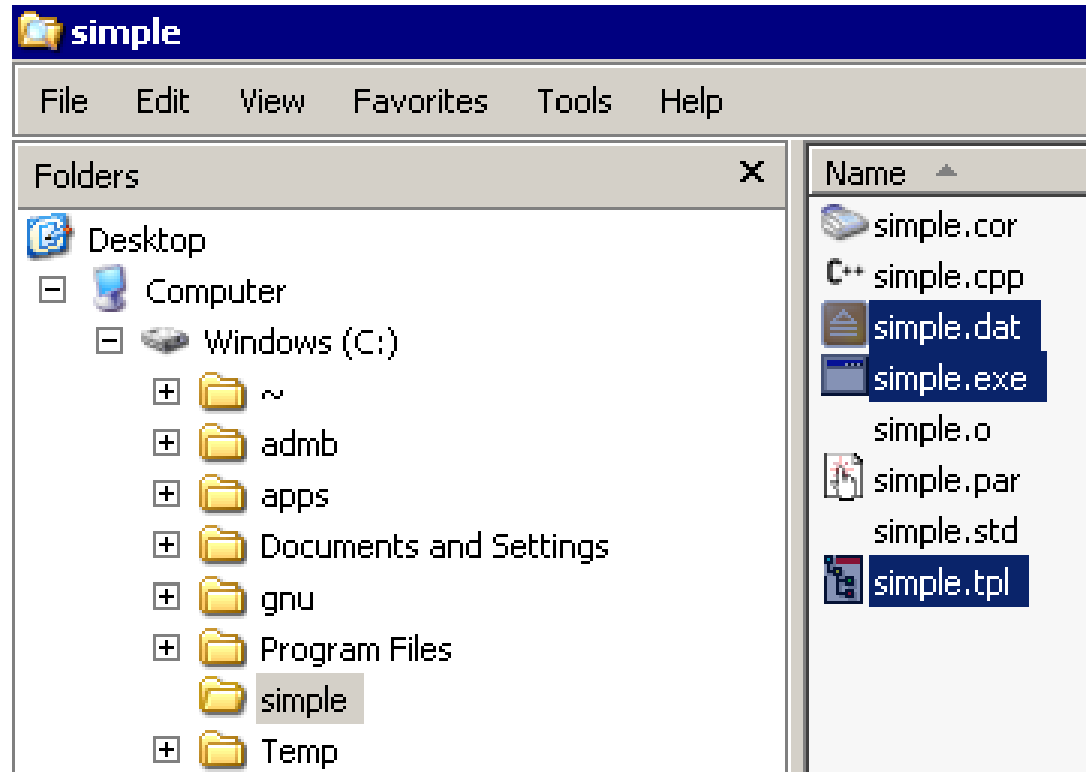
admb-project.org



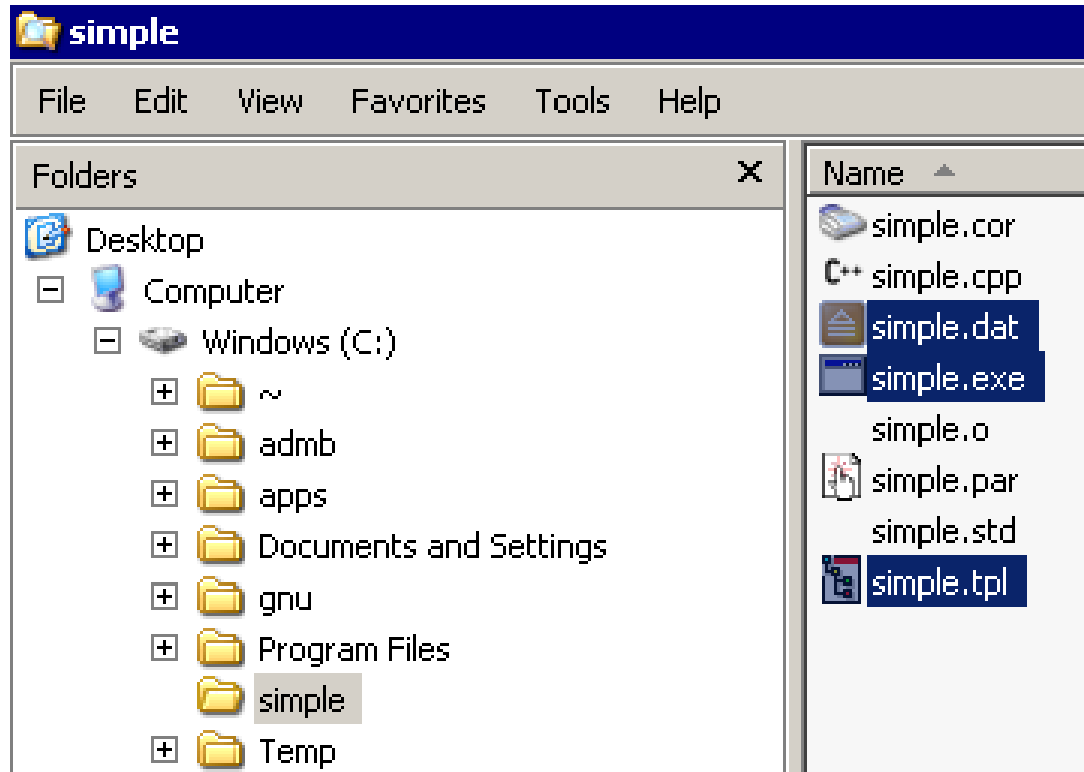
ADMB directory



Model directory



Model directory



Working environment

```
simple.tpl - Notepad
File Edit Format View Help

DATA_SECTION
  init_int n
  init_vector X(1,n)
  init_vector Y(1,n)

PARAMETER_SECTION
  init_number b0
  init_number b1
  vector Yfit(1,n)
  objective_function_value RSS

PROCEDURE_SECTION
  Yfit = b0 + b1*X;
  RSS = norm2(Y-Yfit);
```

1a Edit code

```
simple.dat - Notepad
File Edit Format View Help

# number of observations
10

# observed Y values
1.4 4.7 5.1 8.3 9.0 14.5 14.0 13.4 19.2 18

# observed x values
-1 0 1 2 3 4 5 6 7 8
```

1b Edit data

```
C:\ Dos
C:\simple>admb simple

*** tpl2cpp    simple
xxglobal.tmp
xxhtop.tmp
header.tmp
xxalloc.tmp
xxtopm.tmp
1 file(s) copied.

*** adcomp     simple
g++ -c -O3 -Wno-deprecated -D_GNUDOS__ -Dlinux -DOPT_LIB -DUSE_LAPLACE -fpermissive -I. -Ic:/admb/gcc440/include simple.cpp

*** adlink     simple
g++ -s -static -Lc:/admb/gcc440/lib simple.o -ldf1b2stub -ladmod -lادت -lادو -ldf1b2stub -ladmod -lادت -lادو -o simple

C:\simple>
C:\simple>
C:\simple>
C:\simple>simple

Initial statistics: 2 variables; iteration 0; function evaluation 0
Function value 2.0500000e+002; maximum gradient component mag -1.0682e+003
Var  Value  Gradient  !Var  Value  Gradient  !Var  Value  Gradient
1  0.00000 -7.00000e+001 ! 2  0.00000 -1.06820e+003 !

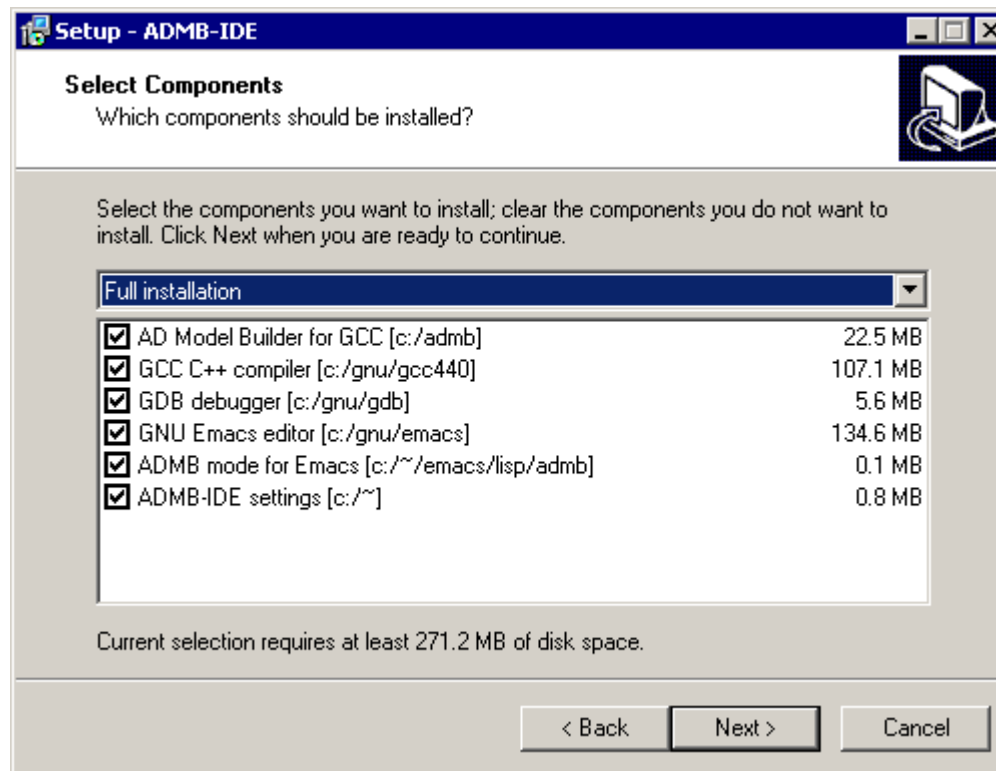
- final statistics:
2 variables; iteration 4; function evaluation 9
Function value 5.1313e+000; maximum gradient component mag -2.6534e-014
Exit code = 1; converg critr 1.0000e-004
Var  Value  Gradient  !Var  Value  Gradient  !Var  Value  Gradient
1 -1.78563 -2.22045e-015 ! 2  0.49123 -2.65344e-014 !
Estimating row 1 out of 2 for hessian
Estimating row 2 out of 2 for hessian

C:\simple>
```

2 Compile

3 Run model

ADMB-IDE



ADMB-IDE

The screenshot displays the ADMB-IDE application window. The main editor shows a C++ source file named `simple.cpp` with the following code:

```
#ifndef __ZTC__
extern unsigned int _stack=10000U;
#endif

long int arrmbldsize=0;

int main(int argc, char * argv[])
{
    ad_set_new_handler();
    ad_exit=ad_boundf;
    gradient_structure::set_NO_DERIVATIVES();
    gradient_structure::set_YES_SAVE_VARIABLES_VALUES();
    #if defined(__GNUDOS__) || defined(DOS386) || defined(__DPMI32__)
        if (arrmbldsize) arrmbldsize=150000;
    #else
        if (!arrmbldsize) arrmbldsize=25000;
    #endif
    model_parameters mp(arrmbldsize,argc,argv);
    mp.iprint=10;
    mp.preliminary_calculations();
    mp.computations(argc,argv);
    return 0;
}

extern "C" {
    void ad_boundf(int i)
    {
        // so we can stop here
    }
}
```

A menu is open over the editor, showing options like Translate, Compile, Link, Build, Run, and View Report. The bottom status bar shows the current line and column: Bot (45,20) (ADMB) and simple.cpp 62% (85,0) (C++/1 Abbrev).

The output window at the bottom displays the following text:

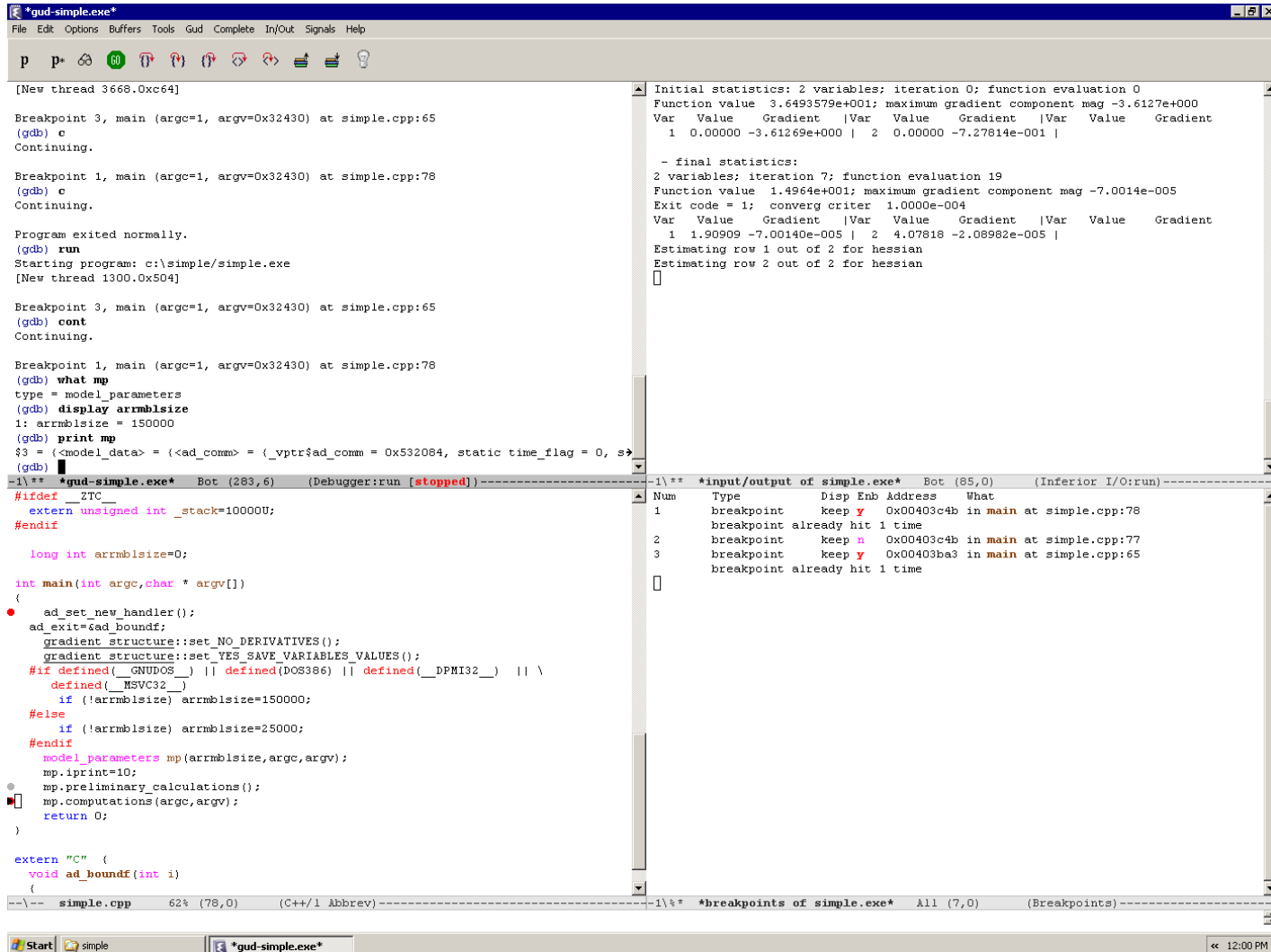
```
--(DOS)-- simple.tpl Bot (45,20) (ADMB)----- simple.cpp 62% (85,0) (C++/1 Abbrev)-----

Initial statistics: 2 variables; iteration 0; function evaluation 0
Function value 3.6493579e+01; maximum gradient component mag -3.6127e+00
Var Value Gradient |Var Value Gradient |Var Value Gradient
1 0.00000 -3.61269e+00 | 2 0.00000 -7.27814e-01 |

- final statistics:
2 variables; iteration 7; function evaluation 19
Function value 1.4964e+01; maximum gradient component mag -7.0014e-05
Exit code = 1; converg crit 1.0000e-04
Var Value Gradient |Var Value Gradient |Var Value Gradient
1 1.90909 -7.00140e-05 | 2 4.07818 -2.08982e-05 |
Estimating row 1 out of 2 for hessian
Estimating row 2 out of 2 for hessian
[]

-u:** *Async Shell Command* All (15,0) (Fundamental)-----
```

ADMB-IDE



```
*gud-simple.exe*
File Edit Options Buffers Tools Gud Complete In/Out Signals Help

p p+ [breakpoint icons]

[New thread 3668.0xc64]

Breakpoint 3, main (argc=1, argv=0x32430) at simple.cpp:65
(gdb) c
Continuing.

Breakpoint 1, main (argc=1, argv=0x32430) at simple.cpp:78
(gdb) c
Continuing.

Program exited normally.
(gdb) run
Starting program: c:\simple\simple.exe
[New thread 1300.0x504]

Breakpoint 3, main (argc=1, argv=0x32430) at simple.cpp:65
(gdb) cont
Continuing.

Breakpoint 1, main (argc=1, argv=0x32430) at simple.cpp:78
(gdb) what mp
type = model_parameters
(gdb) display armblsize
1: armblsize = 150000
(gdb) print mp
$3 = {<model_data> = {<ad_comm> = {_vptr$ad_comm = 0x532084, static time_flag = 0, s...
(gdb)

-1\** *gud-simple.exe* Bot (283,6) (Debugger:run [stopped]) -----1\** *input/output of simple.exe* Bot (85,0) (Inferior I/O:run) -----
Num Type Disp Enb Address What
1 breakpoint keep y 0x00403c4b in main at simple.cpp:78
  breakpoint already hit 1 time
2 breakpoint keep n 0x00403c4b in main at simple.cpp:77
3 breakpoint keep y 0x00403ba3 in main at simple.cpp:65
  breakpoint already hit 1 time

-1\** *breakpoints of simple.exe* All (7,0) (Breakpoints) -----

#ifdef __ZTC
extern unsigned int _stack=10000U;
#endif

long int armblsize=0;

int main(int argc,char * argv[])
{
    ad_set_new_handler();
    ad_exit=ad_boundf;
    gradient_structure::set_NO_DERIVATIVES();
    gradient_structure::set_YES_SAVE_VARIABLES_VALUES();
    #if defined(_GNUDOS_) || defined(DOS386) || defined(_DPHI32_) || \
    defined(_MSVC32_)
        if (!armblsize) armblsize=150000;
    #else
        if (!armblsize) armblsize=25000;
    #endif
    model_parameters mp (armblsize,argc,argv);
    mp.iPrint=10;
    mp.preliminary_calculations();
    mp.computations(argc,argv);
    return 0;
}

extern "C" {
void ad_boundf(int i)
{
}

simple.cpp 62% (78,0) (C++/1 Abbrev) -----1\** *breakpoints of simple.exe* All (7,0) (Breakpoints) -----

Start simple *gud-simple.exe* 12:00 PM
```


R packages

- CRAN: PBSadmb, (scapeMCMC)
- R-Forge: r2admb
- Otter Research: glmmADMB
- ADMB Project: admb2r
- ADMB Newsletter: import functions

Demo

- **ADMB-IDE**
 - download, manuals, simple, pella
- **Control file**
 - initial, limits, phases, priors
- **Advanced statistics**
 - hessian, mcmc, random effects