559

D: Various TMB issues

Fish 559; Day2: 13h30-15h00

Random and Fixed Effects-I

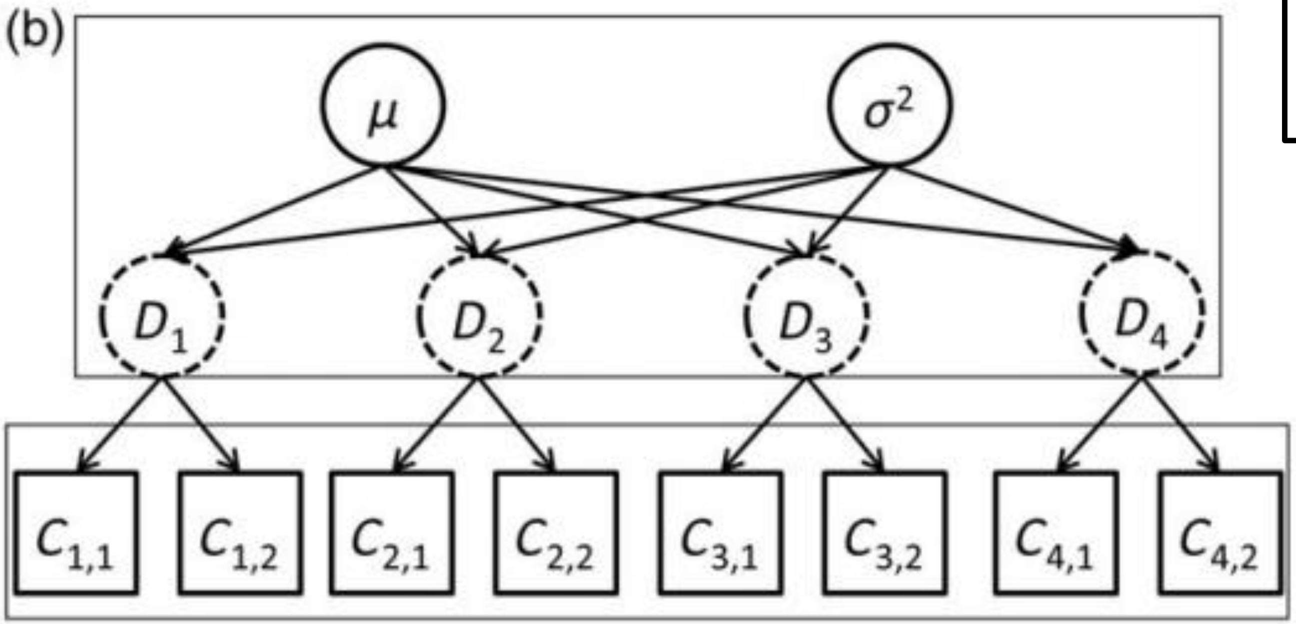
Some of the parameters of a model will be hierarchically motivated:

- *Random selection of sampling units with a population of such units for a growth study*
- *Random changes in recruitment success from a distribution of such changes.*

Known by other names:

Random effects (mixed effects) models
State-space models
Multi-level models
Hierarchical models
Generalized linear (or non-linear) mixed models
GAMMS

Motivating example



Average density and variability among sites

Density at 4 sites

Observed counts from 2 surveys at each site

- The site densities are related (dependent)
- The site densities are not directly observed (latent)

Thorson and Minto 2014

Some Theory

To fit a random (or mixed) effects model, we distinguish the vector of random effects (λ) from the vector of fixed effects (θ). The variance of the random effects is usually one of the fixed effects. The likelihood is the marginal likelihood, i.e.:

$$L(D\,|\,\theta) = \int L(D\,|\,\theta, \lambda)\, p(\lambda\,|\,\theta)\, d\lambda$$

where D is the data, θ is the vector of fixed effects, and λ is the vector of random effects.

We use the Laplace approximation to the integral, and select the values for the fixed effects to maximize the (approximated) integral.

Implementation Details

TMB allows some variables (vectors) to be random effects. A variable is treated as a random effect the same way as any other parameter, except:

- The variance of the parameter needs to be estimated.
- A penalty term needs to be included in the objective function:

```
for (int y=0;y<(MaxYear-1);y++)
  RecPenal -= dnorm(Nrec[y],0,SigmaR,true)
```

- The parameter needs to be declared as random in the MakeAD object:

```
MakeADFun(data=Data,parameters=Pars, random=c("Nrec"))
```

MakeADFun-I
(Key Options)

- data — List containing the data
- parameters — List containing the parameters
- map — Used to decide which parameters to estimate
- random — Which parameters are random? In a vector not a ls
- hessian — Calculate the Hessian at the optimum
- method — Optimization for the outer calculation
- DLL — What is the DLL name?
- control — Parameters to control the minimization (number of function calls, etc)
- silent — don’t report all sorts of stuff

MakeADFun-I
(outputs)

par, fn, and gr: parameter, likelihood and gradient function.

env: lots of good stuff here
- last.par: last parameter vector considered
- last.par.best: best parameter vector considered
- tracepar:

Example-I (An Age-structured Model)

Fit the following simple age-structured population dynamics model to CPUE, catch and catch-at-age data.

$$N_{y,a} = \begin{cases} N_{y_1,a} & \text{if } y = y_1 \\ N_{y,0} & \text{if } a = 0 \\ N_{y-1,a-1}\, e^{-(M + S_{a-1} F_{y-1})} & \text{otherwise} \end{cases}$$

$$S_a = \left(1 + \exp\left[-\ell n(19)\frac{a - a_{50}}{a_{95} - a_{50}}\right]\right)^{-1}$$

$$C_y = \sum_a w_a\, C_{y,a} \qquad C_{y,a} = \frac{S_a F_y}{M + S_a F_y} N_{y,a}\,(1 - e^{-(M + S_a F_y)})$$

Example-II (The Likelihood function)

The negative of the logarithm of the likelihood function includes three contributions: the catch data, the catch-at-age data, and the CPUE data. Likelihoods: normal with constant CV for the first, multinomial for the second, and log-normal for the third.

$$L_1 = \frac{1}{2\sigma_C^2}\left(\frac{C_y^{obs} - C_y}{C_y}\right)^2$$

$$L_2 = -\omega \sum_y \sum_a \rho_{y,a}^{obs} \, \ell\mathrm{n}(\hat{\rho}_{y,a} / \rho_{y,a}^{obs})$$

$$\rho_{y,a}^{obs} = C_{y,a}^{obs} / \sum_{a'} C_{y,a'}^{obs}$$

$$\hat{\rho}_{y,a} = C_{y,a} / \sum_{a'} C_{y,a'}$$

$$L_3 = \frac{1}{2\sigma_q^2} \sum_y \left(\ell\mathrm{n} I_y - \ell\mathrm{n}(q\, B_y)\right)^2$$

$$B_y = \sum_a w_a \, S_a \, N_{y,a} \, \mathrm{e}^{-(M + S_a F_y)/2}$$

Example-II (Parameters and Data)

- Numbers in the first year and for age 0 (estimated).
- Fully-selected fishing mortality by year (estimated).
- Ages-at-50%- and at-95%-selectivity (estimated).
- Natural mortality (pre-specified).
- Weight-at-age (pre-specified).
- Catchability (estimated).
- Weights to apply to the catch, CPUE and catch-at-age data (pre-specified).
- Catch, CPUE and observed catch proportion-at-age data (pre-specified).

Specifying the data

```
DATA_INTEGER(Nyear)
DATA_INTEGER(Nage)
DATA_SCALAR(M)
DATA_VECTOR(Wght)
DATA_SCALAR(SigCatch)
DATA_SCALAR(SigCPUE)
DATA_SCALAR(Omega)
DATA_VECTOR(Catch)
DATA_VECTOR(CPUE)
DATA_MATRIX(Propn)
```

The dimensions of the arrays are determined from R.

SCALAR -> real

If you have variables that are functions of the data, define them in R to save computational demands

Specifying the parameters and derived variables

```
PARAMETER(dummy);
PARAMETER_VECTOR(LogN);
PARAMETER(Sel50);
PARAMETER(Sel95);
PARAMETER_VECTOR(LogFish);
PARAMETER(logq);
```

The dimensions of the arrays are determined from R.

The names must match those in the *parameters* argument in R

```
matrix<Type> N(Nyear+1,Nage);
matrix<Type> F(Nyear,Nage);
matrix<Type> Z(Nyear,Nage);
vector<Type> S(Nage);
vector<Type> FullF(Nyear);
```

Temporary variables that are functions of the parameters

```
Type obj_fun;
```

This is the function to minimize.

Information to report

```
REPORT(S);
REPORT(N);
ADREPORT(FullF);
REPORT(CPUEPred);
REPORT(Like1);
REPORT(Like2);
REPORT(Like3);
REPORT(obj_fun);


return(obj_fun);
```

Tricks in R-I

I write the *cpp* and *R* code at the same time (in particular to check the data and parameters are correctly specified).

I define a "dummy*dummy" objective function to keep things simple (and use "*map*" to make sure the parameters were not being estimated, except *dummy*).

Tricks in R-II

Code to run the function once

```
# test code - for checking for minimization
xx <- model$fn(model$env$last.par)
#cat(model$report()$,"\n")
AAA

# Actual minimzation (with some "Bonus" parameters from
nlminb)
fit <- nlminb(model$par, model$fn, model$gr,
control=list(eval.max=100000,iter.max=1000))
best <- model$env$last.par.best
rep <- sdreport(model)
print(best)
print(rep)
```

Tricks in R-III

Adding bounds (lectD2.R):

```
fit <- nlminb(model$par,model$fn,model$gr,lower=c(4,1,-Inf),upper=c(Inf,1.5,Inf))
```

Three parameter model

Phasing (lectD3.R):

```
print("Phase 1; No log-Sigma")
map <- list(logSigma=factor(NA))
model <- MakeADFun(data, parameters, DLL="LectB2",map=map,silent=T)
fit <- nlminb(model$par, model$fn, model$gr)
best <- model$env$last.par.best
print(as.numeric(best))

print("Phase 2; All parameters - note I ordered my parameters to make this easier")
Phase2Init <- c(as.numeric(best),0)
model <- MakeADFun(data, parameters, DLL="LectB2",silent=T)
fit <- nlminb(Phase2Init, model$fn, model$gr)
```

TECHNICAL HELP FROM COLE MONNAHAN

559

Laplace approximation details I

1. Define joint log-likelihood:

$$f(\theta,\varepsilon) = \log(\Pr(y|\theta_1,\varepsilon)\Pr(\varepsilon|\theta_2))$$

2. Taylor series expansion of joint log-likelihood

$$f(\varepsilon|\theta) \approx f(\hat{\varepsilon}|\theta) + f'(\hat{\varepsilon}|\theta)(\hat{\varepsilon}-\varepsilon) + \frac{1}{2}f''(\hat{\varepsilon}|\theta)(\hat{\varepsilon}-\varepsilon)^2$$

3. Evaluate Taylor series around "inner maximum"

$$\hat{\varepsilon} = \mathrm{argmax}_{\varepsilon}\big(f(\theta,\varepsilon)\big)$$

4. Approximate joint likelihood via Taylor series

$$\Pr(y|\theta_1,\varepsilon)\Pr(\varepsilon|\theta_2) = e^{f(\varepsilon|\theta)} \approx e^{f(\hat{\varepsilon}|\theta)-\frac{1}{2}|f''(\hat{\varepsilon})|(\hat{\varepsilon}-\varepsilon)^2}$$

Laplace approximation details II

- Integrate both sides

$$\int \Pr(y|\theta_1, \varepsilon) \Pr(\varepsilon|\theta_2) \, \mathrm{d}\varepsilon = \int e^{f(\varepsilon|\theta)} \mathrm{d}\varepsilon$$

$$\int \Pr(y|\theta_1, \varepsilon) \Pr(\varepsilon|\theta_2) \, \mathrm{d}\varepsilon \approx e^{f(\hat{\varepsilon}|\theta)} \int e^{-\frac{1}{2}|f''(\hat{\varepsilon})|(\hat{\varepsilon}-\varepsilon)^2} \mathrm{d}\varepsilon$$

- Recognize that it looks like a normal distribution
- $\hat{\varepsilon}$ is the mean of the normal distribution
- $f''(\hat{\varepsilon})$ is the hessian of the normal distribution $(f''(\hat{\varepsilon}) = \Sigma^{-1})$

$$Normal\ PDF\text{:}\ \Pr(\varepsilon|\mu, \Sigma) = \frac{1}{(2\pi)^{k/2}|\Sigma|^{1/2}} \exp\left(\frac{-(\varepsilon-\mu)^{\mathrm{T}}\Sigma^{-1}(\varepsilon-\mu)}{2}\right)$$

559

TMB Laplace approximation steps

1. Write joint log-likelihood $\Pr(y, \varepsilon|\theta)$ in CPP file

$$f(\theta, \varepsilon) = \log(\Pr(y|\theta_1, \varepsilon)\Pr(\varepsilon|\theta_2))$$

2. Choose values for fixed θ_0 and random ε_0
3. "Inner optimization" – Optimize random effects with $\theta_0$held constant

$$\hat{\varepsilon} = \operatorname{argmax}_{\varepsilon}\big(f(\theta_0, \varepsilon)\big)$$

4. Calculate Laplace approx. for marginal likelihood of fixed effects

$$\ln L(\theta_0; y) \cong f(\theta_0, \hat{\varepsilon}) - \frac{1}{2}\log(|\mathbf{H}|)$$

- **TMB also provides the gradient of the penalized likelihood with respect to fixed effects**

5. "Outer optimization" – Repeat steps 2-3

- Outer optimization is done in R using the function value and gradient provided by TMB

559

Predicting random variables

- If we integrate out random effects, how do we make predictions?
- Predict random variables ε via fixed values for θ

$$\hat{\varepsilon} = \operatorname{argmax}_{\varepsilon}(\Pr\left(y\middle|\hat{\theta},\varepsilon\right)\Pr\left(\varepsilon|\hat{\theta}\right))$$

- $\hat{\theta}$ is the MLE of fixed effects θ
- $\hat{\varepsilon}$ are the "**Empirical Bayes**" estimates
- Confusing to think about, but remember these are not estimated parameters!