A short introduction to Template Model Builder (TMB)

Joaquin Cavieres
PhD candidate in Statistics

Instituto de Estadística, Universidad de Valparaíso

Overview

We are interested in modeling spatial observations, and for that we need to be clear about

- Formulation
- Implementation
- Evaluation

We are interested in modeling spatial observations, and for that we need to be clear about

- Formulation
- Implementation
- Evaluation

We are interested in modeling spatial observations, and for that we need to be clear about

- Formulation
- Implementation
- Evaluation

To formulate the models we need

- Building models based on the characteristics of the problem to solve
- Likelihood for the response variable

To implement the models we need

- Fit the model using our data for spatial observations
- Likelihood for the response variable

To evaluate the models

- Consider the Likelihood used
- Residual diagnostics
- Evaluate the uncertainty of the estimations

How we can do that? $\rightarrow$ Template Model Builder (TMB)

To evaluate the models

- Consider the Likelihood used
- Residual diagnostics
- Evaluate the uncertainty of the estimations

How we can do that? → Template Model Builder (TMB)

To evaluate the models

- Consider the Likelihood used
- Residual diagnostics
- Evaluate the uncertainty of the estimations

How we can do that? $\rightarrow$ Template Model Builder (TMB)

What is TMB?

- Template Model Builder (TMB) is an open source R package that enables quick implementation of complex nonlinear random effects (latent variables) (Kristensen et al. (2015)).

- TMB is a frequentist statistical platform that woks with Automatic differentiation (AD) to obtain the first and second derivatives of the a function (e.g. loglikelihood, Skaug and Fournier (2006))

- Offers an easy access to parallel computations and the user can define the joint likelihood for the data and the random effects in a C++ template function, while all the other operations are done in R .

What is TMB?

- TMB is designed for large, and complex hierarchical models
- Uses marginal maximum likelihood
- Uncertainty from the fitted model can be calculated by: estimating asymptotic variance-covariance matrices and computing likelihood profiles for parameters and model outputs
- Compatible with Stan to make Bayesian inference

Why use TMB?

- The objective function (and its derivatives) can be called from R, hence, the parameter optimization can be done via e.g. `nlminb()`.
- The user can specify use the Laplace approximation to obtain the marginal likelihood of the latent variables (random effects).
- Compute the standard deviations of any parameter, or derived parameters by the Delta method.

Why use `TMB`?

- The objective function (and its derivatives) can be called from `R`, hence, the parameter optimization can be done via e.g. `nlminb()`.
- The user can specify use the Laplace approximation to obtain the marginal likelihood of the latent variables (random effects).
- Compute the standard deviations of any parameter, or derived parameters by the Delta method.

Installing TMB

- From github: https://github.com/kaskr/adcomp

  git clone https://github.com/kaskr/adcomp

- From R:

  install.packages("TMB")

  library(TMB)

  #test that TMB is working:

  runExample(all=TRUE)

Note: Maybe you need to install `Rtools`

- https://cran.r-project.org/bin/windows/Rtools/

Restart `R` after installing `Rtools` and run: `library(devtools)`

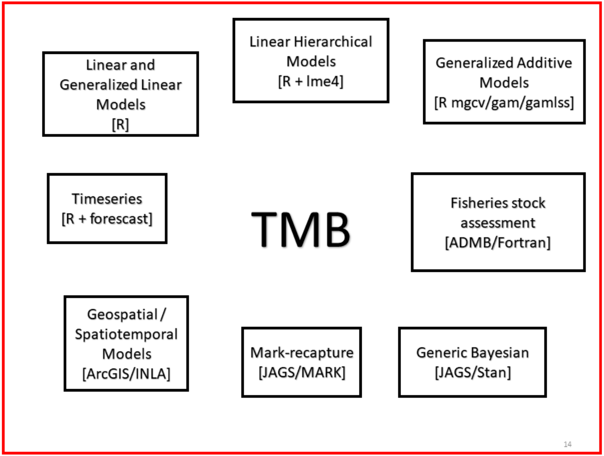

Figure 1: A summary of models what can be implemented in TMB (by Cole. C Monnahan)

TMB workflow

- Propose a statistical model
- Write a C++ template in R to calculate the negative log-likelihood given parameters
- Compile the model and load it in R
- Declare which parameters are "random"
- Fit the model using R and minimize the objective function returned by TMB
- Make inference from the fitted model

TMB in action

TMB modeling

- Compile a C++ file (template)
- Load in R (`dyn.load`)
- Build a `TMB` object (`MakeADFun`) based on a list of data and parameters
- The returned object is a list with many elements, including `fn` and `gr` functions
- The calculus is done in the C++ model (`TMB`)

TMB sections

The model is written in C++ and the structure of that TMB model has this structure:

- Read the data from R
- Set the parameters
- Calculate:
  1. Model expectation, given the parameters
  2. Negative log-likelihood (NLL)
- Report the results back to R
- Return the NLL

DATA section

Importing the data from R

TMB syntax	C++ syntax	R syntax
DATA_VECTOR(x)	vector<Type>	vector()
DATA_MATRIX(x)	matrix<Type>	matrix()
DATA_SCALAR(x)	Type	numeric()
DATA_INTEGER(x)	int	integer()
DATA_FACTOR(x)	vector<int>	factor()
DATA_ARRAY(x)	array<Type>	array()
DATA_SPARSE_MATRIX(x)	Eigen::SparseMatrix<Type>	dgTMatrix()

PARAMETER section

TMB syntax	C++ syntax	R syntax
`PARAMETER_MATRIX(x)`	`matrix<Type>`	`matrix()`
`PARAMETER_VECTOR(x)`	`vector<Type>`	`vector()`
`PARAMETER_ARRAY(x)`	`array<Type>`	`array()`
`PARAMETER(x)`	`Type`	`numeric()`

REPORT section

Reporting objects back to R

- Return objects via REPORT(), for example:

```
REPORT(prediction);
```

- In R you must do:

```
obj$report()
```

- Report parameters from the fitted model

```
obj$report(par)
```

Calculating the -log-likelihood

- Calculate likelihood using functions, e.g,. `dnorm()`

  ```
  nll= -dnorm(y(i), mu(i), sigma, true);
  ```

- For class type vectors, you need to do

  ```
  nll= -dnorm(y, mu, sigma, true).sum();
  ```

- You need to return the negative value
- In the last line of the C++ code you should put `return nll;`
- The `nll` must be a scalar Type variable

Calculating the variances

`TMB` return the assymptototic variances for the parameters using the function `sdreport(obj)`

- Calculate the standard errors from the derived quantities

```
Type nu = exp(eta);
ADREPORT(nu);
```

Some theory behind of `TMB`

Consider to $\boldsymbol{y} = (y_1, ..., y_n)^T$ a vector of observations and $\boldsymbol{u} = (u_1, ..., u_q)$ a vector of latent random variables (spatial random effects) influencing the value of $\boldsymbol{y}$. The conditional density of $\boldsymbol{y}$ given $\boldsymbol{u}$ is denoted by $(\boldsymbol{y} \mid \boldsymbol{u})$, and the marginal density of $\boldsymbol{u}$ is denoted by $h(\boldsymbol{y})$. Generally f and h depend on unknown parameters $\boldsymbol{\theta} = (\theta_1, ..., \theta_m)$ thus, we denoted this dependency by f_θ and h_θ respectively.

For estimation purposes the likelihood function for $\boldsymbol{\theta}$ must be expressed on function of the marginal distribution of $\boldsymbol{y}$, which is obtained by integrating out the random effects $\boldsymbol{u}$ from the joint density $f_\theta(\boldsymbol{y} \mid \boldsymbol{u})h_\theta(\boldsymbol{u})$, then the marginal likelihood can be written as:

$$\mathcal{L}(\boldsymbol{\theta}) = \int f_\theta(\boldsymbol{y} \mid \boldsymbol{u})h_\theta(\boldsymbol{u})d\boldsymbol{u} = \int \exp\{g(\boldsymbol{u}, \boldsymbol{\theta})\}d\boldsymbol{u} \tag{1}$$

where

$$g(\boldsymbol{u}, \boldsymbol{\theta}) = \log\{f_\theta(\boldsymbol{y} \mid \boldsymbol{u}) + \log\{h_\theta(\boldsymbol{u})\} \tag{2}$$

Solving this integral means a challenge and intensive computational work with the purpose of maximizing $\mathcal{L}(\boldsymbol{\theta})$. For the above, numerical methods can be performed to approximate the solution and one of them is the Laplace approximation. How $(\boldsymbol{\theta})$ is the vector of parameters, then we assume the function $g(\boldsymbol{u}, \boldsymbol{\theta})$ is such that:

$$\hat{\boldsymbol{u}}(\boldsymbol{\theta}) = \operatorname*{argmax}_{u} g(\boldsymbol{u}, \boldsymbol{\theta}) \tag{3}$$

and

$$\boldsymbol{H}(\boldsymbol{\theta}) = \frac{\partial^2}{\partial \boldsymbol{u}^2} = g(\boldsymbol{u}, \boldsymbol{\theta})\Big|_{\boldsymbol{u}=\hat{\boldsymbol{u}}(\boldsymbol{\theta})} \tag{4}$$

are well defined in the range of $\boldsymbol{\theta}$. Then, we can derive an efficient algorithm for maximizing the Laplace approximation as:

$$\mathcal{L}^\star(\boldsymbol{\theta}) = |\det\{\boldsymbol{H}(\boldsymbol{\theta})\}|^{-1/2} \exp\{g\{\hat{\boldsymbol{u}}, \theta\}\} \tag{5}$$

respect to $\boldsymbol{\theta}$. The critical point here is the numerical evaluation of the Hessian ($\boldsymbol{H}(\boldsymbol{\theta})$) but this can be solved by Automatic Differentiation.

Laplace approximation

- We must define a joint log-likelihood, for example:

$$f(\boldsymbol{\theta}, \boldsymbol{u}) = log(\Pr(\boldsymbol{y} \mid \theta_1), \boldsymbol{u}))\Pr(\boldsymbol{u} \mid \theta_2)$$

- Taylor series expansion of the joint log-likelioood

$$f(\boldsymbol{\theta}, \boldsymbol{u}) \approx f(\hat{\boldsymbol{u}} \mid \theta) + f^{'}(\hat{\boldsymbol{u}} \mid \theta)(\hat{\boldsymbol{u}} - \boldsymbol{u}) + \frac{1}{2} f^{''}(\hat{\boldsymbol{u}} \mid \theta)(\hat{\boldsymbol{u}} - \boldsymbol{u})^2$$

- Evaluate the second Taylor series around "inner maximum"

$$\hat{\boldsymbol{u}} = \operatorname*{argmax}_{\boldsymbol{u}} f((\boldsymbol{\theta}, \boldsymbol{u}))$$

- Approximate the joint likelihood via Taylor series

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

Laplace approximation

Integrating both sides of the equation

$$\int \Pr(\boldsymbol{y} \mid \theta_1, \boldsymbol{u}) \Pr(\boldsymbol{u} \mid \theta_2) d\boldsymbol{u} = \int e^{f(\boldsymbol{u}|\boldsymbol{\theta})} d\boldsymbol{u}$$

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

Do you recognize that it looks like a normal distribution?

- $\hat{\boldsymbol{u}}$ is the mean of the normal distribution
- $f''(\hat{\boldsymbol{u}})$ is the Hessian matrix ($f''(\hat{\boldsymbol{u}}) = \Sigma^{-1}$), so:

$$\mathcal{N}(\boldsymbol{u} \mid \mu, \Sigma) = \frac{1}{(2\pi)^{n/2}|\Sigma|^{1/2}} e^{\left(\frac{-(\boldsymbol{u}-\mu)^T \Sigma^{-1} (\boldsymbol{u}-\mu)}{2}\right)}$$

Laplace approximation

Integrating both sides of the equation

$$\begin{aligned}\int \Pr(\boldsymbol{y} \mid \theta_1, \boldsymbol{u}) \Pr(\boldsymbol{u} \mid \theta_2) d\boldsymbol{u} &= \int e^{f(\boldsymbol{u}|\boldsymbol{\theta})} d\boldsymbol{u} \\ \int \Pr(\boldsymbol{y} \mid \theta_1, \boldsymbol{u}) \Pr(\boldsymbol{u} \mid \theta_2) d\boldsymbol{u} &\approx e^{f(\hat{\boldsymbol{u}}|\boldsymbol{\theta})} \int e^{-\frac{1}{2}|f''(\hat{\boldsymbol{u}})|(\hat{\boldsymbol{u}}-\boldsymbol{u})^2}\end{aligned}$$

Do you recognize that it looks like a normal distribution?

- $\hat{\boldsymbol{u}}$ is the mean of the normal distribution
- $f''(\hat{\boldsymbol{u}})$ is the Hessian matrix ($f''(\hat{\boldsymbol{u}}) = \Sigma^{-1}$), so:

$$\mathcal{N}(\boldsymbol{u} \mid \mu, \Sigma) = \frac{1}{(2\pi)^{n/2}|\Sigma|^{1/2}} e^{\left(\frac{-(\boldsymbol{u}-\mu)^T \Sigma^{-1} (\boldsymbol{u}-\mu)}{2}\right)}$$

Laplace approximation in TMB

- Declare the joint log-likelihood in the C++ template

$$f(\boldsymbol{\theta}, \boldsymbol{u}) = log(\Pr(\boldsymbol{y} \mid \theta_1), \boldsymbol{u}))\Pr(\boldsymbol{u} \mid \theta_2)$$

- Put some coherent initial values for the fixed parameter $\boldsymbol{\theta}_0$ and for the random parameter $\boldsymbol{u}_0$

  "Inner optimization"

$$\hat{\boldsymbol{u}} = \underset{\boldsymbol{u}}{\operatorname{argmax}}\, f((\boldsymbol{\theta}_0, \boldsymbol{u}))$$

- Calculate the Laplace approximation for the marginal likelihood of the fixed effects

$$\log\mathcal{L}(\boldsymbol{\theta}_0; \boldsymbol{y}) \approx f(\boldsymbol{\theta}_0, \hat{\boldsymbol{u}}) - \frac{1}{2}\log(|\boldsymbol{H}|)$$

- "Outer optimization" (Repeat steps 2 - 3). Here the Outer optimization is done in R using the function and gradient provided by TMB

How to solve in an efficient way a spatial Hierarchical (Bayesian) model?

- `R-INLA` $\Longrightarrow$ Rue et al. (2009), Blangiardo et al. (2013), Lindgren et al. (2015), Bakka et al. (2018)
- `TMB` (Kristensen et al. (2015))

How to solve in an efficient way a spatial Hierarchical (Bayesian) model?

- `R-INLA` $\Longrightarrow$ Rue et al. (2009), Blangiardo et al. (2013), Lindgren et al. (2015), Bakka et al. (2018)
- `TMB` (Kristensen et al. (2015))

How to solve in an efficient way a spatial Hierarchical (Bayesian) model?

- `R-INLA` $\Longrightarrow$ Rue et al. (2009), Blangiardo et al. (2013), Lindgren et al. (2015), Bakka et al. (2018)
- `TMB` (Kristensen et al. (2015))

Examples

Conclusions

- `TMB` is an environment for fitting statistical models
- We can integrate random effects easily, and maximize the marginal likelihood
- It is more flexible, and more powerful than other software programs (`R` packages)
- Can be hard to use because you need to write a C++ code and need to compile it for changes in the model
- Incredibly flexible with the model types: if you can write the likelihood, `TMB` can probably fit it

Acknowledgments

- Cole. C Monnahan (NOAA) by provide me part of these slides
- Hans Skaug (University of Bergen) for his wonderful "case studies" in `TMB`

Thank You

References I

Bakka, H., Rue, H., Fuglstad, G.-A., Riebler, A., Bolin, D., Illian, J., Krainski, E., Simpson, D., and Lindgren, F. (2018). Spatial modeling with r-inla: A review. *Wiley Interdisciplinary Reviews: Computational Statistics*, 10(6):e1443.

Blangiardo, M., Cameletti, M., Baio, G., and Rue, H. (2013). Spatial and spatio-temporal models with r-inla. *Spatial and spatio-temporal epidemiology*, 4:33–49.

Kristensen, K., Nielsen, A., Berg, C. W., Skaug, H., and Bell, B. (2015). Tmb: automatic differentiation and laplace approximation. *arXiv preprint arXiv:1509.00660*.

Lindgren, F., Rue, H., et al. (2015). Bayesian spatial modelling with r-inla. *Journal of Statistical Software*, 63(19):1–25.

References II

Rue, H., Martino, S., and Chopin, N. (2009). Approximate bayesian inference for latent gaussian models by using integrated nested laplace approximations. *Journal of the royal statistical society: Series b (statistical methodology)*, 71(2):319–392.

Skaug, H. J. and Fournier, D. A. (2006). Automatic approximation of the marginal likelihood in non-gaussian hierarchical models. *Computational Statistics & Data Analysis*, 51(2):699–709.