An introduction to the `spaMM` package for mixed models

François Rousset

May 24, 2016

This document was compiled with R Under development (unstable) (2016-05-21 r70655) and `spaMM` 1.9.0.

The `spaMM` package fits mixed models. It was developed first to fit models with spatial correlations, which commonly occur in ecology. These correlations can be taken into account in generalized linear mixed models (GLMMs). However, there has been a dearth of validated software for making inferences under such models. This package has been first designed to fill this gap (Rousset and Ferdy, 2014). It provides likelihood-based estimates of fixed and random effect parameters, including spatial correlation parameters.

`spaMM` has been further extended to fit non-spatial model with other forms of correlation, and mixed models with non-gaussian random effects. It even provides a robust alternative function `GLM()` to the `glm()` function, suitable when the latter diverges or fails to find good starting values. It thus provides a common interface for performing different analyses currently performed by different packages or difficult to perform by other means. A drawback of this is that its algorithms are not the fastest for some (fairly common) classes of problems, but this will be improved in later versions.

This document may serve as a tutorial for using `spaMM` and will (eventually) review the methods used therein. As a first introduction, this document does not address all aspects of inference. A series of examples is first presented in order to introduce the main functions, the four basic families of responses considered (Gaussian, Poisson, binomial and Gamma), and to distinguish different types of models (spatial LMM, GLMM, and HGLM). Examples of HGLMs include the beta-binomial model and models with negative-binomial response. Later sections describe the approximations of likelihood used, introduce a fifth response family (the Conway-Maxwell-Poisson family), and provide comparisons with alternative software. In particular, a Gamma GLMM example will be used to show the meaning of the adjusted profile *h*-likelihoods (APHLs) that approximate likelihood and restricted likelihood in all models analyzed by `spaMM`, and of some other likelihood components important for the further understanding of the methods.

The following concepts are assumed at least superficially known: generalized linear models (GLM), the basic syntax of the `glm` procedure in `R`, the concept of mixed model, and formal inference using likelihood ratio tests.

Contents

1 Worked examples **3**

1.1 Gaussian model with spatial correlation . . . 3

1.1.1 Understanding and fitting the spatial model . . . 3

1.1.2 ML vs. REML . . . 8

1.1.3 Prediction . . . 9

1.2 The main procedures in `spaMM` . . . 9

1.3 Multinomial response . . . 12

1.4 GLMMs with autocorrelated random effects . . . 13

1.5 Beyond spatial GLMMs . . . 14

1.5.1 Overdispersed binomial models with crossed random effects . . . 14

1.5.2 Gamma GLMM, HGLM, and joint GLMs . . . 16

1.6 Fitting random-slope model . . . 20

2 Evaluation of the likelihood approximations **22**

2.1 Conditional and *h*-likelihood . . . 22

2.2 GLM background for likelihood and restricted likelihood approximations . . . 24

2.3 Adjusted profile likelihoods $p_v(h)$ and $p_{\beta,v}(h)$ . . . 24

3 Additional models and fitting methods **27**

3.1 The COMPoisson family . . . 27

4 Comparison with some alternative software **28**

4.1 Spatial models . . . 28

4.2 Gamma GLMM . . . 28

4.2.1 A comparison with Lee et al.'s (2011) estimates . . . 29

4.2.2 A comparison with `glmer` (and `glmmADMB`) . . . 30

4.2.3 PQL vs. `glmmPQL` . . . 33

4.3 Negative binomial model . . . 34

4.4 Random-slope model . . . 35

Bibliography **37**

Index **39**

1 Worked examples

The following sequence of examples is considered:

A linear mixed model (LMM), used to explain the input, output for the different types of parameters estimated, and the widely used Matérn spatial correlation model;

Simple spatial GLMMs, also illustrating another widely used (though more problematic) spatial correlation model, the conditional autoregressive (CAR) model;

The use of `spaMM` for non spatial models is demonstrated, using binomial and Gamma response. This section illustrates alternatives to GLMMs, such as Beta-Binomial, or Gamma-inverse Gamma HGLMs. The Gamma examples further illustrate models with a submodel for the residual variance (structured dispersion models). GLMs, in particular with structured dispersion ("joint GLMs"), can also be fitted;

The syntax and output for random-slope (or "random-coefficient") models are described.

As will be seen, the syntax is mostly similar to that of widely-used procedures such as `glm` and (`g`)`lmer`. If you are used to these syntaxes, you should note the following "silent" difference: given the formula `y~(1|grp)`, `spaMM` will fit a model with only the given random effects, in contrast to `lmer` for which `y~(1|grp)` is equivalent to `y~1+(1|grp)`.

1.1 Gaussian model with spatial correlation

1.1.1 Understanding and fitting the spatial model

We fit data from a simple Gaussian model, according to which each response value y_i is assumed to be of the form

$$y_i = \text{fix}_i + b_i + e_i \tag{1}$$

where a fixed part fix_i represents effects of known predictor variables, and $b_i + e_i$ represent two Gaussian random terms with different correlation structures:

e_i is a residual error with independent values for each observation, while b_i values can be correlated among different observations.

We first generate spatially correlated Gaussian-distributed data as follows

```r
library(MASS)

rSample <- function(nb,rho,sigma2_u,resid,intercept,slope,pairs=TRUE) {
  ## sample pairs of adjacent locations
  if (pairs) {
    x <- rnorm(nb/2); x <- c(x,x+0.001)
    y <- rnorm(nb/2); y <- c(y,y+0.001)
  } else {x <- rnorm(nb);y <- rnorm(nb)}
  dist <- dist(cbind(x,y)) ## distance matrix between locations
  m <- exp(-rho*as.matrix(dist)) ## correlation matrix
  b <- mvrnorm(1,rep(0,nb),m*sigma2_u) ## correlated random ef-
fects
  pred <- sample(nb) ##  some predictor variable
  obs <- intercept+slope*pred + b +rnorm(nb,0,sqrt(resid)) ## re-
sponse
  data.frame(obs=obs,x,y,pred=pred)
}

set.seed(123)
d1 <- rSample(nb=40,rho=3,sigma2_u=0.5,resid=0.5,intercept=-1,slope=0.1)
```

This has generated data in 2D (x, y) space with fixed effects $\text{fixe}_i = -1 + 0.1$`pred` for some predictor variable `pred`, random effect variance 0.5, residual error variance 0.5, and correlations of b_i which are exponentially decreasing with distance d as $\exp(-3d)$. Using standard notation for linear model, the fixed effects are written as $\mathbf{X}\boldsymbol{\beta}$ where $\boldsymbol{\beta} = (-1, 0.1)^\top$ and $\mathbf{X}$ is the design matrix for fixed effects, here a two-column matrix which first column is filled with 1 and the second with the variable `pred`. The random-effect variance will be denoted λ and the residual error variance will be denoted ϕ.

We fit these data using the `corrHLfit` function, as follows.

```r
library(spaMM)
HL1 <- corrHLfit(obs~pred+Matern(1|x+y),data=d1,ranFix=list(nu=0.5))
```

Here the `Matern(1|x+y)` formula term means that the Matérn correlation model is fit to the data. This is a very convenient model for spatial correlation, and includes the exponential $\exp(-\rho d)$ and the squared exponential

$\exp(-\rho d^2)$ as special cases. A conditional autoregressive (CAR) correlation model is implemented in addition to the Matérn model (see example in Section 1.4). The Matérn model is described by two correlation parameters, the scale parameter ρ, and a "smoothness" parameter ν ($\nu = 0.5$ and $\nu \to \infty$ for exponential and squared exponential models, respectively). By declaring `ranFix=list(nu=0.5)`, we have therefore fitted the model with exponential spatial correlation $\exp(-\rho d)$.

The ρ estimate together with the fixed ν are shown as `rho` and `nu` value in the output:

```
summary(HL1)

## formula: obs ~ pred + Matern(1 | x + y)
## REML: Estimation of lambda, phi and rho by REML.
##       Estimation of fixed effects by ML.
## Estimation of rho by 'outer' REML, maximizing p_bv.
## Family: gaussian ( link = identity )
##  ------- Fixed effects (beta) -------
##             Estimate Cond. SE t-value
## (Intercept) -1.07463  0.31494  -3.412
## pred         0.08593  0.01086   7.911
##  ---------- Random effects ----------
## Family: gaussian ( link = identity )
## Correlation parameters: [nu was fixed]
##       nu      rho
## 0.500000 3.160822
## Coefficients for [ lambda = var(u) ]:
##  Group        Term Estimate Cond.SE
##  x + y (Intercept)  -0.9028  0.4268
## Estimate of lambda ( x + y ):  0.4054
## # of obs: 40; # of groups: x + y, 40
##  -------- Residual variance  --------
## phi formula: "phi" ~ 1
## Coefficients for log[ phi= residual var ]
##             Estimate Cond. SE
## (Intercept)  -0.7392   0.2721
## Estimate of phi=residual var:  0.4775
##  -------- Likelihood values  --------
##                        logLik
## p_v(h) (marginal L): -50.39760
##   p_beta,v(h) (ReL): -54.60153
```

The other parameters estimated (with standard errors) are the coefficients beta of the fixed effects, the variance lambda (here σ_u^2) of the random effects, and the residual variance phi (here σ_e^2). All estimates look reasonably close to the simulated values. A confidence interval for a fixed-effect parameter should be based on a maximum-likelihood fit, hence we first perform such a fit by using HLmethod="ML":

```
HLM <- corrHLfit(obs~pred+Matern(1|x+y),data=d1,ranFix=list(nu=0.5),HLmethod="ML")
```

and then use the confint function to obtain the interval:

```
confint(HLM,"pred") ## interval for the 'pred' coefficient

## lower pred upper pred
## 0.06433277 0.10771751
```

In general there is no reason to assume a given ν value, so we fit the full Matérn model by removing the ranFix argument:

```
corrHLfit(obs~pred+Matern(1|x+y),data=d1)

## formula: obs ~ pred + Matern(1 | x + y)
## REML: Estimation of lambda, phi, nu and rho by REML.
##       Estimation of fixed effects by ML.
## Estimation of nu and rho by 'outer' REML, maximizing p_bv.
## Family: gaussian ( link = identity )
##  ------- Fixed effects (beta) -------
##             Estimate Cond. SE t-value
## (Intercept) -1.09450  0.29508  -3.709
## pred         0.08525  0.01064   8.012
##  ---------- Random effects ----------
## Family: gaussian ( link = identity )
## Correlation parameters:
##       nu      rho
## 16.66667 29.54673
## Coefficients for [ lambda = var(u) ]:
##  Group        Term Estimate Cond.SE
##  x + y (Intercept)  -0.8856  0.4259
## Estimate of lambda ( x + y ):  0.4125
## # of obs: 40; # of groups: x + y, 40
##  -------- Residual variance  --------
## phi formula: "phi" ~ 1
## Coefficients for log[ phi= residual var ]
##              Estimate Cond. SE
## (Intercept)  -0.7676   0.2723
## Estimate of phi=residual var:  0.4641
##  -------- Likelihood values  --------
##                          logLik
## p_v(h) (marginal L): -50.04269
##   p_beta,v(h) (ReL): -54.36531
```

The ν and ρ estimates now look very poor. Indeed, it is often easier to estimate $\sqrt{\nu}/\rho$ than each of these two parameters separately.

It may also be difficult to estimate the variances λ and ϕ separately, in particular if spatial correlations are weak, as noted above. Indeed, if b_i has no correlation structure, it is not separable from the residual error term e_i unless there are repeated observations in the same spatial location, because if (using traditional notation)[1] $(b_i) \sim \mathcal{N}(0, \sigma_b^2\mathbf{I})$ and $(e_i) \sim \mathcal{N}(0, \sigma_e^2\mathbf{I})$, $(b_i + e_i) \sim \mathcal{N}[0, (\sigma_b^2 + \sigma_e^2)\mathbf{I}]$ is equally well explained by any $\sigma_b^2\mathbf{I}$ and $\sigma_e^2\mathbf{I}$ such that $\sigma^2 = \sigma_b^2 + \sigma_e^2$.

To illustrate another cause for poor estimation of variances, we draw a new sample

```
set.seed(123)
d2 <- rSample(nb=40,rho=3,sigma2_u=0.5,resid=0.5,intercept=-1,
              slope=0.1,pairs=FALSE)
```

In the previous simulation we had sampled pairs of adjacent locations in space and in the new one there is no such clustering. This tends to yield poorer estimates of λ and/or ϕ:

```
corrHLfit(obs~pred+Matern(1|x+y),data=d2)

## formula: obs ~ pred + Matern(1 | x + y)
## REML: Estimation of lambda, phi, nu and rho by REML.
##       Estimation of fixed effects by ML.
## Estimation of nu and rho by 'outer' REML, maximizing p_bv.
## Family: gaussian ( link = identity )
##  ------- Fixed effects (beta) -------
##             Estimate Cond. SE t-value
## (Intercept)  -1.0635  0.27248  -3.903
## pred          0.1092  0.01123   9.726
```

[1] $X \sim \mathcal{N}(\mu, \sigma^2)$ means that X follows a gaussian distribution with given mean and variance.

```
##  ---------- Random effects ----------
## Family: gaussian ( link = identity )
## Correlation parameters:
##       nu      rho
## 16.66667 20.74381
## Coefficients for [ lambda = var(u) ]:
##  Group        Term Estimate Cond.SE
##  x + y (Intercept)   -2.003  0.5699
## Estimate of lambda ( x + y ):  0.135
## # of obs: 40; # of groups: x + y, 40
##  -------- Residual variance  --------
## phi formula: "phi" ~ 1
## Coefficients for log[ phi= residual var ]
##             Estimate Cond. SE
## (Intercept)  -0.5214   0.2506
## Estimate of phi=residual var:  0.5937
##  -------- Likelihood values  --------
##                         logLik
## p_v(h) (marginal L): -49.16688
##   p_beta,v(h) (ReL): -53.70638
```

In some cases with little information to separate ϕ and λ, the procedure may even warn that it failed to converge in a preset number of iterations.

1.1.2 ML vs. REML

By default `corrHLfit` will fit jointly the fixed effects by maximum likelihood (ML), and the random effect parameters by restricted ML (REML) to correct for small sample bias. For short, this is commonly denoted as an REML fit. The likelihood of the fitted model is here given by `p_v` and the restricted likelihood by `p_beta,v`. For linear mixed models, `p_v` is exactly the likelihood, and `p_beta,v` is exactly the restricted likelihood. For more general models, exact computation is not available, and `p_v` and `p_beta,v` are only approximations, discussed in later sections.

In all cases REML fits are not suitable for likelihood ratio (LR) tests of fixed effects. Ideally ML fits should be used, or else fits where all parameters are fitted by the same method. The function `fixedLRT` may be useful to avoid errors here. It implements different procedures for inference about fixed effects, compared by Rousset and Ferdy (2014). For example, one can test for an effect of variable `pred` by using the `fixedLRT` function, which arguments are similar two those of `corrHLfit` but which takes one formula for each of the two models compared:

```
fixlrt <- fixedLRT(obs~1+Matern(1|x+y),obs~pred+Matern(1|x+y),
          HLmethod="ML",data=d1,ranFix=list(nu=0.5))
summary(fixlrt,verbose=FALSE)

##          LR2 df       pvalue
## 1 36.89369  1 1.247491e-09
```

No such function is available for LR tests of random effect parameters. LR tests based on `p_beta,v` are suitable for LMMs, but have not been more generally ascertained, and due to the various approximations made for non-LMMs, the log LR could even be negative. An alternative approach for such tests is to use likelihood ratios from full ML fits, possibly with some bootstrap correction.

1.1.3 Prediction

The `predict` function returns the predicted value of the response, say $\mathbf{x}\hat{\boldsymbol{\beta}}+z\hat{v}_l$ for a new location l in space, where $\mathbf{x}$ are given values of the predictor variables; z is likewise some given value of the coefficient for the random effect $\hat{v}_l$ (being simply 1 in the above examples, and more generally obtained in the same way as the $\mathbf{Z}$ matrix); and $\hat{v}_l$ is the predicted value of the random effect(s) in the given location. For a spatial Gaussian effect this is the expected value of the Gaussian deviate given the inferred $\hat{\mathbf{v}}$'s in the observed locations and the covariances of the spatial process between the new location and the observed locations.

In general, prediction requires as input the new spatial coordinates, new $\mathbf{x}$ values, and new z values for each random effect if these are not trivially 1 (for block random effects, the grouping variable should thus be provided). Often one wishes to produce a nice map of predicted values without providing new $\mathbf{x}$ in every possible location (e.g. Fig. 1). See the documentation of the `filled.mapMM` function for comments on how this is achieved.

1.2 The main procedures in spaMM

We have illustrated the use of the following five functions:

`corrHLfit` can fit linear mixed models (LMM) as just shown, and it can fit GLMMs, where the random effects are gaussian and the residual variance (i.e. conditional on the realized random effects) can be Poisson, binomial, or Gamma-distributed. It can also fit models with non-gaussian random effects such as the Beta-binomial of negative-binomial

```
data(Loaloa)
lfit <- corrHLfit(cbind(npos,ntot-npos)~
           elev1+elev2+elev3+elev4+maxNDVI1+seNDVI
           +Matern(1|longitude+latitude),HLmethod="HL(0,1)",data=Loaloa,
           family=binomial(),ranFix=list(nu=0.5,rho=2.255197,lambda=1.075))
if (require(maps)) { ## required for add.map=TRUE
  filled.mapMM(lfit,add.map=TRUE,plot.axes={axis(1);axis(2)},
             decorations=quote(points(pred[,coordinates],pch=15,cex=0.3)),
             plot.title=title(main="Inferred prevalence, North Cameroon",
                              xlab="longitude",ylab="latitude"))
  }
```

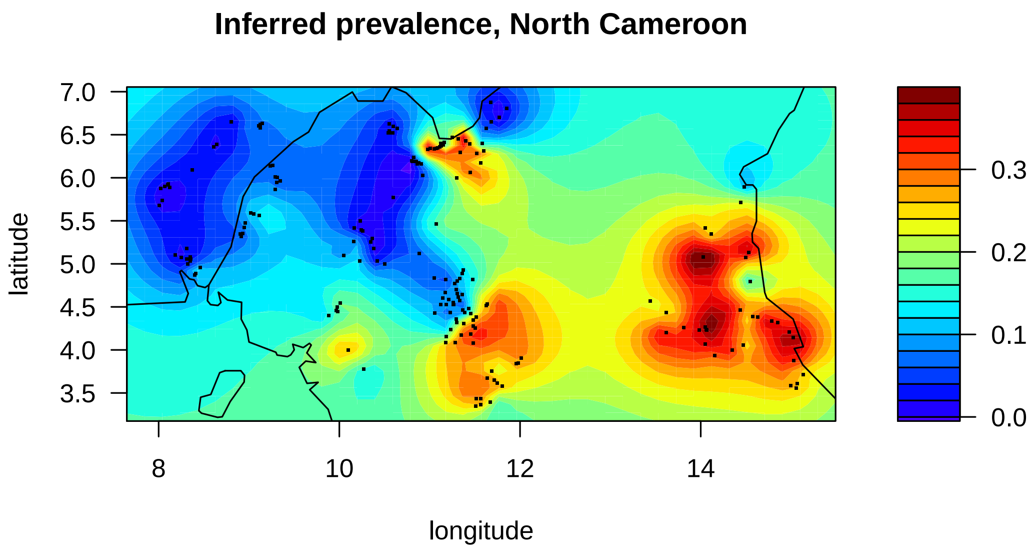

Figure 1: Plotting a map of predictions with `filled.mapMM`

(see examples below), and models which mix gaussian (possibly spatial) and non-gaussian random effects. In this way it can fit spatial models where the residual variance (conditional on realized further random effects) is Beta-Binomial, negative binomial, etc.

The `spaMM` output refers to the following formulation of all models, further illustrated in later examples. The expected response $\boldsymbol{\mu} = \mathrm{E}(\mathbf{y}|\mathbf{b})$ given all realized random effects $\mathbf{b}$ is written as the "linear predictor"

$$g(\boldsymbol{\mu}) = \boldsymbol{\eta} = \mathbf{X}\boldsymbol{\beta} + \mathbf{b} = \mathbf{X}\boldsymbol{\beta} + \mathbf{Z}\mathbf{v} \tag{2}$$

where g is the link function for the GLM response, and the structure of the random effects $\mathbf{b}$ is described in terms of a vector $\mathbf{v}$ with independent elements and of a "design matrix" $\mathbf{Z}$.[2] $\mathbf{v}$ can be further described as $\mathbf{v} = f(\mathbf{u})$ where f is another link function and the elements of $\mathbf{u}$ are independent realizations of some reference distribution (e.g., gaussian). `corrHLfit` will provide estimates of fixed-effect parameters, and of the random-effect parameters classified as dispersion parameters (the variances of u_i and of the residual error e_i) and correlation parameters affecting the elements of $\mathbf{Z}$ (ν and ρ in the previous examples).

The `HLCor` function will provide estimates of fixed effect parameters and of dispersion parameters for given correlation parameters. Only for the CAR model it also allows estimation of the correlation parameter. It acts as a wrapper for the `HLfit` function that does the same but will handle a less intuitive (and sometimes undocumented) specification of the correlations.

`HLfit` is sufficient to fit non-spatial models.

`fixedLRT` will test fixed effects. A bootstrap procedure is implemented to correct for small sample bias of the test.

`confint` will provide confidence interval for a given fixed-effect parameter, also based on profile likelihood ratio. By use of offset terms, `fixedLRT` can also be contrived to provide more general profile likelihood ratio confidence regions; and it incorporates a bootstrap procedure to correct for the small-sample bias of the test.

In addition, version 1.9.0 introduces the `fitme` function, which should eventually replace all the previous ones, and can be much faster than the

[2]Accordingly, the ith row of the expected response vector is denoted $g(\mu_i) = \eta_i = \mathbf{x}_i\boldsymbol{\beta} + b_i = \mathbf{x}_i\boldsymbol{\beta} + \mathbf{z}_i\mathbf{v}$. The i index will commonly be ignored.

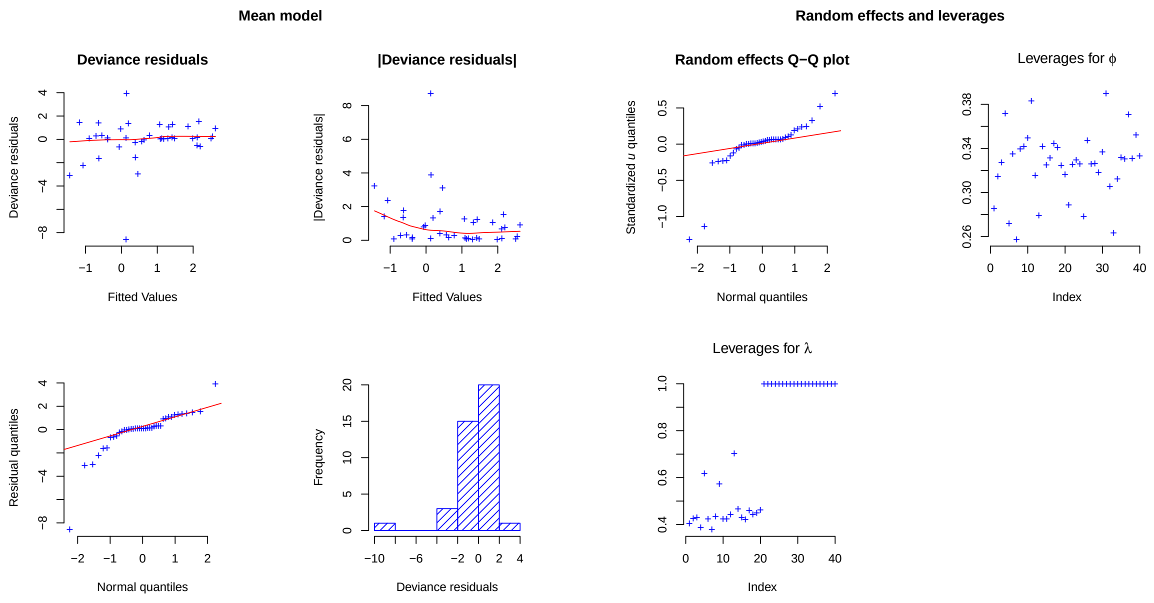

Figure 2: Diagnostic plots produced by `plot(HL1)`.

above ones, in particular for large data sets when the residual variance model is a single constant term (no structured dispersion).

The `mapMM` and `filled.mapMM` functions provide colorful plots of the predicted response. The `predict`, `simulate`, and `update` methods extend the same-named procedures from the `stats` package; an `anova` method that currently only performs a likelihood ratio test, using the results of two model fits as input; sand extractor functions `logLik`, `fitted`, `fixef`, `ranef`, `vcov` comparable to same-named functions from packages `stats` and `nlme`/`lmer`.

Diagnostic plots obtained by `plot`ting he fitted object are shown in Fig. 2. Some are similar to those returned by a GLM fit, others would require more explanation. However, the most important point is that these plots are suspect, as they may suggest that the model is wrong when it is actually true (as can be verified by simulation in binary GLMMs or Poisson GLMMs with moderate expected response values). Designing formal goodness-of-fit methods for general mixed models is a pending issue.

1.3 Multinomial response

This document does not yet gently introduces the analysis of multinomial data. Ask `?multinomial` in your R session to know how to proceed.

1.4 GLMMs with autocorrelated random effects

Non-Gaussian response data can be fitted by combining the `Matern` formula term together with syntax used in other procedures such as `glm` or `glmer`. For example binomial data can be fit by

```
data(Loaloa) ## parasite prevalence data in North Cameroon
binfit <- HLCor(cbind(npos,ntot-npos)~
                  1+Matern(1|longitude+latitude),data=Loaloa,
      family=binomial(),ranPars=list(nu=0.5,rho=1/0.7))
```

using the two-column response format `cbind(npos,ntot-npos)` for binomial data.

The following classical toy example for Poisson-distributed response further introduces a conditional autoregressive (CAR) correlation model. The data describe lip cancer incidence in different Scottish districts (but we do not really care about the details). The model for the logarithm of expectation of the response is

$$\ln(\mu_i) = \ln(a_i) + \beta_1 + \beta_2 x_i/10 + b_i \tag{3}$$

(Clayton and Kaldor, 1987; Breslow and Clayton, 1993), where $\ln(a_i)$ is an offset that describes the effect of population size and of some other variables, not included in the statistical model, on the Poisson mean; x_i is the variable `prop.ag` below; and b_i is a Gaussian random effect.

For the b_is, the CAR model considers a correlation matrix $(\mathbf{I} - \rho\mathbf{N})^{-1}$ where $\mathbf{N}$ is an adjacency matrix between the different districts (a matrix with elements 1 if the districts are adjacent and 0 otherwise), here provided as `NMatrix` included in `data(scotlip)`. The rows of the matrix correspond to the `gridcode` variable in the data. A full fit including estimation of ρ is then given by[3]

```
data(scotlip)
lipfit <- HLCor(cases~I(prop.ag/10)+adjacency(1|gridcode)
                  +offset(log(scotlip$expec)),
          data=scotlip,family=poisson(),adjMatrix=Nmatrix)
```

[3]Up to version 1.5.1, this model could be fitted only using `corrHLfit` (still feasible). The older method was much slower. Further, the results by these different methods differ (although they are here very similar). `corrHLfit` will by default maximize with respect to correlation parameters the restricted likelihood of joint ML/REML fits of fixed-effect and dispersion parameters, while `HLCor` estimates all parameters by alternating ML extimation of fixed effects for given random-effect parameter estimates, and REML estimation of all random-effect parameters for given fixed-effects estimates.

The results are very close to those of Lee and Lee (2012):

```
summary(lipfit)

## formula: cases ~ I(prop.ag/10) + adjacency(1 | gridcode) + offset(log(scotlip$expec
## Estimation of lambda by REML approximation (p_bv).
## Estimation of fixed effects by ML approximation (p_v).
## Family: poisson ( link = log )
##  ------- Fixed effects (beta) -------
##                Estimate Cond. SE t-value
## (Intercept)      0.2377   0.2078   1.144
## I(prop.ag/10)    0.3763   0.1218   3.090
##  ---------- Random effects ----------
## Family: gaussian ( link = identity )
## Correlation parameters:
##       rho
## 0.1740116
## Coefficients for inverse[ lambda_i =var(V'u) ], with:
##     Group        Term Estimate Cond.SE
##  gridcode (Intercept)    6.460   1.716
##  gridcode        adjd   -1.124   0.301
## Estimate of rho ( gridcode CAR):  0.174
## Estimate of lambda factor ( gridcode CAR):  0.1548
## # of obs: 56; # of groups: gridcode, 56
##  -------- Likelihood values  --------
##                          logLik
## p_v(h) (marginal L): -161.5141
##   p_beta,v(h) (ReL): -163.6783
## lambda leverages numerically 1 were replaced by 1 - 1e-8
```

1.5 Beyond spatial GLMMs

Models with arbitrary fixed correlation matrix of random effects can be fitted using the `corrMatrix` argument of the `HLCor` function. This section further details various models where the correlation structure of random effects is specified by the usual `(<formula terms>|<grouping variable>)` syntax.

1.5.1 Overdispersed binomial models with crossed random effects

One can fit Binomial GLMMs using `lme4`:

```
data(salamander)
library(lme4)
glfit <- glmer(cbind(Mate,1-Mate)~TypeF+TypeM+TypeF*TypeM+(1|Female)
              +(1|Male),family=binomial(),data=salamander)
```

and this can be done with `HLfit`:

```
hlfit <- HLfit(cbind(Mate,1-Mate)~TypeF+TypeM+TypeF*TypeM+(1|Female)
              +(1|Male),family=binomial(),data=salamander,HLmethod="ML")
```

The input syntax for both procedures are exactly the same, except that `HLfit` will perform an REML fit if `HLmethod="ML"` is not specified. The results (not shown here) are also very close.

`HLfit` can also fit a Beta-binomial model.[4] As a binomial GLMM, this model assumes that the response follows a binomial distribution with expectation p given conditionally on a realized random effect v

$$\text{logit}(p) = \ln \frac{p}{1-p} = \mathbf{x}\boldsymbol{\beta} + \mathbf{z}\mathbf{v} \tag{4}$$

(assuming the default logit link of the binomial GLM family). But it also assumes that

$$v = \text{logit}(u) = \ln \frac{u}{1-u} \tag{5}$$

where the elements of u are independent Beta-distributed, and where the logit is also the default link for Beta-distributed random effects. Thus, if there are no fixed effects, $p = u$ has a Beta distribution. Since v is not gaussian, this is not a GLMM, but what Lee and Nelder (1996) called a hierarchical GLM (HGLM).

We consider classical seed germination data as a toy example for Beta-binomial fits. For comparison with the results of Lee and Nelder (1996), we fit the model to these data by the method `HL(0,0)` (slightly cryptic at this step of the documentation):

```
data(seeds)
HLfit(cbind(r,n-r)~seed*extract+(1|plate),family=binomial(),
  rand.family=Beta(),HLmethod="HL(0,0)",data=seeds)

## formula: cbind(r, n - r) ~ seed * extract + (1 | plate)
## Estimation of lambda by REML approximation (p_bv).
```

[4]see the Documentation of the `bbmle` package for a list of packages that consider the Beta-Binomial model. I have not tried them.

```
## Estimation of fixed effects by h-likelihood approximation.
## Family: binomial ( link = logit )
##  ------- Fixed effects (beta) -------
##                          Estimate Cond. SE t-value
## (Intercept)              -0.54259   0.1864 -2.9102
## seed073                   0.08003   0.3027  0.2644
## extractCucumber           1.33682   0.2643  5.0579
## seed073:extractCucumber -0.82202   0.4218 -1.9487
##  ---------- Random effects ----------
## Family: beta ( link = logit )
## Coefficients for log[ lambda = 4 var(u)/(1 - 4 var(u)) ]:
##  Group        Term Estimate Cond.SE
##  plate (Intercept)   -3.799  0.5381
## Estimate of lambda ( plate ):  0.02239
## # of obs: 21; # of groups: plate, 21
##  -------- Likelihood values  --------
##                          logLik
##        h-likelihood: -42.16218
## p_v(h) (marginal L): -54.00643
##   p_beta,v(h) (ReL): -56.60453
```

The fixed effects estimates are those of Lee and Nelder (1996). The present parametrization of the Beta distribution is that of Lee and Nelder (2001) as discussed by Lee et al. (2006, p. 181), so that **HLfit**'s λ is $1/(2\alpha)$ for α as shown in Lee and Nelder (1996). The λ and α estimates are then seen to be approximately equivalent. Lee and Nelder (1996) also present a GLMM fit of these data, which is also similarly consistent with the GLMM fit by **HLfit** (not shown).

1.5.2 Gamma GLMM, HGLM, and joint GLMs

This example, derived from Lee et al. (2011), illustrates a Gamma GLMM model with a log link, that is $\boldsymbol{\eta} = \ln(\boldsymbol{\mu}) = \mathbf{X}\boldsymbol{\beta} + \mathbf{Z}\mathbf{v}$ where $\mathbf{v}$ is normally distributed.[5] A notable feature is that there is a linear predictor for the (log

[5]They call the Gamma GLMM with log link the Gamma-lognormal model. They appear to view this model as Gaussian $v = \ln(u)$ for u being lognormal, and to use the distribution of u as a basis for the name of the model (thus the "log" here comes from the $u \mapsto v$ link, not from the response link $\mu \mapsto \eta$). This is ambiguous if the link between u and v is not specified, as we can equally describe this model as a Gamma-Normal model (with identity $u \mapsto v$ link).

This suggests that the semantics for HGLMs should be revised, and for example be based on the distribution of v so that different names for the same distribution cannot result from different specifications of u. In principle the link for the response should be of) the variance of the residual error. There are only batch random effects (which specification determine the elements of **Z**), without any autocorrelated process, so the HLfit function is sufficient to analyze these data.

This example deals with data about semiconductor materials ("wafers") from Robinson et al. Subject-matter details are ignored here; three variables denoted X1, X2 and X3 were experimentally varied. A fixed-effect model for the residual variance ("structured dispersion model") was also considered. This model can be fit by

```
data(wafers)
HLg <- HLfit( y ~ X1+X2+X3+X1*X3+X2*X3+I(X2^2)+(1|batch),
             family=Gamma(log),
             resid.formula = ~ X3+I(X3^2) ,data=wafers)
summary(HLg)

## formula: y ~ X1 + X2 + X3 + X1 * X3 + X2 * X3 + I(X2^2) + (1 | batch)
## Estimation of lambda and phi by REML approximation (p_bv).
## Estimation of fixed effects by ML approximation (p_v).
## Family: Gamma ( link = log )
##  ------- Fixed effects (beta) -------
##             Estimate Cond. SE t-value
## (Intercept)  5.55514  0.05450 101.922
## X1           0.08376  0.02397   3.494
## X2          -0.20861  0.02397  -8.703
## X3          -0.13729  0.03786  -3.626
## I(X2^2)     -0.07641  0.02023  -3.778
## X1:X3       -0.09181  0.04019  -2.284
## X2:X3       -0.08686  0.04019  -2.161
##  ---------- Random effects ----------
## Family: gaussian ( link = identity )
## Coefficients for [ lambda = var(u) ]:
##  Group        Term Estimate Cond.SE
##  batch (Intercept)   -3.688  0.4891
## Estimate of lambda ( batch ):  0.02502
## # of obs: 198; # of groups: batch, 11
##  -------- Residual variance  --------
```

specified although it is usually ignored when it is the canonical link of the GLM (which is not the case for the Gamma examples). According to this logic the Gamma-inverse Gamma model becomes the Gamma-log inverse Gamma (GLInG?) HGLM with log link, the Beta binomial (with canonical link for response) becomes the Binomial logit-Beta (BLoB?), and the usual Binomial GLMM becomes the Binomial logit-normal model (as in Coull and Agresti, 2000). A BLInG HGLM may not look like serious stuff, but it can be fitted...

```
## phi formula: "phi" ~ X3 + I(X3^2)
## Coefficients for log[ phi= scale param. ]
##              Estimate Cond. SE
## (Intercept)  -2.8958   0.1384
## 3             0.1103   0.1142
## I(X3^2)       0.9468   0.1134
##  -------- Likelihood values  --------
##                         logLik
## p_v(h) (marginal L): -1157.609
##   p_beta,v(h) (ReL): -1175.199
```

A gamma-inverse Gamma model was also considered by Lee et al. (2011). Here the log of the expectation of the Gamma response has the form $\boldsymbol{\eta} = \ln(\boldsymbol{\mu}) = X\boldsymbol{\beta} + \mathbf{v} = \mathbf{X}\boldsymbol{\beta} + \ln(\mathbf{u})$ where u has an inverse-Gamma distribution. v being non-Gaussian, this is an HGLM.

```
HLfit( y ~X1+X2+X1*X3+X2*X3+I(X2^2)+(1|batch),
          family=Gamma(log),rand.family=inverse.Gamma(log),
          resid.formula= ~ X3+I(X3^2) ,data=wafers)

## formula: y ~ X1 + X2 + X1 * X3 + X2 * X3 + I(X2^2) + (1 | batch)
## Estimation of lambda and phi by REML approximation (p_bv).
## Estimation of fixed effects by ML approximation (p_v).
## Family: Gamma ( link = log )
##  ------- Fixed effects (beta) -------
##              Estimate Cond. SE t-value
## (Intercept)  5.56854  0.05417 102.794
## X1           0.08373  0.02396   3.494
## X2          -0.20860  0.02396  -8.706
## X3          -0.13735  0.03786  -3.628
## I(X2^2)     -0.07637  0.02022  -3.778
## X1:X3       -0.09194  0.04019  -2.287
## X2:X3       -0.08683  0.04019  -2.160
##  ---------- Random effects ----------
## Family: inverse.gamma ( link = log )
## Coefficients for log[ lambda = var(u)/(1 + var(u)) ]:
##  Group        Term Estimate Cond.SE
##  batch (Intercept)   -3.684  0.4879
## Estimate of lambda ( batch ):  0.02513
## # of obs: 198; # of groups: batch, 11
##  -------- Residual variance  --------
## phi formula: "phi" ~ X3 + I(X3^2)
## Coefficients for log[ phi= scale param. ]
##              Estimate Cond. SE
## (Intercept)  -2.8969   0.1384
## 3             0.1094   0.1141
## I(X3^2)       0.9479   0.1134
##  -------- Likelihood values  --------
##                         logLik
## p_v(h) (marginal L): -1157.523
##   p_beta,v(h) (ReL): -1175.121
```

Lee et al. (2011) also fit GLMs and GLMs with structured dispersion models (known as joint GLMs) to these data. These models can all be fit by `HLfit`. A joint GLM in particular is fit by

```
HLfit( y ~X1+X2+X1*X3+X2*X3+I(X2^2),family=Gamma(log),
      resid.formula= ~ X3+I(X3^2) ,data=wafers)

## formula: y ~ X1 + X2 + X1 * X3 + X2 * X3 + I(X2^2)
## Estimation of phi by REML approximation (p_bv).
## Estimation of fixed effects by ML.
## Family: Gamma ( link = log )
##  ------- Fixed effects (beta) -------
##              Estimate Cond. SE t-value
## (Intercept)  5.57570  0.03131 178.078
## X1           0.08375  0.02795   2.996
## X2          -0.21036  0.02795  -7.526
## X3          -0.13261  0.04019  -3.299
## I(X2^2)     -0.08017  0.02440  -3.286
## X1:X3       -0.09247  0.04383  -2.110
## X2:X3       -0.08201  0.04383  -1.871
##  -------- Residual variance  --------
## phi formula: "phi" ~ X3 + I(X3^2)
## Coefficients for log[ phi= scale param. ]
##              Estimate Cond. SE
## (Intercept)  -2.5119   0.1340
## 3             0.1589   0.1136
## I(X3^2)       0.7366   0.1125
##  -------- Likelihood values  --------
##                         logLik
## p(h)   (Likelihood): -1170.187
##   p_beta(h)   (ReL): -1170.187
```

All these fits are by default REML fits: the argument `HLmethod="ML"` must again be used to perform ML fits. Results from these different fits of the same data are similar to published ones. In the GLM case, the `HLfit` results are quite consistent with `glm` ones (provided the correct `HLmethod` is used in the comparison) and it is easy to check analytically that the likelihood values returned by `HLfit` are more accurate than published ones.

1.6 Fitting random-slope model

A commonly considered random-slope model is a model with the following structure:

$$\boldsymbol{\eta} = \mathbf{1}\beta_{\mathrm{I}} + \mathbf{b}_{\mathrm{I}} + \mathbf{x}_{\mathrm{S}}(\beta_{\mathrm{S}} + \mathbf{b}_{\mathrm{S}}). \tag{6}$$

The distinctive term is here $\mathbf{x}_{\mathrm{S}}\mathbf{b}_{\mathrm{S}}$ as the remainder is of the same form already considered e.g. in eq 3. The additional term means that the "slope" of the regression (the coefficient of the design variable $\mathbf{x}_{\mathrm{S}}$) is random, including the random effect $\mathbf{b}_{\mathrm{S}}$. Hence, there are two realized random effects $b_{\mathrm{I,g}}$ and $b_{\mathrm{S,g}}$ for each level g of the grouping factor. Random-slope models allow each such pair to be correlated, which is the main specificity in fitting these models.

`spaMM` can fit such models although their current implementation is computationally inefficient. The syntax for such random effects is
`(<model term>|<grouping factor>)`,
where `<model term>` gives the explanatory variable $\mathbf{x}_{\mathrm{S}}$, as in
`HLfit(y ~X1+(X1|batch),data=wafers)`. If you want to ignore the correlation (which is often warned against), use two terms as in
`(1|batch)+(X1-1|batch)`; if you further want a random effect on the slope only, consider only the term `(X1-1|batch)` or `(0+X1|batch)`; in all of these cases the syntax is the same as for a fit by `lmer` and is consistent with standard syntax for `formulas` (`lmer` has the additional syntax `(X1||batch)` for models without correlation). In general, different variables can be considered in the fixed and random part, as in
`HLfit(y ~X1+(X2|batch),data=wafers)`.

The output from such models requires careful consideration. Suppose we fit

```
HLfit(y ~X1+(X2|batch),data=wafers)

## formula: y ~ X1 + (X2 | batch)
## REML: Estimation of lambda and phi by REML.
##       Estimation of fixed effects by ML.
## Family: gaussian ( link = identity )
##  ------- Fixed effects (beta) -------
##              Estimate Cond. SE t-value
## (Intercept)   224.92   12.316  18.263
## X1             23.68    9.325   2.539
##  ---------- Random effects ----------
## Family: gaussian ( link = identity )
## Coefficients for [ lambda = var(u) ]:
##  Group        Term Estimate  Cond.SE Var. Corr.
##  batch (Intercept)    8.435    0.467 2760
##  batch          X2  -10.993 2766.956 1844    -1
## Estimate of lambda ( batch ):  4603
## # of obs: 198; # of groups: batch, 22
##  -------- Residual variance  --------
## phi formula: "phi" ~ 1
## Coefficients for log[ phi= residual var ]
##              Estimate Cond. SE
## (Intercept)    9.478   0.1035
## Estimate of phi=residual var:  13060
##  -------- Likelihood values  --------
##                         logLik
## p_v(h) (marginal L): -1228.720
##   p_beta,v(h) (ReL): -1222.139
```

As in spatial models, correlated Gaussian random effects are represented as $\mathbf{b} = \mathbf{L}\mathbf{v}$ where the elements of $\mathbf{v}$ are uncorrelated. The `Var.` column gives the variances of the correlated effects, $\mathbf{b} = \mathbf{L}\mathbf{v}$, which is what `lmer` appears to report both as `Variance` and as `Std.Dev.` for random effects. The correlation coefficient for the "intercept" and "slope" effects is the `Corr` on the right of the random effect output (here as single `-1` value; more generally a lower triangular block when more than two random effects are possibly correlated.

The above random-effect output also contains information more directly comparable to the output of other models. The `Estimate`s are the log variances for uncorrelated $\mathbf{v}$ (and associated `Cond. SE`s). However, it is unclear how far this output is useful, since there is no unique representation of $\mathbf{b}$ as $\mathbf{L}\mathbf{v}$. In the present case, the covariance matrix of $\mathbf{b}$ is represented in terms of its eigensystem, as $\mathbf{C_b} = \mathbf{L}\mathbf{\Lambda}\mathbf{L}'$ where $\mathbf{L}$ contains normed eigenvectors and $\mathbf{\Lambda}$ is the diagonal matrix of eigenvalues. Thus $\mathbf{b} = \mathbf{L}\mathbf{v}$ where the variances of $\mathbf{v}$ are these eigenvalues. Assigning these uncorrelated random effects to the intercept and the slope is a conceptually strained exercise and most users might wish to ignore this part of the output.[6]

[6]A further snag is that for any given ordering of the eigenvectors in $\mathbf{L}$ and for any permutation matrix $\mathbf{P}$, $\mathbf{L}\mathbf{v}$ can be written as $(\mathbf{L}\mathbf{P})(\mathbf{P}^\top\mathbf{v})$ in terms of the permuted design

2 Evaluation of the likelihood approximations

A typical fit will provide approximations $p_v(h)$ (`p_v`) for the likelihood and $p_{\beta,v}(h)$ (`p_bv`) for the restricted likelihood. These approximations are known as APHLs (adjusted profile h-likelihoods). Based on the Gamma GLMM example, this section explains the meaning and illustrates the computation of these approximations, which are exact for LMMs. They are of the form

$$p_v(h) = h - 0.5 \ln \left| -\frac{1}{2\pi} \frac{\partial^2 h}{\partial \mathbf{v}' \partial \mathbf{v}} \right| . \tag{7}$$

and

$$p_{\beta,v}(h) = h - 0.5 \ln \left| -\frac{1}{2\pi} \frac{\partial^2 h}{\partial (\boldsymbol{\beta}, \mathbf{v})' \partial (\boldsymbol{\beta}, \mathbf{v})} \right| . \tag{8}$$

where for a given matrix $\mathbf{H}$, $\ln |\mathbf{H}|$ is the logarithm of the absolute value of its determinant (the “logdet”); and the definition of h, as well as the computation of the two distinct $\mathbf{H}$ matrices involved in $p_v(h)$ and $p_{\beta,v}(h)$, will now be explained.

For concreteness we consider the following fit:

```
HLgs <- HLfit( y ~X1*X3+X2*X3+I(X2^2)+(1|batch),
               family=Gamma(log),data=wafers)
```

where for simplicity there is no modeling of the residual variance of the random effects.

2.1 Conditional and h-likelihood

The log-likelihood for each independent draw y_k of a Gamma GLM can be written

$$c(\mu_k, \nu; y_k) = \nu \ln(\nu y_k / \mu_k) - \nu (y_k / \mu_k) - \ln(\Gamma(\nu)) - \ln(y_k) \tag{9}$$

where μ_k is the expected value (here conditional on the realized random effect), and $1/\nu$ is the variance of the residual term. Thus the conditional likelihood of the data given the realized random effects is

matrix $\mathbf{LP}$ and permuted independent random effects $\mathbf{P}^\top \mathbf{v}$, so that each $\mathbf{P}$ provides a statistically equivalent fit but a different assignment of $\mathbf{v}$ elements to intercept and slope. However, `HLfit` chooses a permutation so as to maintain consistency between the output of models with and without correlation when the correlation vanishes, and to maintain consistency among the different descriptors of variance on each row in the same condition.

```
 mui <- HLgs$fv
 nu <- 1/HLgs$phi
 clik <- with(wafers,sum(nu*log(nu*y/mui)-nu*(y)/mui-log(gamma(nu))-log(y)))
 clik

## [1] -1180.896
```

The h-(log-)likelihood is defined as the sum of this term and of the log-likelihood of the random effects:

$$h(\boldsymbol{\mu}, \nu, \lambda; \mathbf{y}, \mathbf{v}) = \sum_k c(\mu_k, \nu; y_k) + \sum_i \ln(L(v_i)) \tag{10}$$

where the sum over k is over all levels of the response variable and the sum over i is over all levels of the random effect. The random effects are Gaussian with identity link, $u = v$, and dispersion λ:

$$\ln(L(v_i)) = -\frac{1}{2}\left(\frac{v_i^2}{\lambda} + \ln(2\pi\lambda)\right), \tag{11}$$

that is

```
 hlik<-clik+with(HLgs,sum(-(ranef^2)/(2*lambda)
                            -(log(2*pi*lambda))/2))
 hlik

## [1] -1173.457
```

Here the sum is over the 11 values of v =`ranef(HLgs)`.
`clik` and `hlik` are hidden in the output object of the fit:

```
HLgs$APHLs[c("clik","hlik")]

## $clik
## [1] -1180.896
##
## $hlik
## [1] -1173.457
```

We now need some preparation to understand the computation of the logdet terms in $p_v(h)$ and $p_{\beta,v}(h)$.

2.2 GLM background for likelihood and restricted likelihood approximations

We refer to standard notation for a GLM (McCullagh and Nelder, 1989, eq. 2.4). The likelihood of an observation is written in the form

$$L(y;\theta,\phi)=\exp\{[y\theta-b(\theta)]/a(\phi)+c(y,\phi)\}. \tag{12}$$

Three quantities are distinguished: θ, "the canonical parameter", which is what factors with y; μ, the expectation of y, and the linear predictor $\eta=\sum_j x_j\beta_j$. The assumed relationship between $\eta=g(\mu)$ defines the link g used, while the relationship between θ and μ defines the "canonical link".

In a Gamma GLM, $a(\phi)=\phi=1/\nu$, $\theta=-1/\mu$ (canonical link), $b(\theta)=\ln(\mu)=-\ln(-\theta)$ (McCullagh and Nelder, 1989, p. 30). The variance of Y is $b''(\theta)a(\phi)=\phi\mu^2$.

The gradient of the log-likelihood l can be written in the form

$$\frac{\partial l}{\partial\beta_p}=\frac{\partial l}{\partial\theta}\frac{\partial\theta}{\partial\eta}\frac{\partial\eta}{\partial\beta_p} \tag{13}$$

We consider the Hessian matrix of l with respect to fixed-effect parameters, i.e. the matrix which prth element is $\partial^2 l/\partial\beta_p\partial\beta_r$. From eq. 13, it involves either $\partial^2 l/\partial\theta\partial\beta_r$, $\partial^2\theta/\partial\eta\partial\beta_r$, or $\partial^2\eta/\partial\beta_p\partial\beta_r$. The last term is null since η is linear, the second one is exactly null if the link is canonical ($\eta=\theta$), and the first one is $-\partial\mu/\partial\beta_r$.

If the link is not canonical, the second term is not null, so it should be either computed or approximated. The Hessian is generally approximated by its expectation, given by

$$\mathrm{E}\left[\frac{\partial^2 l}{\partial\beta_p\partial\beta_r}\right]=\mathrm{E}\left[\frac{\partial^2 l}{\partial\theta\partial\beta_r}\frac{\partial\theta}{\partial\beta_p}+\frac{\partial l}{\partial\theta}\frac{\partial^2\theta}{\partial\beta_p\partial\beta_r}\right]. \tag{14}$$

Upon sampling for given μ, $\mathrm{E}[\partial l/\partial\theta]=\mathrm{E}[y-\mu]=0$ hence the second term is null. Therefore, we find that the second term of the expected Hessian is zero, and this can be used as an approximation for the equivalent term of the realized Hessian in the case of a non-canonical link.

2.3 Adjusted profile likelihoods $p_v(h)$ and $p_{\beta,v}(h)$

We can now compute the elements of the matrices of which the "logdet" is required to compute the APHLs p_v and $p_{\beta,v}$.[7]

[7] Usefully detailed computations are also presented by Molas and Lesaffre (2010) for a Poisson "hurdle" HGLM. Our computations differ from theirs as we have only one random

As in a GLM, we can write

$$\frac{\partial h}{\partial \beta_p} = \nu \sum_{ij} (y_{ij} - \mu_{ij}) w_{ij} \frac{\partial \eta_{ij}}{\partial \mu_{ij}} \frac{\partial \eta_{ij}}{\partial \beta_p} \tag{15}$$

$$\frac{\partial h}{\partial v_k} = \nu \sum_{ij} (y_{ij} - \mu_{ij}) w_{ij} \frac{\partial \eta_{ij}}{\partial \mu_{ij}} \frac{\partial \eta_{ij}}{\partial v_k} - \frac{v_k}{\lambda} \tag{16}$$

where $\eta_{ij} = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3 + \beta_{13} x_1 x_3 + \beta_{23} x_2 x_3 + \beta_{22} x_2^2 + z_{ij,k} v_k$ is the linear predictor for the "mean" part and w_{ij} is the diagonal element of the weight matrix

$$\mathbf{W} \equiv \operatorname{diag} \left(\frac{\partial \mu_{ij}}{\partial \eta_{ij}} \right)^2 / [b''(\theta)]. \tag{17}$$

Here with a log link, $\partial \mu / \partial \eta = \partial \mu / \partial \ln(\mu) = \mu$, $w_{ij} = 1$ and $\mathbf{W} = \mathbf{I}$.

The design matrix for random effects has elements here $z_{k,i} = \delta_{\text{batch}(k),i}$ for observation k. The design matrix for fixed effects has elements denoted $x_{ij,p}$.

We consider a log link ($\eta = \ln(\mu)$), so the link is not canonical ($\eta \neq \theta$) and we use the expected Hessian approximation to the observed Hessian, as explained in Section 2.2. In particular for the derivatives with respect to β_r of the different factors in (15), we again note that the last factor has a null derivative, and that the derivative of the middle ones can ignored when the $(y - \mu)$ is approximated by its null expectation. Thus we only consider the remaining derivative

$$\frac{\partial \eta_{ij}}{\partial \mu_{ij}} \frac{\partial (y_{ij} - \mu_{ij})}{\partial \beta_r} = - \frac{\partial \eta_{ij}}{\partial \mu_{ij}} \frac{\partial \mu_{ij}}{\partial \beta_r} = - \frac{\partial \eta_{ij}}{\partial \beta_r} = -x_{ij,r}. \tag{18}$$

Hence the nonzero elements of the expected Hessian matrix are

$$\mathrm{E} \frac{\partial^2 h}{\partial \beta_p \partial \beta_r} = -\nu \sum_{ij} x_{ij,p} \frac{\partial \eta_{ij}}{\partial \beta_p} = -\nu \sum_{ij} x_{ij,p} x_{ij,r}, \tag{19}$$

$$\mathrm{E} \frac{\partial^2 h}{\partial \beta_p \partial v_k} = -\nu \sum_{ij} x_{ij,p} z_{ij,k}, \text{ and} \tag{20}$$

$$\mathrm{E} \frac{\partial^2 h}{\partial v_k^2} = -\nu \sum_{ij} z_{ij,k}^2 - 1/\lambda. \tag{21}$$

effect in the mean part and the model is Gamma (with a dispersion parameter $\phi \equiv 1/\nu$) rather than Poisson (without any overdispersion parameter). We do not need their correction term $M(\theta_{ijk})$ for truncation of the Poisson distribution.

Only the derivatives with respect to ∂v_k^2 are involved in the APHL approximation to the marginal likelihood of the fitted model, which is

$$p_v(h) = h - 0.5 \ln \left| -\frac{1}{2\pi} \frac{\partial^2 h}{\partial \mathbf{v}' \partial \mathbf{v}} \right|. \tag{22}$$

Here the Hessian matrix is diagonal, so the logdet term is simple to compute:

```
p_v <- hlik-(sum(log((nu*as.numeric(table(wafers$batch))
                      +1/HLgs$lambda)/(2*pi))))/2
p_v

## [1] -1191.096
```

but a more generally applicable code is[8]

```
designZ <- apply(matrix(1:11,ncol=1),1,
                 function(v) {as.numeric(wafers$batch==v)})
crossprods <- apply(designZ,2,function(v) {t(v)%*%designZ})
hess <- - nu[1]*crossprods -diag(rep(1/HLgs$lambda,11))
hlik-determinant(hess/(2*pi))$modulus[[1]]/2

## [1] -1191.096
```

Finally, the APHL used to estimate the dispersion parameter λ depends on the Hessian for both the parameters of the fixed effects and for $\mathbf{v}$:

$$p_{\beta,v}(h) = h - 0.5 \ln \left| -\frac{1}{2\pi} \frac{\partial^2 h}{\partial(\boldsymbol{\beta}, \mathbf{v})' \partial(\boldsymbol{\beta}, \mathbf{v})} \right|. \tag{23}$$

Using brute computation rather than any simplification of the determinant:

```
designZ <- apply(matrix(1:11,ncol=1),1,
                 function(v) {as.numeric(wafers$batch==v)})
## the joint design matrix for fixed and random effects:
designXZ <- with(wafers,as.matrix(cbind(1,X1,X2,X3,X2^2,
                X1*X3,X2*X3,designZ)))
```

[8]`determinant(.)$modulus` directly returns the logarithm and avoids numerical overflows

```
## sum-of-squares-and-products matrix
crossprods<-apply(designXZ,2,function(v) {t(v)%*%designXZ})
hess <- - nu[1]*crossprods -diag(c(rep(0,7),rep(1/HLgs$lambda,11)))
p_bv <- hlik-determinant(hess/(2*pi))$modulus[[1]]/2
p_bv

## [1] -1207.51
```

We have now recovered all four likelihood components of the Gamma GLMM fit.

3 Additional models and fitting methods

3.1 The COMPoisson family

The Conway-Maxwell-Poisson family for count data is a generalization of the Poisson family that can describe over- and underdispersion, relative to Poisson response (e.g., Shmueli et al., 2005). The quasi-poisson method available in the `MASS` package is often used for such purposes, but it is not based on a probability model for count data. Overdispersion can also be represented by mixed models, but underdispersion in count data is less easy to represent, and the COMPoisson family is of particular interest in the latter case. The distribution of a response y is

$$\Pr(y; \lambda, \nu_{\text{CMP}}) = \frac{\lambda^y}{(y!)^{\nu_{\text{CMP}}} Z(\lambda, \nu_{\text{CMP}})} \tag{24}$$

where $Z(\lambda, \nu_{\text{CMP}}) := \sum_{k=0}^{\infty} \lambda^k/(k!)^{\nu_{\text{CMP}}}$. The Poisson distribution is recovered for $\nu_{\text{CMP}} = 1$, in which case $Z = e^\lambda$, which also happens to be the mean μ of the distribution. However, for $\nu_{\text{CMP}} \neq 1$, the mean is not e^λ.

It is a probability model of the form that can be fitted by `glm`. Examples using `glm` are shown in the `spaMM` documentation for this family (see `help("COMPoisson")`), and `spaMM` can also fit mixed models with this response family. Its main drawback is that the Z function has no expression in terms of standard "elementary" (efficiently implemented) functions, and involves an infinite summation that must be approximated by truncation. The number of terms required for accurate evaluation increases with decreasing ν_{CMP}. Further, the inverse function of Z (needed fo fitting by `glm`) has no explicit expression. Altogether, this implies that fitting the COMPoisson model can be relatively slow (and perhaps inaccurate) for highly overdispersed data (or, more precisely, for highly overdispersed conditional response).[9] However, it is easier to fit models on underdipersed ($\nu_{\text{CMP}} > 1$) conditional responses.

4 Comparison with some alternative software

In the following we compare the spaMM output to the output of some related packages and to a few literature results.

4.1 Spatial models

The widely used lme4 package may be thorough in many ways but does not fit spatial GLMMs. Some tricks commonly used to constrain the functions lmer, and glmmPQL (from MASS), to analyse spatial models are discussed in Rousset and Ferdy (2014) (in particular, in Appendix G, independently available here). In summary, they should be avoided.

Some packages based on stochastic algorithms (typically, MCMC), such as geoRglm, can fit spatial models but have not been thoroughly assessed for such applications. MCMC methods are typically difficult to assess, particularly in the absence of automated procedures for choosing Markov chain parameters. Such software may actually not provide procedures for LRTs of fixed effects.

4.2 Gamma GLMM

We reconsider the previously introduced Gamma GLMM, and some variations of it. When there is no spatial correlation, so that lme4 can now be considered, together with other packages based on some of the methods implemented in spaMM. HGLMMM (Molas and Lesaffre, 2011) was previously considered in this documentation, but has been "removed from the CRAN repository" on 21/12/2013 (it is still available from the "archive"). In the example developed below, it gave exactly the same point estimates and likelihoods as the HLfit fit shown p. 17. By default, the hglm package (Rönnegård et al., 2010) returns estimates similar to those produced by HLfit with option HLmethod="EQL-". The default method in spaMM corresponds to what has been called HL(1,1) in the literature.

[9]The COMPoissonReg package provides an alternative method to fit efficiently this GLM, but which may involve a severe truncation of the infinite sum, and may also have problems in evaluating the likelihood for low ν_{CMP}.

4.2.1 A comparison with Lee et al.'s (2011) estimates

The non-spatial Gamma GLMM fit considered here was considered by Lee et al. (2011), and the following analysis suggests that spaMM (and HGLMMM) are more accurate than the software used in that study (presumably Genstat). The likelihood values they give for this model are slightly higher than the HLfit ones but even higher than those that can be recomputed by HLfit for the estimates reported in the paper, which are given by[10]

```
phiGiven <- with(wafers,
  exp(as.matrix(cbind(1,X3,X3^2)) %*% matrix(c(-2.90,0.10,0.95))))
etaGiven <- with(wafers,
  5.55+0.08*X1-0.21*X2-0.14*X3-0.08*X2^2-0.09*X1*X3-0.09*X2*X3)
wafers <- cbind(wafers,etaGiven=etaGiven)
HLfit(y ~(1|batch)+offset(etaGiven),
      family=Gamma(log),data=wafers,
      REMLformula=y ~X1*X3+X2*X3+I(X2^2)+(1|batch),
      ranFix=list(lambda=exp(-3.67),phi=phiGiven))

## formula: y ~ (1 | batch) + offset(etaGiven)
## Family: Gamma ( link = log )
## No fixed effect
##  ---------- Random effects ----------
## Family: gaussian ( link = identity )
## Fixed lambda value ( batch ):  0.02548
## # of obs: 198; # of groups: batch, 11
##  -------- Residual variance  --------
## phi was fixed.
##  -------- Likelihood values  --------
##                            logLik
## p_v(h) (marginal L): -1157.663
##   p_beta,v(h) (ReL): -1175.251
```

Such discrepancies may be due to small differences in the data, but attempts to explain them led to the following observations (beyond validating the HLfit results). Discrepancies already occur in the fit of a simple Gamma GLM (still with log link), where HLfit computations can easily been checked. In this case, the GLM weights being 1, the exact ML estimates of fixed effects

[10]this is an instructive example of how to constrain HLfit fits. The REMLformula further allows to obtain the restricted likelihood, although no REML estimation is actually performed in this fit.

are independent of the ϕ estimate, and it is easy to check that HLfit gives the same fixed effect estimates as glm does. Given known fixed effect estimates, the exact likelihood and exact restricted likelihoods are known functions of ϕ, and are also easily checked.

Such comparisons also highlight some subtleties with respect to dispersion estimation. glm estimates ϕ as residual deviance/residual degrees of freedom, and this is not the maximum likelihood estimator for two reasons: it uses deviance residuals as an approximation for the likelihood function, and is uses residual degrees of freedom which is a basic form of REML correction (McCullagh and Nelder, 1989, p. 363) hence not ML. What glm does can be described as an REML procedure using extended quasi likelihood (EQL), something that can be reproduced by HLfit by using the arguments HLmethod="RE(1,1,0)" or HLmethod="EQL-", for example. HLmethod="ML" will provide full ML estimates (exact for a GLM). Without such specifications, for all models HLfit performs an REML analysis that does not rest on approximations by deviance residuals (though it is still approximate in other ways).

Lee et al. (2011) also considered a Gamma-inverse Gamma HGLM which is not implemented in all the above R packages. For this model HLfit and GenStat exhibit small discrepancies similar to those discussed above.

4.2.2 A comparison with glmer (and glmmADMB)

Comparisons with glmer were attempted, but it is not clear how to analyse a structured dispersion model (i.e., a model for the variance of the residual error) with glmer. Also it does not perform REML (or something pretending to be REML) for non-Gaussian response data. For comparison, we therefore first perform an ML fit without structured dispersion by HLfit:

```
glmmfit <- HLfit( y ~X1+X2+X1*X3+X2*X3+I(X2^2)+(1|batch),
      family=Gamma(log),HLmethod="ML",data=wafers)
glmmfit

## formula: y ~ X1 + X2 + X1 * X3 + X2 * X3 + I(X2^2) + (1 | batch)
## Estimation of lambda and phi by ML approximation (p_v).
## Estimation of fixed effects by ML approximation (p_v).
## Family: Gamma ( link = log )
##  ------- Fixed effects (beta) -------
##              Estimate Cond. SE t-value
## (Intercept)  5.61229  0.05623  99.802
## X1           0.08815  0.03262   2.702
## X2          -0.21165  0.03262  -6.488
## X3          -0.13903  0.03262  -4.262
## I(X2^2)     -0.10383  0.03264  -3.181
## X1:X3       -0.08992  0.04262  -2.110
## X2:X3       -0.08765  0.04262  -2.056
##  ---------- Random effects ----------
## Family: gaussian ( link = identity )
## Coefficients for [ lambda = var(u) ]:
##  Group        Term Estimate Cond.SE
##  batch (Intercept)   -3.955  0.5159
## Estimate of lambda ( batch ):  0.01916
## # of obs: 198; # of groups: batch, 11
##  -------- Residual variance  --------
## phi formula: "phi" ~ 1
## Coefficients for log[ phi= scale param. ]
##              Estimate Cond. SE
## (Intercept)   -1.833   0.1011
## Estimate of phi:  0.1599  (residual var = phi * mu^2)
##  -------- Likelihood values  --------
##                            logLik
## p_v(h) (marginal L): -1191.025
##   p_beta,v(h) (ReL): -1191.025
```

`glmer` also provides fits of Gamma GLMMs:

```
library(lme4)
glmer(formula= y ~X1+X2+X1*X3+X2*X3+I(X2^2)+(1|batch),
      family=Gamma(log),data=wafers)

## Generalized linear mixed model fit by maximum likelihood (Laplace
##   Approximation) [glmerMod]
##  Family: Gamma  ( log )
## Formula: y ~ X1 + X2 + X1 * X3 + X2 * X3 + I(X2^2) + (1 | batch)
##    Data: wafers
##       AIC       BIC    logLik  deviance  df.resid
##  2395.344  2424.939 -1188.672  2377.344       189
## Random effects:
##  Groups   Name        Std.Dev.
##  batch    (Intercept) 0.1176
##  Residual             0.3919
## Number of obs: 198, groups:  batch, 11
## Fixed Effects:
## (Intercept)           X1           X2           X3      I(X2^2)
##     5.60820      0.08834     -0.21153     -0.14208     -0.10351
##       X1:X3        X2:X3
##    -0.08957     -0.08860
```

Parameter estimates slightly differ but the maximized likelihood is substantially higher, which is intriguing. However, evaluation of the likelihood by numerical integration (which is straightforward given the simple structure of the random effects) shows that `HLfit`'s approximation of the likelihood is more accurate than `glmer`'s one, and that `HLfit` more closely maximize the true likelihood. In particular, numerical integration shows that the log likelihood is -1191.273 for the estimates given by `glmer`, which is distinct from `glmer`'s likelihood value, but very close to the value given by `HLfit` for the parameters estimates obtained with `glmer`:

```
etaLGiven <- with(wafers,5.60820+0.08834*X1-0.21153*X2
      -0.14208*X3-0.10351*X2^2-0.08957*X1*X3-0.08860*X2*X3)
wafers <- cbind(wafers,etaLGiven=etaLGiven)
HLfit( y ~(1|batch)+offset(etaLGiven),
  family=Gamma(log),data=wafers,
  ranFix=list(lambda=0.1176^2,phi=0.3919^2))

## formula: y ~ (1 | batch) + offset(etaLGiven)
## Family: Gamma ( link = log )
## No fixed effect
##  ---------- Random effects ----------
## Family: gaussian ( link = identity )
## Fixed lambda value ( batch ):  0.01383
## # of obs: 198; # of groups: batch, 11
##  -------- Residual variance  --------
## phi was fixed to 0.153586
##  -------- Likelihood values  --------
##                          logLik
## p_v(h) (marginal L): -1191.271
##   p_beta,v(h) (ReL): -1191.271
```

Numerical integration also shows that the likelihood is -1191.02 for parameters estimates given by `HLfit`, which are therefore the better fit. The p_v approximation is here again very close (-1191.025).

The result with glmmADMB, version 0.8.3.3, is unsatisfactory:

```
library(glmmADMB)
glmmadmb( y ~X1+X2+X1*X3+X2*X3+I(X2^2)+(1|batch),
      family="gamma",link="log",data=wafers)

## Warning:  running command 'C:\windows\system32\cmd.exe /c glmmadmb
-maxfn 500 -maxph 5 -noinit -shess' had status 1
## Parameters were estimated, but standard errors were not:  the
most likely problem is that the curvature at MLE was zero or neg-
ative
## Error in glmmadmb(y ~ X1 + X2 + X1 * X3 + X2 * X3 + I(X2^2) +
(1 | batch), :  The function maximizer failed (couldn't find parameter
file) Troubleshooting steps include (1) run with 'save.dir' set and
inspect output files; (2) change run parameters:  see '?admbControl';(3)
re-run with debug=TRUE for more information on failure mode
```

4.2.3 PQL vs. glmmPQL

spaMM's implementation of PQL is Breslow and Clayton's (1993) PQL as also discussed by Lee and Nelder (1996). glmmPQL is described as "equivalent to PQL up to details in the approximations" (Venables and Ripley, 2002). One would have to dig into the glmmPQL code to find the details, but the results indeed apears to differ slightly. E.g., one can compare the two following fits

```
data(wafers)
hfit <- HLfit(y ~X1*X3+X2*X3+I(X2^2)+(1|batch),family=Gamma(log),
      data=wafers,HLmethod="PQL")
if(require(MASS,quietly = TRUE)) {
  gfit <- glmmPQL(y ~X1*X3+X2*X3+I(X2^2),random= ~ 1|batch,family=Gamma(log),
      data=wafers)
}
```

The full output is not shown to save space, but e.g. ϕ estimates are 0.1649 vs 0.1508, and λ estimates are 0.02171 vs 0.01966. glmmPQL does not return likelihood values for comparison with spaMM's ones.

4.3 Negative binomial model

The standard negative binomial model can be fitted using `family=negbin()`:

```
fitme(cases~I(prop.ag/10)+(1|gridcode)
               +offset(log(scotlip$expec)),data=scotlip,
               family=negbin(),HLmethod="ML")

## formula: cases ~ I(prop.ag/10) + (1 | gridcode) + offset(log(scotlip$expec))
## Estimation of lambda and NB_shape by ML approximation (p_v).
## Estimation of fixed effects by ML approximation (p_v).
## Estimation of lambda and NB_shape by 'outer' ML, maximizing p_v.
## Family: Neg.binomial(shape=2.984) ( link = log )
##  ------- Fixed effects (beta) -------
##              Estimate Cond. SE t-value
## (Intercept)    -0.3528   0.1495  -2.359
## I(prop.ag/10)   0.7148   0.1324   5.398
##  ---------- Random effects ----------
## Family: gaussian ( link = identity )
## Outer estimate of lambda ( gridcode ):  1e-06
## # of obs: 56; # of groups: gridcode, 56
##  -------- Likelihood values  --------
##                           logLik
## p_v(h) (marginal L): -171.4703
##   p_beta,v(h) (ReL): -171.4703
```

The maximum likelihood fit of ngative binomial GLMs is identical to the `glm.nb` fit (from the `MASS` package):

```
nbfit <- glm.nb(cases~I(prop.ag/10)+offset(log(scotlip$expec)),
       data=scotlip)
summary(nbfit)

##
## Call:
## glm.nb(formula = cases ~ I(prop.ag/10) + offset(log(scotlip$expec)),
##     data = scotlip, init.theta = 2.984280248, link = log)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.9030  -0.8597  -0.1937   0.5310   1.7205
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept)    -0.3528     0.1495  -2.359   0.0183 *
## I(prop.ag/10)   0.7148     0.1324   5.398 6.75e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for Negative Binomial(2.9843) family taken to be 1)
##
##     Null deviance: 86.474  on 55  degrees of freedom
## Residual deviance: 62.227  on 54  degrees of freedom
## AIC: 348.94
##
## Number of Fisher Scoring iterations: 1
##
##
##               Theta:  2.984
##           Std. Err.:  0.786
##
##  2 x log-likelihood:  -342.941
```

Alternatively, the negative binomial is the Poisson-Gamma model with $v = \ln(u)$, which can therefore be fitted with `family=poisson()` and `rand.family=Gamma(log)`. The shape parameter of `negbin` should then be compared to $1/\lambda$, the reciprocal of the variance of the Gamma random effects. However, this mixed-model representation uses a Laplace approximation for the likelihood, which yields less accurate results for high variance of the random effects.

4.4 Random-slope model

As pointed above, the current implementation of random-slope models in `spaMM` is computationally inefficient. `lmer` can be much faster at fitting them, but will occasionally fail, as for the following example illustrates

```
set.seed(5432); J <- 15; N <- 30
test.df <- data.frame( unit = sort(rep(c(1:N),J)),
                       J = rep(c(1:J),N) , x = rnorm(n = J*N) )
```

```
beta <- 3 + .2*rnorm(N)
test.df$beta <- beta[test.df$unit]
test.df$y <- 1 + test.df$x * test.df$beta + .75*rnorm(n = J*N)
HLfit(y ~ x + (1+x|unit), data = test.df)

## formula: y ~ x + (1 + x | unit)
## REML: Estimation of lambda and phi by REML.
##       Estimation of fixed effects by ML.
## Family: gaussian ( link = identity )
##  ------- Fixed effects (beta) -------
##             Estimate Cond. SE t-value
## (Intercept)    1.026  0.03704   27.71
## x              3.055  0.04059   75.28
##  ---------- Random effects ----------
## Family: gaussian ( link = identity )
## Coefficients for [ lambda = var(u) ]:
##  Group        Term Estimate Cond.SE     Var. Corr.
##   unit (Intercept)  -18.608  521.28 0.004076
##   unit           x   -4.045    0.45  0.01344    -1
## Estimate of lambda ( unit ):  1e-08
## # of obs: 450; # of groups: unit, 60
##  -------- Residual variance  --------
## phi formula: "phi" ~ 1
## Coefficients for log[ phi= residual var ]
##             Estimate Cond. SE
## (Intercept)  -0.6111  0.06756
## Estimate of phi=residual var:  0.5428
##  -------- Likelihood values  --------
##                            logLik
## p_v(h) (marginal L): -506.4373
##   p_beta,v(h) (ReL): -511.1044
```

This result is practically equivalent to the one reported for this example using an old unspecified version of `lmer`.[11] Results with a more recent version of `lmer` may be different. In particular, version 1.1-11 reports a NaN correlation value, after a convergence warning.

[11]http://www.r-bloggers.com/random-regression-coefficients-using-lme4/

Acknowledgements

spaMM development has benefitted from a PEPS grant from CNRS and universities Montpellier 1 & 2, and from collaborations with Jean-Baptiste Ferdy and Alexandre Courtiol.

Bibliography

Breslow, N. E., and Clayton, D. G. 1993. Approximate inference in generalized linear mixed models, *J. Am. Stat. Assoc.* **88**, 9–25.

Clayton, D., and Kaldor, J. 1987. Empirical Bayes estimates of age-standardized relative risks for use in disease mapping, *Biometrics* **43**, 671–681.

Coull, B. A., and Agresti, A. 2000. Random effects modeling of multiple binomial responses using the multivariate binomial logit-normal distribution, *Biometrics* **56**, 73–80.

Lee, W., and Lee, Y. 2012. Modifications of REML algorithm for HGLMs, *Stat. Computing* **22**, 959–966.

Lee, Y., and Nelder, J. A. 1996. Hierarchical generalized linear models, *J. R. Stat. Soc. B* **58**, 619–678.

Lee, Y., and Nelder, J. A. 2001. Hierarchical generalised linear models: A synthesis of generalised linear models, random-effect models and structured dispersions, *Biometrika* **88**, 987–1006.

Lee, Y., Nelder, J. A., and Park, H. 2011. HGLMs for quality improvement, *Applied Stochastic Models in Business and Industry* **27**, 315–328.

Lee, Y., Nelder, J. A., and Pawitan, Y. 2006. "Generalized linear models with random effects: unified analysis via H-likelihood," Chapman & Hall.

McCullagh, P., and Nelder, J. A. 1989. "Generalized linear models," Chapman & Hall, second edn.

Molas, M., and Lesaffre, E. 2010. Hurdle models for multilevel zero-inflated data via h-likelihood, *Stat.Med.* **29**, 3294–3310.

Molas, M., and Lesaffre, E. 2011. Hierarchical generalized linear models: The R package HGLMMM, *J. stat. Software* **39**, 1–20.

Rönnegård, L., Shen, X., and Alam, M. 2010. hglm: A package for fitting hierarchical generalized linear models, *R Journal* **2**, 20–27.

Rousset, F., and Ferdy, J.-B. 2014. Testing environmental and genetic effects in the presence of spatial autocorrelation, *Ecography* **37**, 781–790.

Shmueli, G., Minka, T. P., Kadane, J. B., Borle, S., and Boatwright, P. 2005. A useful distribution for fitting discrete data: revival of the Conway–Maxwell–Poisson distribution, *appl. Stat.* **54**, 127–142.

Venables, W. N., and Ripley, B. D. 2002. "Modern applied statistics with S," Springer-Verlag, New York, fourth edn.

Index

$\mathbf{X}$ (design matrix), 4
$\mathbf{Z}$ (design matrix), 11
$\boldsymbol{\beta}$ (fixed-effect coefficients), 4
$\boldsymbol{\eta}$ (linear predictor), 11
λ (variance of random effects u), 4
$\boldsymbol{\mu}$ (expected response), 11
ν (smoothness of correlation), 5
$p_{\beta,v}$ (restricted likelihood approximation), 8
p_v (likelihood approximation), 8
ϕ (variance of residual error), 4
ρ (scale of correlation), 5
$\mathbf{u}$ (random effect), 11
$\mathbf{v}$ (random effect on linear scale), 11

APHLs, 22

Confidence interval, 6
Correlation model
  arbitrary given, 14
  Conditional autoregressive (CAR), 13
  Matérn, 4

EQL, 28, 30

GLM
  canonical form, 24
glm, 30

HGLM, 15

Joint GLM, 19

logdet, 22

Prediction, 9

Random effects
  Beta, 15
  crossed, 14
  Gamma
    log link, 34
  inverse-Gamma, 18, 30
  random coefficients, 20, 36
Random-slope, *see* Random effects, random coefficients
REML, 8, 30
Response families
  binomial, 13, 15
  COMPoisson, 27
  Gamma, 16
  multinomial, 12
  Negative-binomial, 35
  Poisson, 13