

Evaluating the Laplace approximation by Automatic Differentiation in nonlinear hierarchical models

Avd. C. October 24, 2003

Hans J. Skaug
Institute of Marine Research,
Bergen, Norway

Joint work with David Fournier, Otter Research

Outline

- Nonlinear hierarchical models
 - Mixed logistic regression
 - Poisson regression with random effects
- The Laplace approximation
- Automatic differentiation
- AD Model Builder (prototype)
 - Ordinal logistic regr., survival analysis

Hierarchical models

- Data: $\mathbf{y} = (y_1, \dots, y_n)$
 - Gaussian, binomial, Poisson, ...
- Latent variables: $\mathbf{u} = (u_1, \dots, u_q)$
 - Typically Gaussian
- Hyper parameters: $\boldsymbol{\theta} = (\theta_1, \dots, \theta_m)$
 - No prior knowledge
- Densities: $f_{\boldsymbol{\theta}}(\mathbf{y} | \mathbf{u})$ and $h_{\boldsymbol{\theta}}(\mathbf{u})$

Estimation (empirical Bayes)

- Marginal likelihood

$$L(\theta) = \int f_{\theta}(\mathbf{y} | \mathbf{u}) h_{\theta}(\mathbf{u}) d\mathbf{u}$$

- Computational challenge:
 - High dimensional integration with the view of maximizing $L(\theta)$
- Joint loglikelihood:

$$g(\mathbf{y}, \mathbf{u}, \theta) = \log[f_{\theta}(\mathbf{y} | \mathbf{u})] + \log[h_{\theta}(\mathbf{u})]$$

Mixed logistic regression

$$y_i | \mathbf{u} \sim \text{Bernoulli}(\pi_i),$$

$$\eta_i = \log\left(\frac{\pi_i}{1 - \pi_i}\right) = \mathbf{X}_i \boldsymbol{\beta} + \mathbf{Z}_i \mathbf{u}, \quad u_i \sim N(0, \sigma^2)$$

Hyperparameters: $\boldsymbol{\theta} = (\boldsymbol{\beta}, \sigma)$

$$\log[f_{\boldsymbol{\theta}}(\mathbf{y} | \mathbf{u})] = \sum_{i=1}^n [y_i \eta_i - \log(1 + \exp(\eta_i))]$$

$$\log[h_{\boldsymbol{\theta}}(\mathbf{u})] = -q \log(\sigma) - \frac{1}{2\sigma^2} \sum_{j=1}^p u_j^2.$$

ADMB code (C++)

```
g = -q*log(sigma) - .5*norm2(u/sigma);  
  
for(int i=1;i<=n;i++)  
{  
    eta = X(i)*b - Z(i)*u;  
    g += y(i)*eta - log(1+exp(eta));  
}
```

Poisson regression with spatially correlated random effects (Varin, Høst, Skare, 2003)

- $\{\xi(\mathbf{z}), \mathbf{z} \in R^2\}$ Gaussian random field

$$\text{cov}(\xi(\mathbf{z}), \xi(\mathbf{z}')) = \sigma^2 \exp(-\alpha^{-1}d(\mathbf{z}, \mathbf{z}'))$$

- y_i Poisson (λ_i) distributed

$$\log(\lambda_i) = \mathbf{X}_i\boldsymbol{\beta} + \xi(\mathbf{z}_i).$$

- Hyperparameters: $\boldsymbol{\theta} = (\boldsymbol{\beta}, \alpha, \sigma)$

ADMB code (C++)

```
// Correlation matrix C from distance matrix d
for(i=1;i<=n;i++)
  for(j=1;j<=n;j++)
    C(i,j) = exp(-d(i,j)/alpha);

g = -.5*norm2(u);           // log[h(u)]

L = choleski_decomp(C);
xi = sigma*L*u;

for (i=1;i<=n;i++) {
  eta = X(i)*b + xi(i);
  lambda = exp(eta);
  g += y(i)*eta - lambda;   // log[f(y(i)|u)]
}
```

Laplace approximation

$$L(\boldsymbol{\theta}) = \int f_{\boldsymbol{\theta}}(\mathbf{y} \mid \mathbf{u}) h_{\boldsymbol{\theta}}(\mathbf{u}) d\mathbf{u}$$

Second order Taylor expansion around:

$$\hat{\mathbf{u}}(\boldsymbol{\theta}) = \underset{\mathbf{u}}{\operatorname{argmax}} g(\mathbf{y}, \mathbf{u}, \boldsymbol{\theta}).$$

Likelihood approximation:

$$L^*(\boldsymbol{\theta}) = |\det(\mathbf{H}(\boldsymbol{\theta}))|^{-1/2} \exp(g(\mathbf{y}, \hat{\mathbf{u}}(\boldsymbol{\theta}), \boldsymbol{\theta})),$$

Hessian matrix:

$$\mathbf{H}(\boldsymbol{\theta}) = \frac{\partial^2}{\partial \mathbf{u}^2} g(\mathbf{y}, \mathbf{u}, \boldsymbol{\theta}) \Big|_{\mathbf{u}=\hat{\mathbf{u}}(\boldsymbol{\theta})}.$$

Evaluating the Laplace approximation

- By numerical optimization: $\hat{\mathbf{u}}(\theta)$
- Requires 2nd order derivatives: $\mathbf{H}(\theta)$
 - Calculated by Automatic Differentiation (AD)
- To help the optimization algorithm: $\nabla l(\theta)$
 - Involves 3rd order partial derivatives

AD (Automatic Differentiation)

- Given a computer code for $l(\theta)$
 - Calculates derivatives using the chain rule
- ‘Inserts’ derivative code into your program
 - or use operator overloading in C++
- AD gives you $\nabla l(\theta)$
 - at machine precision
 - ‘no’ extra programming
 - efficiently (reverse mode AD)
$$\text{cost}(\nabla l) \approx 4 \cdot \text{cost}(l)$$
- Software: ADIFOR, ADOL-C, ADMB ...

Reverse mode AD

Function: $l(\theta_1, \theta_2) = (\theta_1^2 - \theta_2) \sin(\theta_1 + \theta_2)$

Source code:

$$t_1 = \theta_1,$$

$$t_2 = \theta_2,$$

$$t_3 = t_1 \cdot t_1,$$

$$t_4 = t_3 - t_2,$$

$$t_5 = t_1 + t_2,$$

$$t_6 = \sin(t_5),$$

$$t_7 = t_4 \cdot t_6.$$

Reverse algorithm:

$$\frac{\partial t_7}{\partial t_6} = t_4,$$

$$\frac{\partial t_7}{\partial t_5} = \frac{\partial t_7}{\partial t_6} \cos(t_5),$$

$$\frac{\partial t_7}{\partial t_4} = t_6,$$

$$\frac{\partial t_7}{\partial t_3} = \frac{\partial t_7}{\partial t_4},$$

$$\frac{\partial t_7}{\partial t_2} = \frac{\partial t_7}{\partial t_5} - \frac{\partial t_7}{\partial t_4},$$

$$\frac{\partial t_7}{\partial t_1} = \frac{\partial t_7}{\partial t_5} + 2t_1 \frac{\partial t_7}{\partial t_3}.$$

Problems with AD (reverse)

- Large memory requirements
 - Can cause the computer to slow down

Empirical Bayes modeling

- What kind of framework do we want?
- Gilks, Thomas & Spiegelhalter (1994) about the goals for BUGS:
 - Accommodate a very large class of models
 - Fit the model automatically

ADMB prototype

- AD Model Builder (ADMB) is Fournier's system for parameter estimation
 - Model formulated in C++
- Allows variables to be declared as `random_effects`
- How are the goals from Gilks et al. achieved?
 - Flexibility: most of C++ to our disposition
 - Automatic (the user does not care about integration):
 - Laplace approximation
 - Automatic differentiation
- Prototype developed under BeMatA program

A comparison with BUGS

- Mixed logistic regression
 - $n=200, p=5, q=30$
- Speed (time to convergence):
 - WinBUGS (2,000 iterations): 700 seconds
 - ADMB: 27 seconds
- Robustness test ($n=50, p$ and q the same):
 - WinBUGS crashes
 - ADMB converges

Ordinal logistic regression

- Software review by Centre for Multilevel Modelling
 - Major statistical packages for multilevel modeling participates

$$P(y_i \leq y^{(s)}) = \frac{\exp(\kappa_s - \eta_i)}{1 + \exp(\kappa_s - \eta_i)}, \quad s = 1, \dots, S - 1.$$

	β_1	β_2	β_3	σ	κ_1	κ_2	κ_3	κ_4	κ_5	κ_6
ADMB-RE	1.953	0.684	2.775	2.229	-4.127	-2.390	0.402	1.337	2.225	3.265
aML	2.064	0.688	2.841	2.283	-4.056	-2.300	0.510	1.449	2.341	3.384

Survival analysis with frailties

- Cox's proportional hazard model

$$h_i(t) = h_0(t) \exp(\eta_i),$$

- Weibull baseline hazard: $h_0(t) = rt^{r-1}$

$$\eta_i = \beta_0 + \beta_{\text{age}} \cdot \text{age}_i + \beta_D \mathbf{x}_i + \beta_{\text{sex}} \cdot \text{sex}_i + u_i,$$

	β_0	β_{age}	β_1	β_2	β_3	β_{sex}	r	σ
ADMB-RE	-4.344	0.003	0.1208	0.6058	-1.1423	-1.8767	1.1624	0.5617
Std. dev.	0.872	0.0137	0.5008	0.5011	0.7729	0.4754	0.1626	0.297
BUGS	-4.6	0.003	0.1329	0.6444	-1.168	-1.938	1.215	0.6374

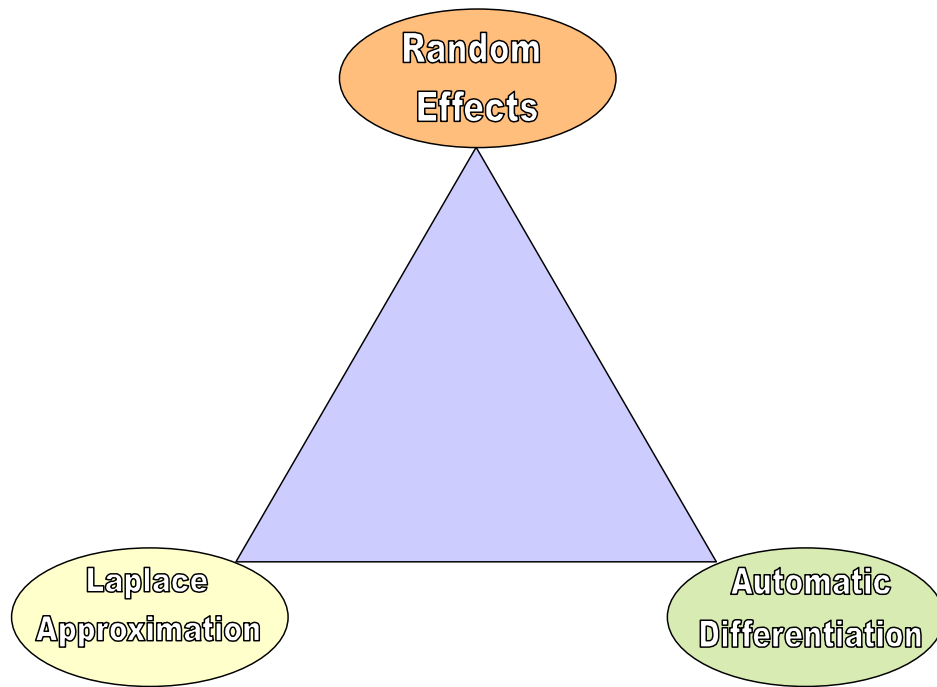
Comparison to MCMC

- Laplace approximation cannot handle discrete latent variable
 - MCMC methods more general
- Conditional independence structure causes difficulties
 - Trivial with MCMC (Gibbs sampling)
- Parallelization difficult with MCMC
- Simple convergence criterion
 - Difficult with MCMC

Other

- Laplace approximation can be used as a starting point for
 - Metropolis-Hastings (independent proposal)
 - Importance sampling

Summary



Annotated references

AD

Griewank, A. (2000). Evaluating Derivatives: Principles and Techniques of Algorithmic Differentiation, *SIAM, Philadelphia*.

AD and statistics

Skaug, H.J. (2002). Automatic differentiation to facilitate maximum likelihood estimation in nonlinear random effects models. *Journal of Computational and Graphical Statistics*. 11 p. 458-470.

BeMatA project

Title: Latent variable models handled with optimization aided by automatic differentiation; application to marine resource

<http://bemata.imr.no/>

Centre for Multilevel Modelling

Software review: <http://multilevel.ioe.ac.uk/softrev/index.html>

Laplace approximation in statistics

Tierney, L. and Kadane, J.B. (1986). Accurate approximations for posterior moments and marginal distributions. *Journal of the American Statistical Association* 81 p. 82-86