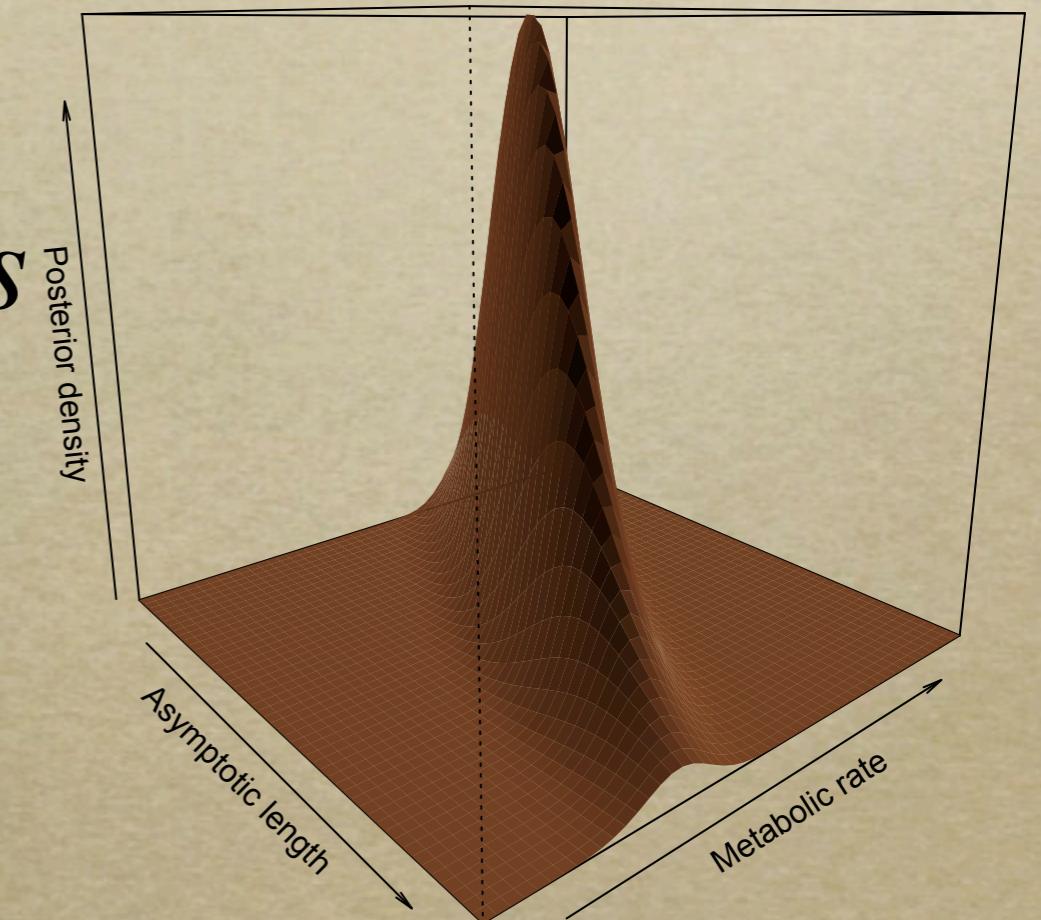


Introduction to nonlinear parameter estimation, likelihoods and AD Model Builder

ADMB Short Course
Steven Martell
James Ianelli

Optimization methods

- *Brute Force (Grid based)*
- *Derivative-based methods*
 - *e.g., Newtons' method*
- *derivative free methods*
 - *simplex method*



Derivative based methods

- *Most algorithms use finite differences*

$$\frac{df(x)}{dx} = \lim_{\Delta x \rightarrow 0} \frac{\Delta f(x)}{\Delta x} \approx \frac{f(x + \Delta x) - f(x)}{\Delta x}$$

- *i.e. a numerical approximation to the derivative; Simple, Inaccurate, & Slow.*

Analytical Derivatives

- *ADMB uses analytical derivatives; less simple, more accurate, & fast*
- *ADMB carries out the not-so simple derivative calculations behind the scenes!*

Numerical vs Analytical derivatives

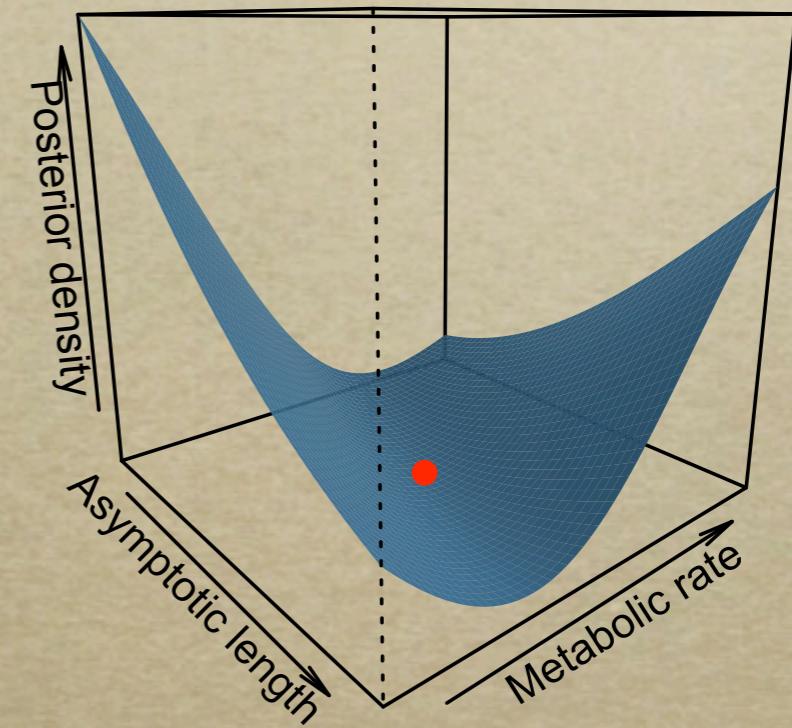
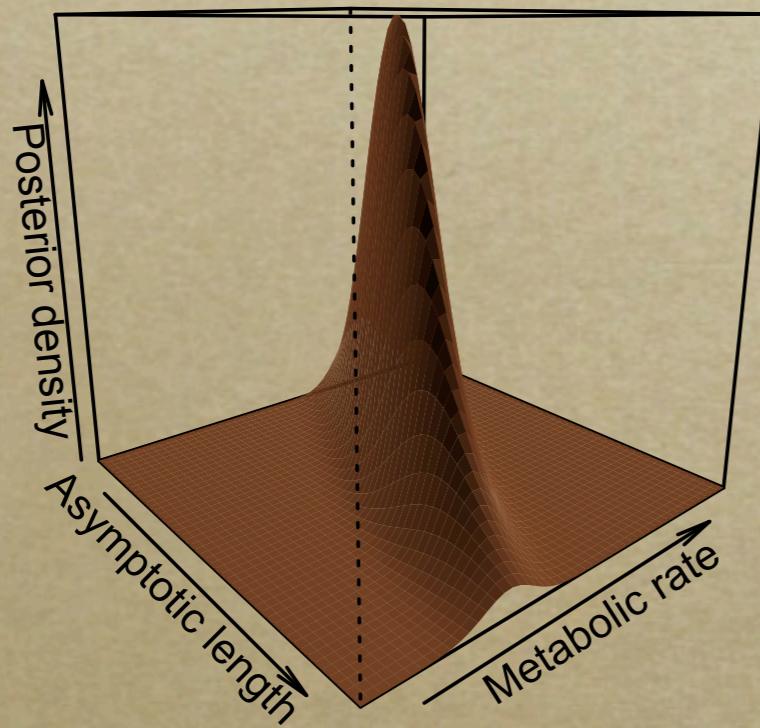
$$f(x) = 5x - \ln(x); \quad f'(x) = 5 - 1/x; \quad f''(x) = 1/x^2$$

| Iteration | Analytical Derivative | | | | Numerical derivative | | |
|-----------|-----------------------|-----------|----------|--|----------------------|-----------|----------|
| | x | f'(x) | f''(x) | | x | f'(x) | f''(x) |
| 1 | 0.01 | -9.5E+01 | 1E+04 | | 0.01 | -9.45E+01 | 9.8E+03 |
| 2 | 0.0195 | -4.63E+01 | 2.63E+03 | | 0.01964 | -4.58E+01 | 2.57E+03 |
| 3 | 0.0371 | -2.2E+01 | 7.27E+02 | | 0.03748 | -2.16E+01 | 7.08E+02 |
| 4 | 0.0673 | -9.86E+00 | 2.21E+02 | | 0.06805 | -9.68E+00 | 2.15E+02 |
| 5 | 0.112 | -3.93E+00 | 7.98E+01 | | 0.11303 | -3.84E+00 | 7.81E+01 |
| 6 | 0.1613 | -1.2E+00 | 3.85E+01 | | 0.16222 | -1.16E+00 | 3.8E+01 |
| 7 | 0.1925 | -1.95E-01 | 2.7E+01 | | 0.19285 | -1.84E-01 | 2.69E+01 |
| 8 | 0.1997 | -7.1E-03 | 2.51E+01 | | 0.1997 | -6.22E-03 | 2.5E+01 |
| 9 | 0.2 | 0E+00 | 2.5E+01 | | 0.19995 | -4.61E-06 | 2.5E+01 |
| 10 | 0.2 | 0E+00 | 2.5E+01 | | 0.19995 | 2E-09 | 2.5E+01 |
| 11 | 0.2 | 0E+00 | 2.5E+01 | | 0.19995 | 0E+00 | 2.5E+01 |

Minimize negative log-likelihood

$$L(y_i|\mu, \sigma) = \prod_{i=1}^n \frac{1}{\sigma\sqrt{2\pi}} \exp \left[-\frac{(y_i - \mu)^2}{2\sigma^2} \right]$$

$$\ell(y_i|\mu, \sigma) = n \left[\ln(\sigma) + \frac{1}{2} \ln(2\pi) \right] + \sum_{i=1}^n \frac{(y_i - \mu)^2}{2\sigma^2}$$



Likelihoods

$$\ell(\text{data} | \Theta)$$

What is a likelihood

- A *likelihood function is a probability model.*
$$\ell(\text{data} | \Theta)$$
- What is the likelihood of the data given the parameters?
- We treat the data as fixed and vary the parameters to find the most likely values

Why do we use -ve log likelihoods?

- *Likelihoods are multiplicative*

$$L(y_i|\mu, \sigma) = \prod_{i=1}^n \frac{1}{\sigma\sqrt{2\pi}} \exp\left[-\frac{(y_i - \mu)^2}{2\sigma^2}\right]$$

- *log likelihoods are additive*

$$\ell(y_i|\mu, \sigma) = n \left[\ln(\sigma) + \frac{1}{2} \ln(2\pi) \right] + \sum_{i=1}^n \frac{(y_i - \mu)^2}{2\sigma^2}$$

- *what happens if the number of observations gets very large?*

Four common negative log likelihoods kernels

Normal $\ell(x_i|\mu, \sigma) = n \ln(\sigma) + \frac{\sum_i (x_i - \mu)^2}{2\sigma^2}$

Lognormal $\ell(x_i|\mu, \sigma) = n \ln(\sigma) + \sum_i \ln(x_i) + \frac{\sum_i (\ln(x_i) - \mu)^2}{2\sigma^2}$

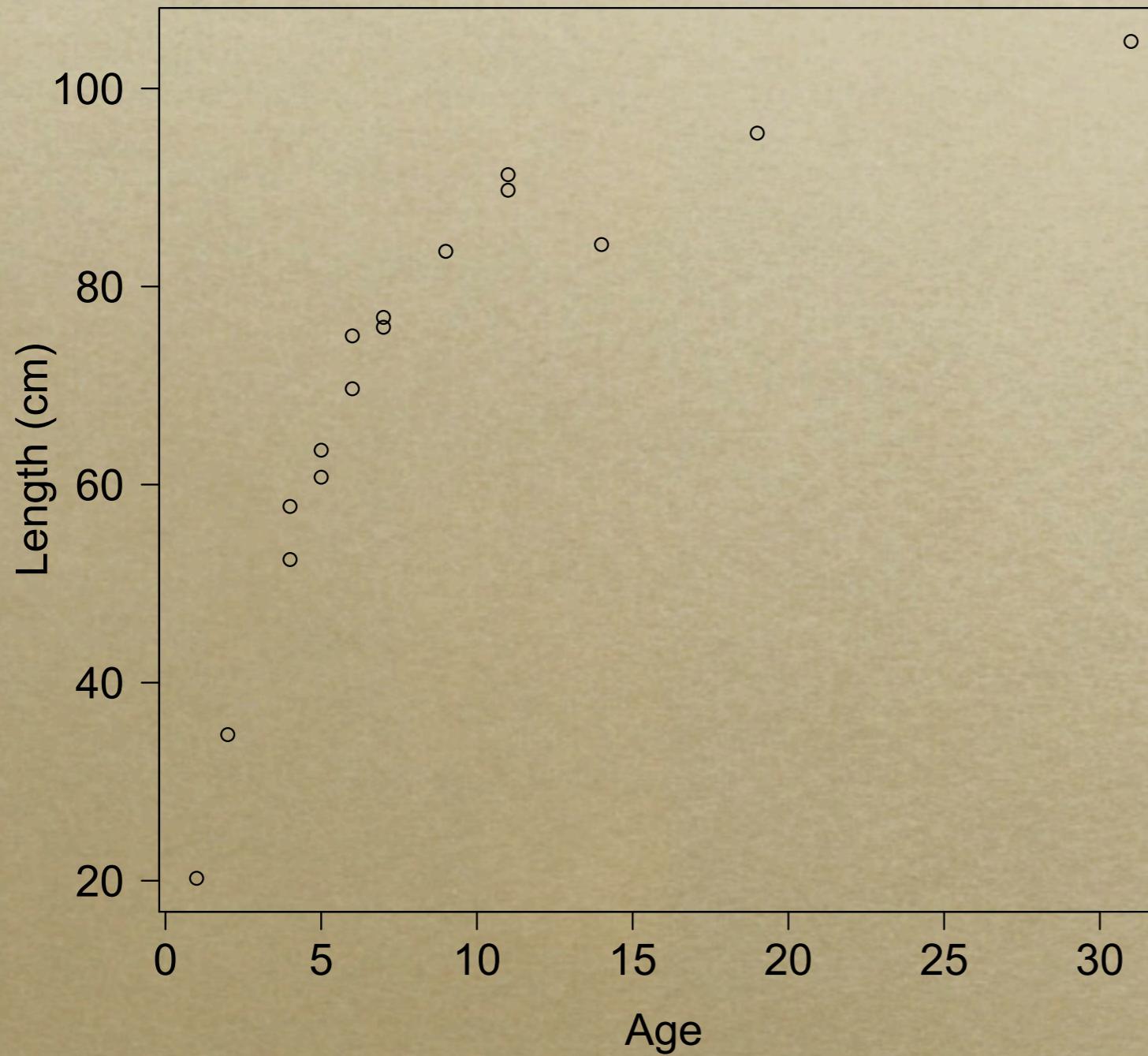
Poisson $\ell(x_i|\mu_i) = \sum_i \mu_i \ln(x_i) - \sum_i \mu_i$

Multinomial $\ell(x_i|p_i) = \sum_i x_i \ln(p_i)$

Example

- a simple growth model

Exercise: von Bertalanffy growth



| age | length |
|-----|--------|
| 9 | 84 |
| 11 | 90 |
| 11 | 91 |
| 7 | 77 |
| 6 | 70 |
| 7 | 76 |
| 5 | 63 |
| 4 | 58 |
| 6 | 75 |
| 2 | 35 |
| 1 | 20 |
| 5 | 61 |
| 4 | 52 |
| 19 | 95 |
| 31 | 105 |
| 14 | 84 |

ADMB Template -Bare Bones-

- DATA_SECTION
 - read in data and process data using LOC_CALCS
- PARAMETER_SECTION
 - declare model parameters
 - objective_function_value f; //minimize this value
 - declare global dvariables
- PROCEDURE_SECTION
 - define model procedures and calculate f;
 - must use “;” at the end of each line
- REPORT_SECTION
 - write results to a report file.

Model notation

Data

$$a_i, l_i$$

Parameters

$$l_\infty, k$$

Model & Residuals

$$\hat{l}_i = l_\infty (1 - \exp(-ka)) + \epsilon_i$$

$$\epsilon_i = l_i - \hat{l}_i$$

$$\sigma = \sqrt{\frac{1}{n} \sum_{i=1}^n \epsilon_i^2}$$

Negative loglikelihood

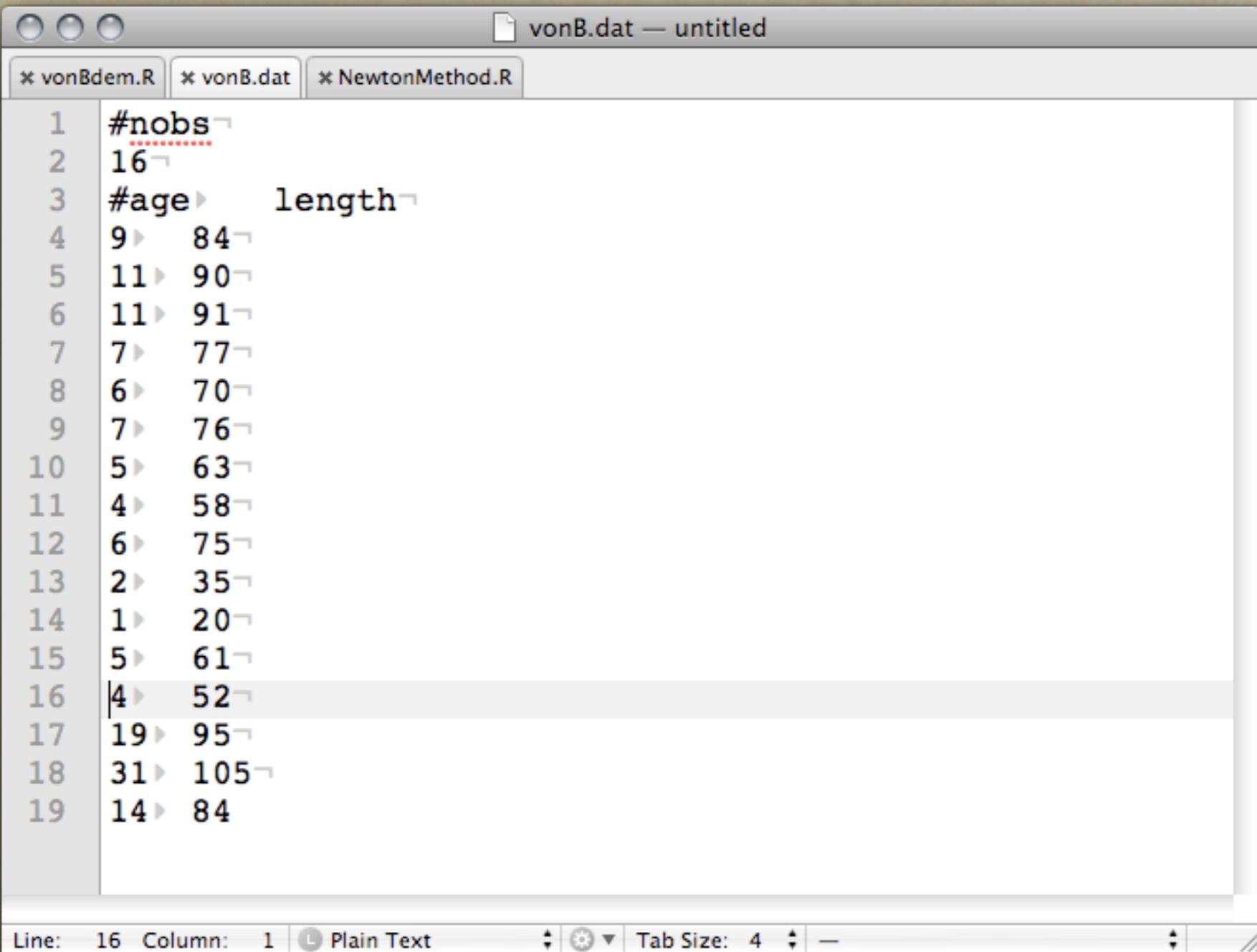
$$\ell(l_i | l_\infty, k, \tau) = n \left[\ln(\sigma) + \frac{1}{2} \ln(2\pi) \right] + \sum_{i=1}^n \frac{\epsilon_i^2}{2\sigma^2}$$

How ADMB works

- A 3 step process:
 - `tpl2cpp.exe` -- translates the `*.tpl` to a `*.cpp` file
 - `adcomp` -- translates the `*.cpp` file to `*.obj` file
 - `adlink` -- links the `*.obj` file to an `*.exe` file
- Errors can occur at each of these steps; this is useful for tracking down coding errors.
- Always use the `-s` option when developing models (safe model with array bound checking).

The data file

- default file name is the program name with the extension .dat



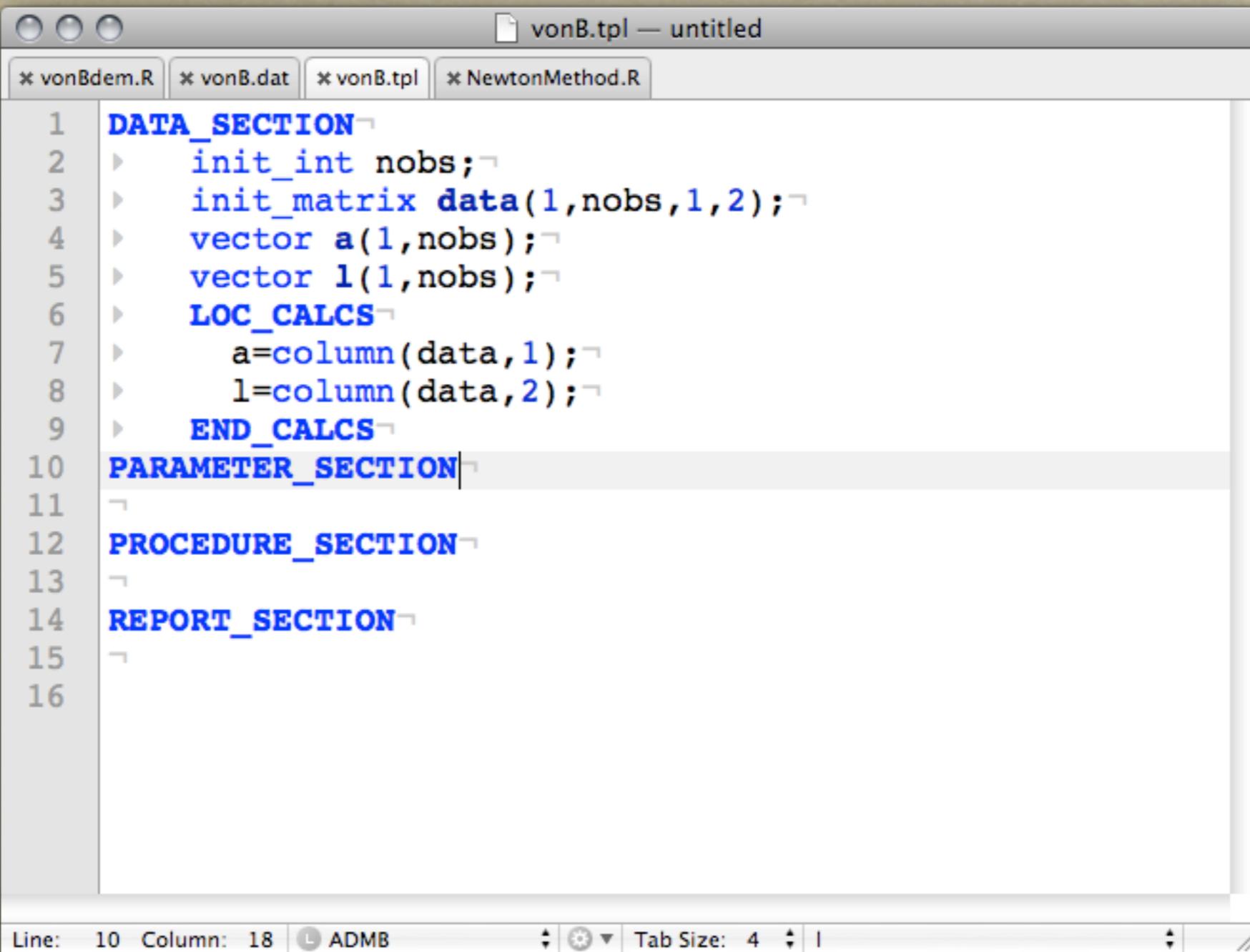
The screenshot shows a Mac OS X-style text editor window titled "vonB.dat — untitled". The window has three tabs at the top: "* vonBdem.R", "* vonB.dat", and "* NewtonMethod.R". The main pane displays a data file with 19 rows. The first row contains "#nobs" followed by a value of 16. Subsequent rows are labeled "#age" followed by a value of "length", with each row containing two values separated by a tab. The data is as follows:

| #age | length |
|------|--------|
| 9 | 84 |
| 11 | 90 |
| 11 | 91 |
| 7 | 77 |
| 6 | 70 |
| 7 | 76 |
| 5 | 63 |
| 4 | 58 |
| 6 | 75 |
| 2 | 35 |
| 1 | 20 |
| 5 | 61 |
| 4 | 52 |
| 19 | 95 |
| 31 | 105 |
| 14 | 84 |

At the bottom of the editor window, the status bar shows "Line: 16 Column: 1 Plain Text Tab Size: 4".

ADMB Template file

- use init_ prefix to read data from file



The screenshot shows a text editor window titled "vonB.tpl — untitled". The window contains the following ADMB template code:

```
1 DATA_SECTION
2   init_int nobs;
3   init_matrix data(1,nobs,1,2);
4   vector a(1,nobs);
5   vector l(1,nobs);
6   LOC_CALCS
7     a=column(data,1);
8     l=column(data,2);
9   END_CALCS
10 PARAMETER_SECTION
11 
12 PROCEDURE_SECTION
13 
14 REPORT_SECTION
15 
16
```

The code is color-coded: blue for section names like DATA_SECTION, LOC_CALCS, etc., and black for variable declarations and assignments.

ADMB Template file

- Use init_ prefix to declare parameters & !! is shorthand for LOC_CALCS

The screenshot shows a text editor window titled "vonB.tpl – untitled". The file contains ADMB template code. The code includes sections for data declaration, parameter initialization, and procedure/reporting sections. It uses the "init_" prefix for parameters and the "LOC_CALCS" and "END_CALCS" blocks for calculations. The "!!" operator is used to initialize parameters. The code is color-coded for readability.

```
2  ▷ init_int nobs;
3  ▷ init_matrix data(1,nobs,1,2);
4  ▷ vector a(1,nobs);
5  ▷ vector l(1,nobs);
6  ▷ LOC_CALCS
7  ▷     a=column(data,1);
8  ▷     l=column(data,2);
9  ▷ END_CALCS
10 ▷
11 PARAMETER_SECTION
12 ▷ init_number linf; //asymptotic length
13 ▷ init_number k;    //metabolic rate parameter
14 ▷ !! linf=max(1); //initial value for linf
15 ▷ !! k=0.2;        //initial guess for k
16 ▷ objective_function_value f;
17 ▷ vector lhat(1,nobs);
18 ▷
19 PROCEDURE_SECTION
20 ▷
21 REPORT_SECTION
22 ▷
```

Line: 17 Column: 25 ADMB Tab Size: 4 Ihat

ADMB Template

- Calculate objective function in the PROCEDURE SECTION

The screenshot shows a computer screen with a window titled "vonB.tpl — untitled". The window contains an ADMB template code. The code is color-coded: blue for sections like **PROCEDURE_SECTION**, green for comments like **//asymptotic length**, and black for variables and operations. The code defines parameters `linf` and `k`, initializes them, and calculates an objective function `f` based on observed data `l` and `a`. The **PROCEDURE_SECTION** is highlighted in light blue.

```
* vonBdem.R * vonB.dat * vonB.tpl * NewtonMethod.R
  DOC_CALCS
  7  ▶   a=column(data,1);^
  8  ▶   l=column(data,2);^
  9  ▶ END_CALCS^
  10 ▶ ^
  11 PARAMETER_SECTION
  12 ▶ init_number linf; //asymptotic length^
  13 ▶ init_number k;    //metabolic rate parameter^
  14 ▶ !! linf=max(l); //initial value for linf^
  15 ▶ !! k=0.2;        //initial guess for k^
  16 ▶ objective_function_value f;^
  17 ▶ vector lhat(1,nobs);^
  18 ^
  19 PROCEDURE_SECTION
  20 ▶ lhat=linf*(1.-exp(-k*a));^
  21 ▶ dvar_vector epsilon=l-lhat; //local declaration^
  22 ▶ dvariable sigma=sqrt(var(epsilon));^
  23 ▶ f=nobs*(log(sigma)+0.5*log(2.*3.141593))^
  24 ▶ +sum(square(epsilon))/(2.*sigma*sigma);^
  25 ^
  26 REPORT_SECTION
  27 ^

Line: 25 Column: 1 ADMB Tab Size: 4 log
```

ADMB Vs. optim() in R

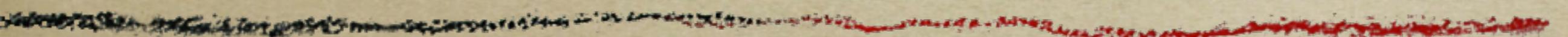
ADMB (n=12 function evaluations, f=42.680)

| Parameter | MLE | SD | Correlation | |
|-----------|--------|-------|-------------|------|
| | | | linf | k |
| linf | 99.000 | 2.210 | 1.00 | |
| k | 0.207 | 0.012 | -0.85 | 1.00 |

R (n=29 function evaluations, f=42.695)

| Parameter | MLE | SD | Correlation | |
|-----------|--------|-------|-------------|------|
| | | | linf | k |
| linf | 99.000 | 2.216 | 1.00 | |
| k | 0.207 | 0.012 | -0.84 | 1.00 |

Tools for estimating uncertainty in ADMB



Methods for estimating uncertainty

- Inverse Hessian
- Profile likelihood
- Markov Chain Monte Carlo (MCMC)

Inverse Hessian

- ADMB uses the delta method to calculate the inverse hessian.
- H^{-1} approximates the covariance matrix if the objective function is quadratic near its minimum.
- Standard deviations for all parameters, sdreport and likprof variables

$$H = \frac{\partial^2 \ell}{\partial \Theta_i \partial \Theta_j}$$

vonb.std — ADMB Course

* vonB.tpl * vonb.rep * vonb.prf * vonb.par * vonb.std * vonBdem.R

| 1 | index | name | value | std dev |
|----|-------|---------|------------|------------|
| 2 | 1 | linf | 9.9007e+01 | 2.2078e+00 |
| 3 | 2 | k | 2.0701e-01 | 1.1987e-02 |
| 4 | 3 | lp_linf | 9.9007e+01 | 2.2078e+00 |
| 5 | 4 | std_len | 1.8513e+01 | 6.5120e-01 |
| 6 | 5 | std_len | 3.3565e+01 | 1.0124e+00 |
| 7 | 6 | std_len | 4.5802e+01 | 1.1751e+00 |
| 8 | 7 | std_len | 5.5750e+01 | 1.2094e+00 |
| 9 | 8 | std_len | 6.3839e+01 | 1.1692e+00 |
| 10 | 9 | std_len | 7.0415e+01 | 1.0971e+00 |
| 11 | 10 | std_len | 7.5761e+01 | 1.0273e+00 |
| 12 | 11 | std_len | 8.0108e+01 | 9.8511e-01 |
| 13 | 12 | std_len | 8.3642e+01 | 9.8446e-01 |
| 14 | 13 | std_len | 8.6515e+01 | 1.0256e+00 |
| 15 | 14 | std_len | 8.8851e+01 | 1.0987e+00 |
| 16 | 15 | std_len | 9.0750e+01 | 1.1911e+00 |
| 17 | 16 | std_len | 9.2294e+01 | 1.2919e+00 |
| 18 | 17 | std_len | 9.3550e+01 | 1.3934e+00 |
| 19 | 18 | std_len | 9.4570e+01 | 1.4909e+00 |
| 20 | 19 | std_len | 9.5400e+01 | 1.5816e+00 |
| 21 | 20 | std_len | 9.6074e+01 | 1.6642e+00 |
| 22 | 21 | std_len | 9.6623e+01 | 1.7384e+00 |
| 23 | 22 | std_len | 9.7069e+01 | 1.8042e+00 |
| 24 | 23 | std_len | 9.7431e+01 | 1.8620e+00 |
| 25 | 24 | std_len | 9.7726e+01 | 1.9125e+00 |
| 26 | 25 | std_len | 9.7966e+01 | 1.9563e+00 |
| 27 | 26 | std_len | 9.8160e+01 | 1.9941e+00 |
| 28 | 27 | std_len | 9.8319e+01 | 2.0267e+00 |

Line: 16 Column: 22 Plain Text

Tab Size: 4

Parameter correlation from inverse hessian

```
vonb.cor — ADMB Course
* vonB.tpl * vonb.rep * vonb.prf * vonB.dat * vonb.hst * vonb.par * lp_linf.plt * vonb.std * vonb.cor * vonBdem.R
1 | The logarithm of the determinant of the hessian = 8.54727
2 index name      value      std dev      1      2      3
3   1 linf  9.9007e+01 2.2078e+00  1.0000
4   2 k    2.0701e-01 1.1987e-02 -0.8502  1.0000
5   3 lp_linf 9.9007e+01 2.2078e+00  1.0000 -0.8502  1.0000
6
```

Likelihood profiles

- Declare a *likeprof_number* to carryout a likelihood profile.
- Implement with the -lprof command line option.
- Results are reported in a *.plt file for each *likeprof_* variable.

Ip_linf.plt — ADMB Course

* vonB.tpl * vonb.rep * vonb.prf * vonB.dat * vonb.hst * vonb.par * Ip_linf.plt * vonb.std * vonb.cor * vonBdem.R

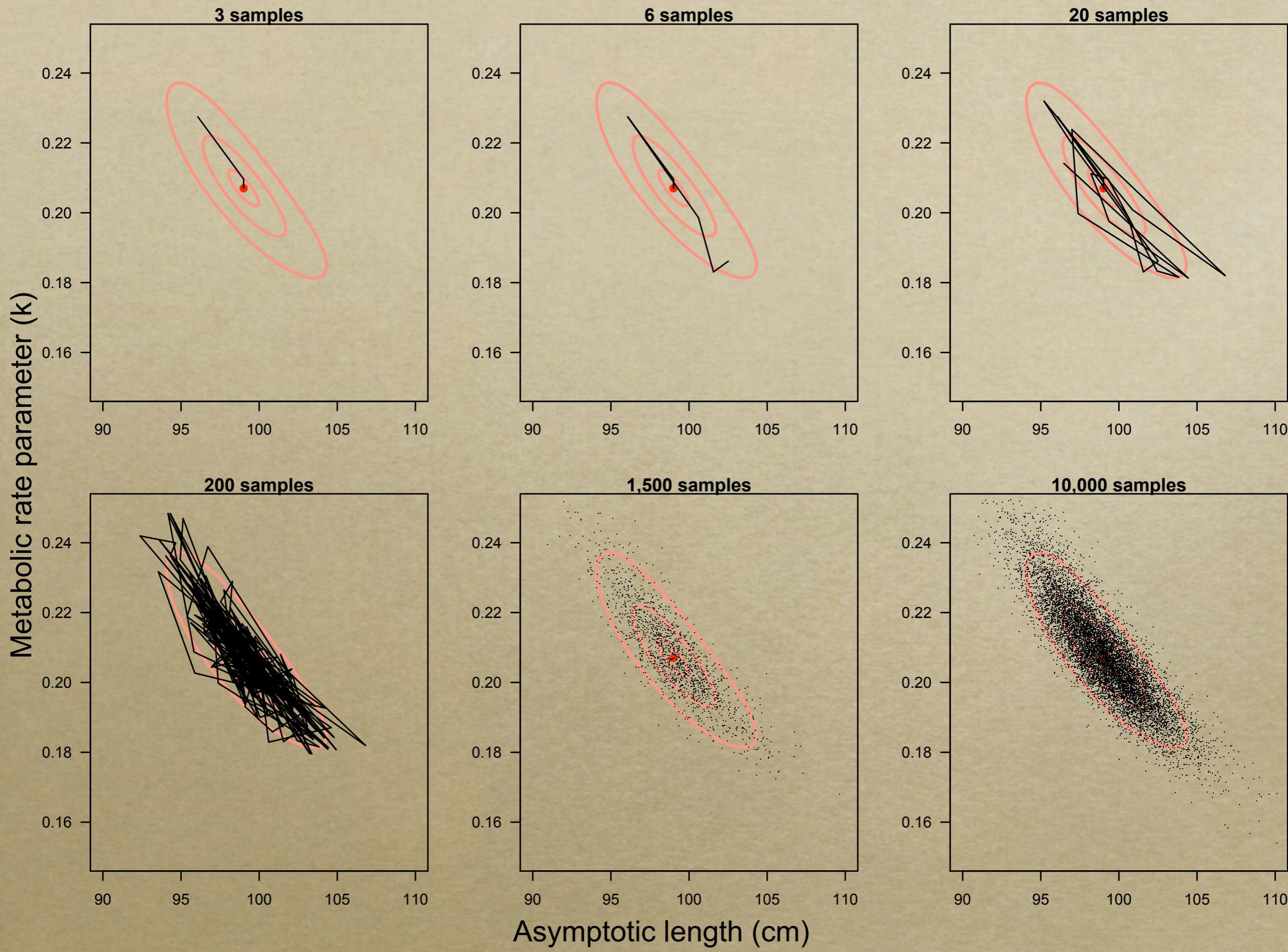
```
85 3.61326 2.0022
86 3.6176 1.93352
87 3.62194 1.88274
88 3.62628 1.83197
89 3.63062 1.7812
90 3.63496 1.73043
91 3.6393 1.67966
92 3.64364 1.62888
93 3.64798 1.57811
94 Minimum width confidence limits:
95 significance level lower bound upper bound
96 0.9 3.44667 3.60005
97 0.95 3.44628 3.62408
98 0.975 3.44589 3.63763
99
100 One sided confidence limits for the profile likelihood:
101
102 The probability is 0.9 that lp_linf is greater than 3.46321
103 The probability is 0.95 that lp_linf is greater than 3.45785
104 The probability is 0.975 that lp_linf is greater than 3.45417
105
106 The probability is 0.9 that lp_linf is less than 3.60022
107 The probability is 0.95 that lp_linf is less than 3.61994
108 The probability is 0.975 that lp_linf is less than 3.63245
109
110 Normal approximation
111 3.4054 0.632449
112 3.41195 1.23102
113 3.4185 1.82959
```

Markov Chain Monte Carlo (MCMC)

- Sampling of the likelihood surface to numerically approximate the joint posterior distribution.
- Two algorithms available:
 - Metropolis-hastings (-mcmc n)
 - a new Hybrid MCMC using Hamiltonian dynamics (-mcmc n -hybrid)
 - see Neal 2010 for more info

Metropolis-Hastings Algorithm

- (1) Start Markov chain at modal estimate of $\hat{\Theta}$,
- (2) take a random step from Θ_i to a new candidate point Θ'_i ,
- (3) accept trial Θ'_i with probability: $\min \left[\frac{P(\Theta'_i)}{P(\Theta_i)}, 1 \right]$,
- (4) if Θ'_i is accepted it becomes the new point in the chain,
otherwise Θ_i is repeated,
- (5) repeat steps 2-4 many many times.

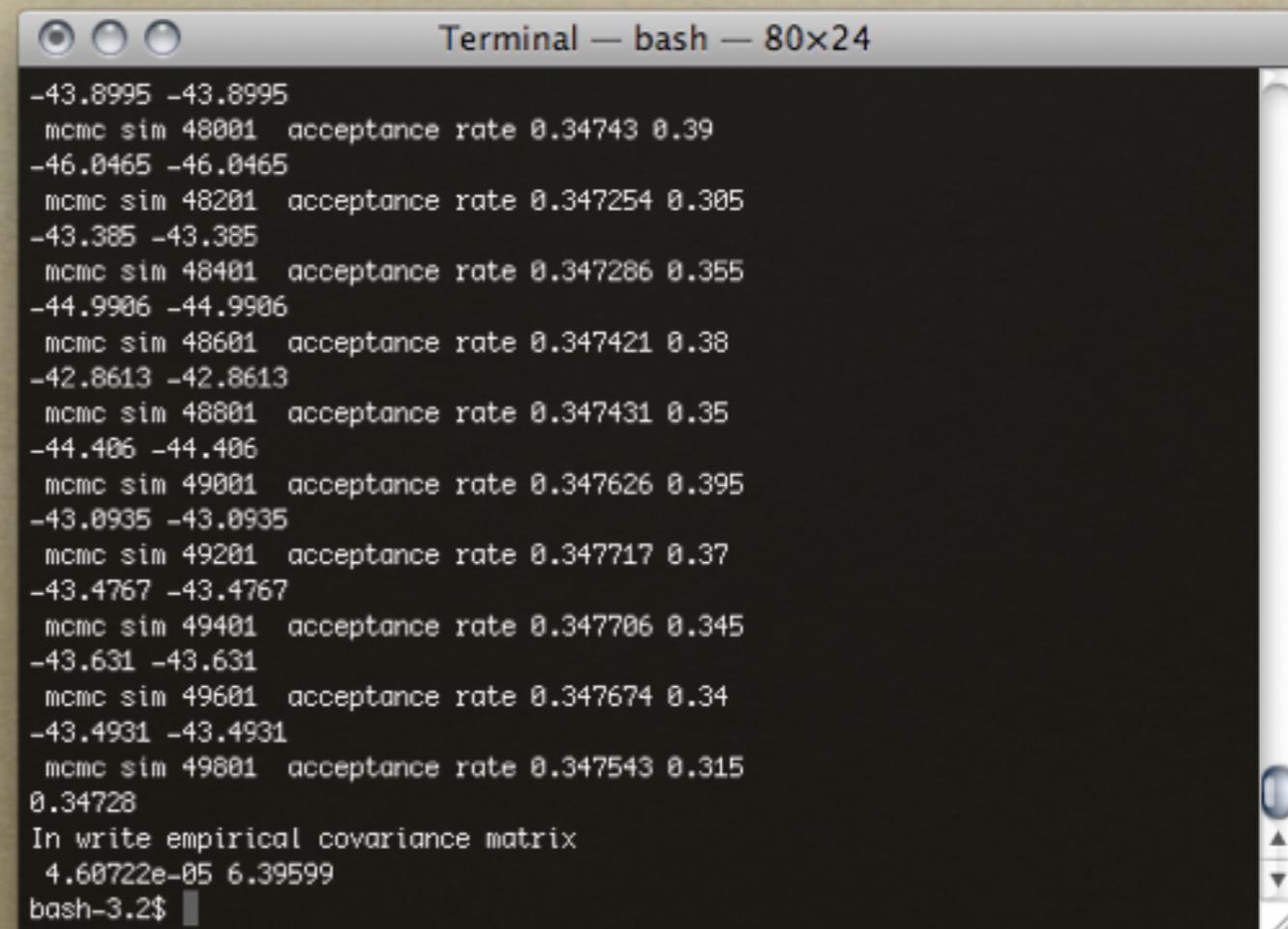


MCMC in AMDB

- To invoke the mcmc routines, you must have at least 1 `sdreport_` variable
- Use the `-mcmc n` command line option to run in MCMC mode
- Use `-mcsave n` to save samples from the joint posterior distribution
- Use `-mceval` to run the model using posterior samples.

MCMC diagnostics & options

- monitor acceptance rate
- must have positive definite hessian matrix
- -mcseed # to repeat results
- see command line options -? for more options



A screenshot of a terminal window titled "Terminal — bash — 80x24". The window displays a series of command-line outputs from an MCMC simulation. Each line shows a coordinate pair followed by the command "mcmc sim" and its parameters. The parameters include an acceptance rate (e.g., 0.34743, 0.39) and other numerical values. The last few lines of the output are: "In write empirical covariance matrix", "4.60722e-05 6.39599", and "bash-3.2\$".

```
-43.8995 -43.8995
mcmc sim 48001 acceptance rate 0.34743 0.39
-46.0465 -46.0465
mcmc sim 48201 acceptance rate 0.347254 0.305
-43.385 -43.385
mcmc sim 48401 acceptance rate 0.347286 0.355
-44.9906 -44.9906
mcmc sim 48601 acceptance rate 0.347421 0.38
-42.8613 -42.8613
mcmc sim 48801 acceptance rate 0.347431 0.35
-44.406 -44.406
mcmc sim 49001 acceptance rate 0.347626 0.395
-43.0935 -43.0935
mcmc sim 49201 acceptance rate 0.347717 0.37
-43.4767 -43.4767
mcmc sim 49401 acceptance rate 0.347706 0.345
-43.631 -43.631
mcmc sim 49601 acceptance rate 0.347674 0.34
-43.4931 -43.4931
mcmc sim 49801 acceptance rate 0.347543 0.315
0.34728
In write empirical covariance matrix
4.60722e-05 6.39599
bash-3.2$
```

Posterior density for L_{\inf}

