Getting started with the \texttt{glmmTMB} package

Ben Bolker

January 11, 2019

1 Introduction/quick start

\texttt{glmmTMB} is an R package built on the Template Model Builder automatic differentiation engine, for fitting generalized linear mixed models and extensions. (Not-yet-implemented features are denoted \textit{like this})

- response distributions: Poisson, binomial, negative binomial (NB1 and NB2 parameterizations), Gamma, Beta, Gaussian; truncated Poisson and negative binomial; \textit{Student t}; Tweedie
- link functions: log, logit, probit, complementary log-log, inverse, identity
- zero-inflation with fixed and random-effects components; hurdle models via truncated Poisson/NB
- single or multiple (nested or crossed) random effects
- offsets
- fixed-effects models for dispersion
- diagonal, compound-symmetric, or unstructured random effects variance-covariance matrices; first-order autoregressive (AR1) variance structures

In order to use \texttt{glmmTMB} effectively you should already be reasonably familiar with generalized linear mixed models (GLMMs), which in turn requires familiarity with (i) generalized linear models (e.g. the special cases of logistic, binomial, and Poisson regression) and (ii) ‘modern’ mixed models (those working via maximization of the marginal likelihood rather than by manipulating sums of squares). Bolker et al. (2009) and Bolker (2015) are reasonable starting points in this area (especially geared to biologists and less-technical readers), as are Zuur et al. (2009), Millar (2011), and Zuur et al. (2013).

In order to fit a model in \texttt{glmmTMB} you need to:

- specify a model for the conditional effects, in the standard R (Wilkinson-Rogers) formula notation (see \texttt{?formula} or Section 11.1 of the \texttt{Introduction to R}). Formulae can also include \texttt{offsets}. 

1
• specify a model for the random effects, in the notation that is common to
the \textit{nlme} and \textit{lme4} packages. Random effects are specified as \texttt{x|g}, where
\texttt{x} is an effect and \texttt{g} is a grouping factor (which must be a factor variable,
or a nesting of interaction among factor variables). For example, the
formula would be \texttt{1|block} for a random-intercept model or \texttt{time|block}
for a model with random variation in slopes through time across groups
specified by \texttt{block}. A model of nested random effects (block within site)
would be \texttt{1|site/block}; a model of crossed random effects (block and
year) would be \texttt{(1|block)+(1|year)}.

• choose the error distribution by specifying the family (\texttt{family} argument).
In general, you can specify the function (\texttt{binomial}, \texttt{gaussian}, \texttt{poisson},
\texttt{Gamma} from base R, or one of the options listed at \texttt{family.glmmTMB}[\texttt{nbinom2},
\texttt{beta_family()}, \texttt{betabinomial}, ...]).

• choose the error distribution by specifying the family (\texttt{family} argument).
For standard GLM families implemented in R, you can use the function
name (\texttt{binomial}, \texttt{gaussian}, \texttt{poisson}, \texttt{Gamma}). Otherwise, you should
specify the family argument as a list containing (at least) the (character)
elements \texttt{family} and \texttt{link}, e.g. \texttt{family=list(family="nbinom2",link="log")}.

• optionally specify a zero-inflation model (via the \texttt{ziformula} argument)
with fixed and/or random effects

• optionally specify a dispersion model with fixed effects

This document was generated using R version 3.5.2 (2018-12-20) and package
versions:

```r
## bbmle glmmTMB
## 1.0.20 0.2.3
```

The current citation for \texttt{glmmTMB} is:

Brooks ME, Kristensen K, van Benthem KJ, Magnusson A, Berg
Balances Speed and Flexibility Among Packages for Zero-inflated

# 2 Preliminaries: packages and data

Load required packages:
library("glmmTMB")
library("bbmle")  ## for AICtab
library("ggplot2")
## cosmetic
theme_set(theme_bw() +
  theme(panel.spacing = grid::unit(0, "lines")))

The data, taken from Zuur et al. (2009) and ultimately from Roulin and Bersier (2007), quantify the number of negotiations among owlets (owl chicks) in different nests prior to the arrival of a provisioning parent as a function of food treatment (deprived or satiated), the sex of the parent, and arrival time. The total number of calls from the nest is recorded, along with the total brood size, which is used as an offset to allow the use of a Poisson response.

Since the same nests are measured repeatedly, the nest is used as a random effect. The model can be expressed as a zero-inflated generalized linear mixed model (ZIGLMM).

Various small manipulations of the data set: (1) reorder nests by mean negotiations per chick, for plotting purposes; (2) add log brood size variable (for offset); (3) rename response variable and abbreviate one of the input variables.

Owls <- transform(Owls,
  Nest = reorder(Nest, NegPerChick),
  NCalls = SiblingNegotiation,
  FT = FoodTreatment)

(If you were really using this data set you should start with summary(Owls) to explore the data set.)

We should explore the data before we start to build models, e.g. by plotting it in various ways, but this vignette is about glmmTMB, not about data visualization ...

Now fit some models:

The basic glmmTMB fit — a zero-inflated Poisson model with a single zero-inflation parameter applying to all observations (ziformula~1). (Excluding zero-inflation is glmmTMB’s default: to exclude it explicitly, use ziformula~0.)

fit_zipoisson <- glmmTMB(NCalls~(FT+ArrivalTime)*SexParent +
  offset(log(BroodSize))+(1|Nest),
  data=Owls,
  ziformula="1,
  family=poisson)

summary(fit_zipoisson)
## Family: poisson ( log )
## Formula:
## NCalls ~ (FT + ArrivalTime) * SexParent + offset(log(BroodSize)) +
##   (1 | Nest)
## Zero inflation: ~1
## Data: Owls
##
## AIC  BIC  logLik deviance df.resid
## 4015.6  4050.8  -1999.8  3999.6   591
##
## Random effects:
##
## Conditional model:
## Groups   Name Variance Std.Dev.
## Nest     (Intercept) 0.1294  0.3597
##
## Number of obs: 599, groups: Nest, 27
##
## Conditional model:
## Estimate Std. Error z value  Pr(>|z|)
## (Intercept) 2.53995   0.35656  7.123 1.05e-12 ***
## FTSatiated -0.29111   0.05961 -4.884  1.04e-06 ***
## ArrivalTime -0.06808   0.01427 -4.771  1.84e-06 ***
## SexParentMale 0.44885   0.45002  0.997   0.319
## FTSatiated:SexParentMale 0.10473   0.07286  1.437   0.151
## ArrivalTime:SexParentMale -0.02140   0.01835 -1.166   0.244
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Zero-inflation model:
## Estimate Std. Error z value  Pr(>|z|)
## (Intercept) -1.05753   0.09412 -11.24 <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

We can also try a standard zero-inflated negative binomial model; the default is the “NB2” parameterization (variance = \(\mu(1 + \mu/k)\); Hardin and Hilbe (2007)). To use families (Poisson, binomial, Gaussian) that are defined in R, you should specify them as in ?glm (as a string referring to the family function, as the family function itself, or as the result of a call to the family function: i.e. family="poisson", family=poisson, family=poisson(), and family=poisson(link="log") are all allowed and all equivalent (the log link is the default for the Poisson family). Some of the additional families that are not defined in base R (at this point nbinom2 and nbinom1) can be specified using the same format. Otherwise, for families that are implemented in glmmTMB but for which glmmTMB does not provide a function, you should specify the family argu-
ment as a list containing (at least) the (character) elements family and link, e.g. `family=list(family="nbinom2",link="log")`. (In order to be able to retrieve Pearson (variance-scaled) residuals from a fit, you also need to specify a variance component; see `?family.glmmTMB`.)

```r
fit_zinbinom <- update(fit_zipoisson,family=nbinom2)
```

Alternatively, we can use an “NB1” fit (variance = φμ).

```r
fit_zinbinom1 <- update(fit_zipoisson,family=nbinom1)
```

we should have a `getFamily` function: ideally it would also specify which are really implemented (although that’s harder), and specify default links.

Relax the assumption that total number of calls is strictly proportional to brood size (i.e. using log(brood size) as an offset):

```r
fit_zinbinom1_bs <- update(fit_zinbinom1,.
. ~ (FT+ArrivalTime)*SexParent+BroodSize+(1|Nest))
```

Every change we have made so far improves the fit — changing distributions improves it enormously, while changing the role of brood size makes only a modest (-1 AIC unit) difference:

```r
AICtab(fit_zipoisson,fit_zinbinom,fit_zinbinom1,fit_zinbinom1_bs)
```

<table>
<thead>
<tr>
<th></th>
<th>dAIC</th>
<th>df</th>
</tr>
</thead>
<tbody>
<tr>
<td>fit_zinbinom1_bs</td>
<td>0.0</td>
<td>10</td>
</tr>
<tr>
<td>fit_zinbinom1</td>
<td>1.2</td>
<td>9</td>
</tr>
<tr>
<td>fit_zinbinom</td>
<td>68.7</td>
<td>9</td>
</tr>
<tr>
<td>fit_zipoisson</td>
<td>666.0</td>
<td>8</td>
</tr>
</tbody>
</table>

### 2.1 Hurdle models

In contrast to zero-inflated models, hurdle models treat zero-count and non-zero outcomes as two completely separate categories, rather than treating the zero-count outcomes as a mixture of structural and sampling zeros.

`glmmTMB` includes truncated Poisson and negative binomial families and hence can fit hurdle models.

```r
fit_hnbinom1 <- update(fit_zinbinom1_bs,
ziformula=".",
data=Owls,
family=list(family="truncated_nbinom1",link="log"))
```
Then we can use AICtab to compare all the models.

```r
AICtab(fit_zipoisson, fit_zinbinom, fit_zinbinom1, fit_zinbinom1_bs, fit_hnbinom1)
```

<table>
<thead>
<tr>
<th></th>
<th>dAIC</th>
<th>df</th>
</tr>
</thead>
<tbody>
<tr>
<td>fit_hnbinom1</td>
<td>0.0</td>
<td>17</td>
</tr>
<tr>
<td>fit_zinbinom1_bs</td>
<td>57.8</td>
<td>10</td>
</tr>
<tr>
<td>fit_zinbinom1</td>
<td>59.0</td>
<td>9</td>
</tr>
<tr>
<td>fit_zinbinom</td>
<td>126.5</td>
<td>9</td>
</tr>
<tr>
<td>fit_zipoisson</td>
<td>723.9</td>
<td>8</td>
</tr>
</tbody>
</table>

### 3 Sample timings

To get a rough idea of glmmTMB's speed relative to lme4 (the most commonly used mixed-model package for R), we try a few standard problems, enlarging the data sets by cloning the original data set (making multiple copies and sticking them together).

Figure 1 shows the results of replicating the Contraception data set (1934 observations, 60 levels in the random effects grouping level) from 1 to 40 times. glmmADMB is sufficiently slow (∼1 minute for a single copy of the data) that we didn’t try replicating very much. On average, glmmTMB is about 2.3 times faster than glmer for this problem.

Figure 2 shows equivalent timings for the InstEval data set, although in this case since the original data set is large (73421 observations) we subsample the data set rather than cloning it: in this case, the advantage is reversed and lmer is about 5 times faster.

In general, we expect glmmTMB’s advantages over lme4 to be (1) greater flexibility (zero-inflation etc.); (2) greater speed for GLMMs, especially those with large number of “top-level” parameters (fixed effects plus random-effects variance-covariance parameters). In contrast, lme4 should be faster for LMMs (for maximum speed, you may want to check the MixedModels.jl package for Julia); lme4 is more mature and at present has a wider variety of diagnostic checks and methods for using model results, including downstream packages.

### References

Figure 1: Timing for fitting the replicated Contraception data set.
Figure 2: Timing for fitting subsets of the InstEval data set.


