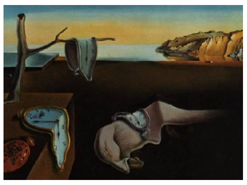

Journal of Statistical Software

November 2021, Volume 100, Issue 10. doi: 10.18637/jss.v100.i10

dalmatian: A Package for Fitting Double Hierarchical Linear Models in R via JAGS and nimble

Simon Bonner
University of Western Ontario

Han-Na Kim
University of Western Ontario

David Westneat
University of Kentucky

Ariane Mutzel
University of Kentucky

Jonathan Wright
NTNU

Matthew Schofield
University of Otago

Abstract

Traditional regression models, including generalized linear mixed models, focus on understanding the deterministic factors that affect the mean of a response variable. Many biological studies seek to understand non-deterministic patterns in the variance or dispersion of a phenotypic or ecological response variable. We describe a new R package, **dalmatian**, that provides methods for fitting double hierarchical generalized linear models incorporating fixed and random predictors of both the mean and variance. Models are fit via Markov chain Monte Carlo sampling implemented in either **JAGS** or **nimble** and the package provides simple functions for monitoring the sampler and summarizing the results. We illustrate these functions through an application to data on food delivery by breeding pied flycatchers (Ficedula hypoleuca). Our intent is that this package makes it easier for practitioners to implement these models without having to learn the intricacies of Markov chain Monte Carlo methods.

Keywords: Bayesian inference, diversity patterns, hierarchical models, generalized linear models, Markov chain Monte Carlo, structured residual variance, variance patterns.

1. Introduction

Linear models and their extensions, including generalized linear models, generalized additive models, and even random or mixed effects models, focus on describing how the mean of a response variable varies either as a function of known explanatory variables or as a result of unexplained variation between observational units. In all of these models, the residuals are assumed to be random noise generated from some distribution whose variance does not depend directly on the explanatory variables or random effects. Simple linear regression, for example, commonly assumes that the residuals are normally distributed with mean zero and constant variance while the variance of the response in a generalized linear model is determined by the mean and, possibly, a constant dispersion parameter. However, there are many cases in which it is as important to understand changes in the variation of a response as it is to understand changes in its mean.

Our own work is motivated by problems of understanding differences in the variability of a biological response that may be associated with ecological or evolutionary effects. Ecological and organismal variation exists in hierarchical structures: population dynamics vary with spatial scale (e.g., Bjørnstad, Ims, and Lambin 1999; Liebhold, Koenig, and Bjørnstad 2004), indices of diversity vary with spatial scale and trophic level (e.g, Willig, Kaufman, and Stevens 2003), and phenotypes vary within , among individuals, and among higher taxonomic units (e.g., Westneat, Wright, and Dingemanse 2015). Partitioning variation according to these hierarchical structures is key to understanding how differences between individuals, species, and ecological communities are produced and maintained. Linear mixed effects models provide an important tool for describing patterns in variance and covariance, but impose the assumption of constant residual variance conditional on the fixed and random effects in the mean. However, this assumption may be violated, and recent work has shown that residual variance may differ as the result of some relatively under-studied ecological or evolutionary processes (Westneat *et al.* 2015) and may also exhibit hierarchical structure itself (Westneat, Schofield, and Wright 2012).

The **dalmatian** package facilitates fitting of double hierarchical generalized linear models (DHGLM) in R (R Core Team 2021) via Markov chain Monte Carlo (MCMC) sampling implemented in **JAGS** (Plummer 2003) or **nimble** (de Valpine, Turek, Paciorek, Anderson-Bergman, Temple Lang, and Bodik 2017; de Valpine, Paciorek, Turek, Michaud, Anderson-Bergman, Obermeyer, Wehrhahn Cortes, Rodríguez, Temple Lang, and Paganin 2020). These models extend traditional generalized linear mixed effects models (GLMM) by allowing the dispersion parameter to depend on both fixed and random effects. The package allows users to construct these models through a simple syntax that describes the structure of the mean and dispersion components. The package also provides functions to generate initial values, assess the MCMC sampler's performance, and summarize the output, both visually and numerically. Our hope is that this makes these complex models more accessible to other researchers who are interested in modeling changes in the variability in ecological problems, or in any other field.

2. Getting the package

Package **dalmatian** (Bonner and Kim 2021) is available from the Comprehensive R Archive Network (CRAN) at `https://CRAN.R-project.org/package=dalmatian`. The **dalmatian** package can be installed in R by running `install.packages("dalmatian")`. At the time of writing the version available is 1.0.0. Updates to the package will be made available on CRAN and are developed on GitHub at `https://github.com/sjbonner/dalmatian`.

3. Model structure

Data are assumed to comprise a sample of n observations with Y_i denoting the response for observation i, $\boldsymbol{x}_{\mu,i}$ the vector of observed covariates associated with the mean, and $\boldsymbol{x}_{\phi,i}$ the vector of observed covariates associated with the dispersion. Note that $\boldsymbol{x}_{\mu,i}$ and $\boldsymbol{x}_{\phi,i}$ may share components. Let $\boldsymbol{Y} = (Y_1, \dots, Y_n)^\top$ denote the combined vector of responses, X_μ the matrix of covariates for the mean with $\boldsymbol{x}_{\mu,i}^\top$ in the i-th row and X_ϕ the matrix of covariates for the dispersion with $\boldsymbol{x}_{\phi,i}^\top$ in the i-th row. Similarly, let Z_μ and Z_ϕ represent the design matrices for the random effects of the mean and dispersion components.

Models fit by **dalmatian** assume the following structure:

1. Conditional on the fixed and random effects, $Y_1, \dots, Y_n$ represent independent draws from some distribution which can be parameterized in terms of the mean, μ_i, and a dispersion parameter, ϕ_i, such that $E(Y_i) = \mu_i$ and $\mathrm{Var}(Y_i)$ is a function of μ_i and ϕ_i. This definition includes distributions in the exponential family, but broadens the class of models to include distributions like the negative binomial and beta-binomial that are commonly applied to over-dispersed counts or success/failure data. Table 1 describes the distributions currently available. We plan to add further distributions as the package is developed.

2. The mean and dispersion for each observation, after suitable transformation, are linear functions of the associated fixed and random effects such that
$$g(\boldsymbol{\mu}) = \boldsymbol{\eta} = X_\mu \boldsymbol{\alpha} + Z_\mu \boldsymbol{\epsilon} \tag{1}$$
and
$$h(\boldsymbol{\phi}) = \boldsymbol{\lambda} = X_\phi \boldsymbol{\psi} + Z_\phi \boldsymbol{\xi}. \tag{2}$$
Here $g(\boldsymbol{\mu}) = (g(\mu_1), \dots, g(\mu_n))^\top$ and $h(\boldsymbol{\phi}) = (h(\phi_1), \dots, h(\phi_n))^\top$ represent the vectors formed by applying the link functions $g(\cdot)$ and $h(\cdot)$ to each element of $\boldsymbol{\mu}$ and $\boldsymbol{\phi}$ respectively, and $\boldsymbol{\eta} = (\eta_1, \dots, \eta_n)^\top$ and $\boldsymbol{\lambda} = (\lambda_1, \dots, \lambda_n)^\top$ denote the linear predictors for the two components.

3. The random effects are assumed to be independent and normally distributed such that
$$\epsilon_j \sim N(0, \tau_\epsilon^2), \quad j = 1, \dots, p$$
and
$$\xi_k \sim N(0, \tau_\xi^2), \quad k = 1, \dots, q$$
where p and q denote the length of $\boldsymbol{\epsilon}$ and $\boldsymbol{\xi}$, respectively. Current versions of the package do not allow for non-normal random effects or correlation between the elements of $\boldsymbol{\epsilon}$ and $\boldsymbol{\xi}$, though we plan to relax this assumption in the future.

4. Sample data

As an example, we consider data from a study of pied flycatchers (Ficedula hypoleuca) collected in the Abergwyngregyn National Nature Reserve, North Wales, United Kingdom, in

	Gaussian	Gamma	Negative binomial	Beta-binomial
Support	$(-\infty, \infty)$	$(0, \infty)$	$0, 1, 2, \ldots$	$0, 1, 2, \ldots, m$
JAGS parameters	μ, τ^2	r, λ	r, p	α, β
Mean	$\mu \in (-\infty, \infty)$	$\frac{r}{\lambda} > 0$	$\frac{r(1-p)}{p} > 0$	$\frac{m\alpha}{(\alpha+\beta)} \in (0, m)$
Variance	$\frac{1}{\tau^2}$	$\frac{r}{\lambda^2}$	$\frac{r(1-p)}{p^2}$	$\frac{m\alpha\beta(\alpha+\beta+m)}{(\alpha+\beta)^2(\alpha+\beta+1)}$
Dispersion parameter	$\frac{1}{\tau^2} > 0$	$\frac{1}{\lambda} > 0$	$\frac{1}{p} > 1$	$\frac{1}{\alpha+\beta+1} \in (0, 1)$

Table 1: Response distributions currently implemented in **dalmatian**. The columns describe the distributions in terms of their support, the parameterization implemented in **JAGS**, the relationship between the **JAGS** parameters and the mean and variance, and the dispersion parameter.

the summers of 1998 and 1999. Westneat, Mutzel, Bonner, and Wright (2017) presented full details of the study. Briefly, the researchers observed pairs of flycatchers nesting in 21 nest boxes in the first year and 16 nest boxes in the second year to study the effects of manipulating brood size on the parents' provisioning behavior. Chicks were moved between the nest boxes 2 or 3 days after hatching to artificially increase or decrease the number of offspring in each brood. Video cameras affixed to the boxes enabled the researchers to monitor the timing of the parents' visits to the nests, types of prey delivered, and begging behavior of chicks, and electronic balances fitted to each box monitored the amount of food left by the parents on these visits (the load), rounded to the nearest tenth of a gram. Each box was monitored for 6 bouts of approximately 1.5 hours (dependent on the camera's battery) over one 24 hour period 7–12 days after hatching. All data from 7 boxes observed in 1998 and 3 in 1999 were excluded because the nests were predated or only a single parent attended the chicks. Individual visits were excluded if the parent removed a fecal sac from the nest, since the load could not be determined. The final data contained 4693 records from 52 parents of 26 broods. These data can be accessed with `data("pied_flycatchers_1", package = "dalmatian")` once the package is loaded in R.

The initial model we fitted to these data considered the log of the load on each visit as the response variable. In many cases load was recorded as zero even though the parent was carrying food indicating that the load was below the tolerance of the scale. For these records, we set the response equal to $\log(0.01) \approx -4.6$. We assumed that residuals were normally distributed and considered the mean as a function of the log of the time between visits (the inter-visit interval or IVI) and allowed both the mean and dispersion (the variance) to depend on the treatment, i.e., increased or decreased brood size, and the parent's sex. We also included random effects in both components of the model to allow for variation among individuals and to account for dependence between the repeated observations from each individual. Mathematically, our model implies that the k-th observation of log load from the j-th parent in box i, denoted y_{ijk}, has mean

$$\mu_{ijk} = \alpha_0 + \alpha_1 \log(\text{IVI})_{ijk} + \alpha_2 \text{Brood}_i + \alpha_3 \text{Sex}_{ij} + \epsilon_{ij} \tag{3}$$

and variance σ^2_{ij} such that

$$\log(\sigma^2_{ij}) = \psi_0 + \psi_1 \text{Brood}_i + \psi_2 \text{Sex}_{ij} + \xi_{ij} \tag{4}$$

where $\text{Brood}_i = 1$ if the brood size in box i was increased, and 0 otherwise, and $\text{Sex}_{ij} = 1$ if the parent is male, and 0 otherwise. We let $\tau^2_\epsilon = \text{Var}(\epsilon_{ij})$ and $\tau^2_\xi = \text{Var}(\xi_{ij})$ denote the random effects variances.

The following code loads the library, makes the data available, and computes the appropriate response variable. We also set the random seed in order that our results can be reproduced exactly (which would not be necessary in new analyses):

```
R> library("dalmatian")
R> set.seed(27182)
R> data("pied_flycatchers_1", package = "dalmatian")
R> pfdata$logLoad <- log(pmax(pfdata$load, 0.01))
```

5. Model fitting

5.1. Model definition

The structure of the model is defined through two named lists corresponding to the mean and dispersion components of the model, which are passed as arguments to the package's main function, `dalmatian()`. Each of these lists may contain three elements named `fixed`, `random`, and `link`. Both `fixed` and `random` are themselves lists containing the following elements:

- `name`: The base name for the corresponding coefficients other function that can appear on the left side of an expression in **JAGS**/**nimble**.
- `formula`: A one-side formula defining the associated linear predictor.
- `priors`: A list defining the prior distributions for the associated parameters.

The `link` element must be a text string specifying the link function. Choices include `"log"`, `"logit"`, `"cloglog"`, and `"probit"`. If this element is not defined then the identity link will be used. Each element of the list of prior distributions must be a vector that specifies the class of distribution, by name of the corresponding **JAGS**/**nimble** function (e.g., `dnorm` for the normal distribution and `dt` for the *t* distribution), and its parameters according to the default BUGS parameterization as described in the manuals for both **JAGS** (Plummer 2017, Section 9.2) and **nimble** (de Valpine *et al.* 2020, Section 5.2.4). The list of prior distributions may either have the same length as the number of effects or length one, in which case the same prior is assigned to all parameters in that component of the model. For example, if a component of the model contained an intercept and a slope then `priors = list(c("dnorm", 0, 0.001))` would specify that both parameters are assigned independent normal priors with mean 0 and precision 0.001 (variance 1000). Alternatively, `priors = list(c("dnorm", 0, 0.001), c("dexp", 0, 0.001))` would specify that the intercept is assigned a normal prior while the slope is assigned a double exponential prior with mean, both with mean 0 and precision 0.001.

Prior distributions for each parameter are assumed to be independent and any univariate distribution supported by the chosen MCMC engine may be assigned to any parameter. It is generally up to the user to select appropriate distributions. However, **dalmatian** will, by default, constrain the values generated by the MCMC sampler such that the mean and dispersion remain within the allowable ranges for the selected model (e.g., $\phi_i > 0$ for the Gaussian model or $\mu_i > 0$ and ϕ_0 for the gamma model). If the list of prior distributions is omitted then default prior distributions are assigned. These are normal with mean 0 and variance 1000 for fixed effects and half-t with 3 degrees of freedom and scale parameter 5 for the standard deviations of any random effects.

The mean component of the pied flycatcher load model in Equation 3 is defined as:

```
R> mymean <- list(
+    fixed = list(name = "alpha", formula = ~ log(IVI) + broodsize + sex),
+    random = list(name = "epsilon", formula = ~ -1 + indidx)
+  )
```

The `fixed` element indicates that the linear predictor contains three known predictors and the associated parameters will be called `alpha.log(IVI)`, `alpha.broodsize`, and `alpha.sex`. These parameters will be assigned the default prior. The `random` element indicates that the random effects are defined according to the variable `indix`, a factor with distinct levels for each of the 60 individuals observed during the experiment. The vector of standard deviations for the random effects will be named `sd.epsilon` and will have length equal to the number of effects in the formula. The `-1` in the formula of the random effects is required to remove the intercept from the design matrix and maintain the mean of zero. The identity link function will be used since `link` is not specified.

The dispersion component in Equation 4 is defined similarly:

```
R> mydisp <- list(
+    fixed = list(name = "psi", formula = ~ broodsize + sex),
+    random = list(name = "xi", formula = ~ -1 + indidx),
+    link = "log"
+  )
```

The only differences are that we add the `link` element to specify that the dispersion be modeled on the log scale, remove `log(IVI)` from the formula for the fixed effects, and change the base name of the parameters. Once again, we assign the default priors.

5.2. MCMC arguments

The choice of MCMC sampling package is controlled by the `engine` argument of the function `dalmatian()` (either `engine = "JAGS"` or `engine = "nimble"`). Further arguments also need to be supplied to control the MCMC sampler. These arguments are defined in two named lists, `jags.model.args` and `coda.samples.args`, which take their names from the functions `jags.model()` and `coda.samples()` for creating and sampling from model objects in the **rjags** package (Plummer 2021). The only required element of `jags.model.args` is `file` that names the file to which the model code will be written. Further, optional, arguments include `inits`, `n.adapt`, and `quiet`, as described in the help for `jags.model`. If initial values are provided then **dalmatian** will take the number of chains, `n.chains`, from the length of `inits`. Otherwise, three chains will be run in parallel and the initial values will be generated as described in Section 5.3. If `n.adapt` is not set then the chains will be adapted over 1000 iterations (the default given by **rjags**). The second list must include the element `n.init` specifying the number of iterations to run the sampler after the adaption phase. Further arguments may include `thin` and `na.rm` as defined in the help file for `coda.samples()`.

The following code will sample from the posterior distribution of the model of the pied flycatcher data by running three parallel chains, since initial values are not specified, with 1000 iterations in the adaption phase and 2000 iterations in the sampling phase. The model file will be named `pied_flycatchers_1.jags`:

```
R> jm.args <- list(file = "pied_flycatchers_1.jags", n.adapt = 1000)
R> cs.args <- list(n.iter = 2000)
```

5.3. Initial values

Our package also handles automatic generation of diffuse initial values to start parallel chains for assessing convergence and mixing. If initial values are not specified then **dalmatian** generates three sets of initial values by fitting models including only the fixed effects (X_μ and X_ϕ), only the random effects (Z_μ and Z_ϕ), and both the fixed and random effects. These models are fit via the package **dglm** for fitting double generalized linear models (DGLMs, Dunn and Smyth 2020) with the random effects treated as fixed effects in order to speed the computations. Initial variances for the random effects are then set equal to the empirical variance of the random effects. For example, the second set of initial values for the pied flycatcher model are generated by fitting the DGLM with:

$$\mu_{ijk} = \alpha_0 + \epsilon_{ij} \quad \text{and} \quad \log(\sigma^2_{ij}) = \psi_0 + \xi_{ij}$$

treating ϵ_{ij} and ξ_{ij} as fixed, not random, effects. Initial values of the random effects variances are then computed as $\text{Var}(\epsilon_{ij}) = \sum_{i,j} \hat{\epsilon}^2_{ij}$ and $\text{Var}(\xi_{ij}) = \sum_{i,j} \hat{\xi}^2_{ij}$ (noting that $\sum_{i,j} \hat{\epsilon}_{ij} = 0$ and $\sum_{i,j} \hat{\xi}_{ij} = 0$). In some cases the mixed effects model will be over-parameterized when the random effects are treated as fixed effects and some values of ξ_{ij} and ϵ_{ij} will not be estimated by **dglm**. When this occurs we compute the variances based on the estimated values and then fill in the missing values with random draws from a mean zero normal distribution. For the first set of initial values we set $\text{Var}(\epsilon_{ij}) = \text{Var}(\xi_{ij}) = 0.001$, since these values must be positive, and allow the sampler to generate the initial values of ϵ_{ij} and ξ_{ij}. Our package also allows the user to specify initial values directly, which is often most useful if initial values can be pulled from a previous run of a similar model as illustrated in Section 7.1.

The initial states of the random number generators for each chain can also be passed to **JAGS** through the list of initial values in the `jags.model.args` list (see Plummer 2017, Section 3.3.2, for details). This is not usually required, but we set the specify the random number generators and their seeds for each model in order that the results shown below can be reproduced:

```
R> jm.args$inits <- list(
+    list(".RNG.name" = "base::Wichmann-Hill", ".RNG.seed" = 13),
+    list(".RNG.name" = "base::Marsaglia-Multicarry", ".RNG.seed" = 1597),
+    list(".RNG.name" = "base::Super-Duper", ".RNG.seed" = 196418))
```

Note that the output in this paper was generated with R 4.1.1, g++ compiler provided by the Gnu Compiler Collection (GCC) version 9.3.0-17, and **JAGS** version 4.3.0 running on Ubuntu Linux 20.04. Exact results may depend on the operating system, version of R, compiler, and compiler version. However, the results in other setups will be very similar and lead to the same conclusions qualitatively.

5.4. Running the sampler

Having defined the model structure and arguments controlling the MCMC sampler, the model can now be fit via the function `dalmatian()`. Required arguments are:

- `df`: The data frame containing the response and predictor values.
- `mean.model`: The list defining the structure of the mean.
- `dispersion.model`: The list defining the structure of the dispersion.
- `jags.model.args`: The list of arguments passed to `jags.model()`.
- `coda.samples.args`: The list of arguments passed to `coda.samples()`.

The user must also specify either the name of the response variable, `response`, or both `lower` and `upper` if the response is censored (see Section 7.1). Further arguments to this function allow the fitting to be fine-tuned and are described in the documentation.

The following command runs the MCMC sampler in **JAGS** for the previously defined model of the pied flycatcher data. As described in Section 5.2, the sampler could be run with **nimble** by adding the argument `engine = "nimble"`.

```
R> pfresults <- dalmatian(df = pfdata, mean.model = mymean,
+    dispersion.model = mydisp, jags.model.args = jm.args,
+    coda.samples.args = cs.args, response = "logLoad")
```

Sampling for this model in **JAGS** took approximately 50 minutes on a machine with an Intel i7-6700K 4.00 GhZ CPU and 32 GB of RAM running Ubuntu Linux 18.04 LTS. The output of `dalmatian()` is an object of class '`dalmatian`' which contains the core input arguments and the sampled values in an `mcmc.list` object called `coda` which can be manipulated with the functions from the **coda** package (Plummer, Best, Cowles, and Vines 2006), see the associated help file for further details.

6. Post-processing

Our package provides several functions to process the MCMC output either to assess convergence and mixing of the sampling algorithm or to summarize the posterior distribution. Most are wrappers for corresponding functions from the **coda** and **ggmcmc** packages (see Plummer *et al.* 2006; i Marín 2016, respectively). All of these functions are constructed as S3 generic functions with specific methods for objects of class '`dalmatian`'.

6.1. Convergence diagnostics

The function `traceplots()` creates trace plots for visually assessing the convergence and mixing of the Markov chains constructed by the sampler. Specifically, the function constructs trace plots for the fixed effects and the standard deviations of the random effects for both the mean and dispersion components, as applicable. If the argument `show = TRUE` hen the plots will be displayed immediately in the open graphics device. Otherwise the saved trace plots for the four components can be accessed individually as follows:

```
R> pftraceplots <- traceplots(pfresults, show = FALSE)
R> pftraceplots$meanFixed
R> pftraceplots$dispersionFixed
R> pftraceplots$meanRandom
R> pftraceplots$dipsersionRandom
```

Trace plots for the parameters of the pied flycatcher example are displayed in Figure 1, Figure 2, and Figure 3 in Appendix A.1. The traces suggest that the chains have converged well, but might be mixing slowly for some parameters indicating that the sampler should be run for longer to compute accurate summaries of the posterior distribution.

Numerical convergence diagnostics including the Brooks-Gelman-Rubin potential scale reduction factor (Gelman and Rubin 1992; Brooks and Gelman 1998), Raftery and Lewis run length diagnostic (Raftery and Lewis 1995), and effective sample size are computed by the function `convergence()`. Again, the individual elements may be saved to be accessed at a later time:

```
R> pfconvergence <- convergence(pfresults)
R> pfconvergence$gelman
R> pfconvergence$raftery
R> round(pfconvergence$effectiveSize)
```

Convergence diagnostics for the pied flycatcher example, provided in Appendix A.2, confirm our suspicions from examining the trace plots. The Brooks-Gelman-Rubin diagnostics indicate that the chains have converged. However, the Raftery and Lewis diagnostics indicate that many more iterations would be required to estimate the 2.5-th percentile of all fixed effects 0.005 with probability 0.95. The estimated effective sample size for these parameters is also below 300; too small to provide precise estimates of the 95% credible intervals.

6.2. Posterior summaries

Functions are also provided to construct numerical and visual summaries of the posterior distribution. Numerical summaries including the mean, standard deviation, and limits of highest posterior density credible intervals are computed via the function `summary()`:

```
R> summary(pfresults)
```

Caterpillar plots which provide a graphical representation of the same information are constructed with `caterpillar()`. As with the trace plots, caterpillar plots for all parameters can be displayed immediately or saved and accessed later as follows:

```
R> pfcaterpillar <- caterpillar(pfresults, show = FALSE)
R> pfcaterpillar$meanFixed
R> pfcaterpillar$dispersionFixed
R> pfcaterpillar$meanRandom
R> pfcaterpillar$dispersionRandom
```

Caterpillar plots of the fixed effects for the analysis of the pied flycatchers shown in Appendix A.4 indicate that there are significant effects of the brood size manipulation, sex, and the log of the IVI on the mean of the log load. There is also some evidence of an effect of sex, but not the brood size manipulation, on the variance. We do not discuss these results further and instead point the reader to the complete analysis in Westneat *et al.* (2017).

6.3. Fitted values

The function `predict()` computes either the posterior mean or mode of the mean and dispersion components for specified values of the covariates. Primary arguments include:

- `object`: An object of class '`dalmatian`' containing previous model output.
- `newdata`: An optional data frame specifying the combinations of covariates at which to compute the fitted values.

If `newdata` is not provided then fitted values are computed for each entry in the original data. Names and formats of variables in `newdata` should match the original data, and any factor variables must have the same number of levels with the same names. Population level fitted values can be obtained by setting the values of variables defining the random effects to `NA`. Further arguments include:

- `method`: Either `mean` (default) or `mode`.
- `se`: A boolean flag indicating whether posterior standard deviations should be computed (defaults to `TRUE`).
- `ci`: A boolean flag indicating whether credible intervals should be computed (defaults to `TRUE`).
- `type`: Identifies whether prediction are on the scale of the linear predictor (the default) or the response.
- `level`: A vector of levels for the credible intervals (defaults to `0.95`).

For example, the following code would compute the fitted value of the linear predictor of the mean and variance of the log load, with 95% credible intervals, for the individual represented in the first line of data, the female at the first box, for values of IVI ranging between 1 and 100:

```
R> newdata <- cbind(pfdata[1, c("broodsize", "sex", "indidx")],
+    data.frame(IVI = 1:100))
R> predict(object = pfresults, newdata = newdata)
```

Alternatively, population level fitted values for the mean and variance for all combinations of sex, brood size manipulation, and values IVI ranging from 1 to 100 can be computed using package **tidyr** (Wickham 2014; Wickham, François, Henry, and Müller 2021) as:

```
R> library("tidyr")
R> newdata1 <- crossing(broodsize = factor(c("Decreased", "Increased")),
+    IVI = 1:100, sex = factor(c("Female", "Male")))
R> newdata1$indidx <- factor(NA, levels(pfdata$indidx))
R> fitted1 <- predict(object = pfresults, newdata = newdata1,
+    type = "response")
```

Plots (created with package **ggplot2**, Wickham 2016) comparing the fitted values of fitted population median load (back-transformed from the fitted mean on the log scale) as a function of IVI for males and females with reduced and increased brood sizes are provided in Appendix A.5. As expected from the model output, the plots indicate that load is higher, on average, for females than for males and for parents in boxes with increased rather than decreased brood sizes.

6.4. Print and plot

The package also provides `print()` and `plot()` methods to provide easy access to several of the above functions and summarize `dalmatian()` objects. Calling the `print()` method displays basic information about the structure of the fitted model, and calls the previously described functions that compute numerical summaries and convergence diagnostics. The `plot()` method calls the previous functions that create trace plots and caterpillar plots.

7. Further features

7.1. Rounding

One feature of **dalmatian** is that it provides the ability to model censored response variables. Loads returned by the parents in the pied flycatcher study were measured on analog scales visible in the video recordings and were rounded to the nearest 0.10 g. For a small number of observations, less than 4%, the researchers attempted to round the load to the nearest 0.05 g. To account for this, we treat load as a censored response variable known only to lie in the interval $(y_{ijk} - 0.049, y_{ijk} + 0.05)$ if the observed load is positive, $y_{ijk} > 0$, and in $(0.001, 0.05)$ if $y_{ijk} = 0$.

To implement this in **dalmatian** we create two new variables bounding the true response variable, which is the logarithm of the load:

```
R> pfdata$lowerLoad <- log(pmax(0.001, pfdata$load - 0.049))
R> pfdata$upperLoad <- log(pfdata$load + 0.05)
```

We then specify that `rounding = TRUE` in the call to `dalmatian()` and supply the new variables as the bounds on the response via the arguments `lower` and `upper`. Here we also illustrate how initial values may be supplied by extracting initial values from previous results.

Initial values for the true load are randomly drawn from a uniform distribution between the lower and upper bounds. Note that initial values are specified in the list passed to the `jags.model.args` argument of `dalmatian()`. Any uninitialized variables are initialized by the sampler. The following code extracts initial values from the final iteration of the previous model and then fits the new model, saving the **JAGS** code to `pied_flycatchers_2.jags`. As before, we set specify the initial states of the random number generators in order that the results can be reproduced:

```
R> jm.args2 <- list(file = "pied_flycatchers_2.jags", n.adapt = 1000,
+    inits = lapply(1:3, function(i) {
+      last <- pfresults$coda[[i]][2000, ]
+      setJAGSInits(mymean, mydisp,
+        fixed.mean = last[1:4], random.mean = last[5:56],
+        fixed.dispersion  = last[57:59],
+        sd.mean = last[60], sd.variance = last[61],
+        random.dispersion = last[62:113],
+        y = runif(nrow(pfdata), pfdata$lowerLoad, pfdata$upperLoad))
+    })
+  )
R > jm.args2$inits[[1]] <- c(jm.args2$inits[[1]],
+    list(".RNG.name" = "base::Wichmann-Hill", ".RNG.seed" = 21))
R> jm.args2$inits[[2]] <- c(jm.args2$inits[[2]],
+    list(".RNG.name" = "base::Marsaglia-Multicarry", ".RNG.seed" = 2584))
R> jm.args2$inits[[3]] <- c(jm.args2$inits[[3]],
+    list(".RNG.name" = "base::Super-Duper", ".RNG.seed" = 317811))
R> pfresults2 <- dalmatian(df = pfdata,
+    mean.model = mymean, dispersion.model = mydisp,
+    jags.model.args = jm.args2, coda.samples.args = cs.args,
+    rounding = TRUE, lower = "lowerLoad", upper = "upperLoad")
```

7.2. Weights

Another feature of **dalmatian** is its ability to handle weighted regression in the context of double linear models. This would arise, for example, if the recorded value Y_i is the average of n_i independent and identically distributed observations each with mean μ_i and variance σ_i^2 as defined in Equation 1 and Equation 2. In this case, $\mathrm{Var}(Y_i) = \sigma_i^2/n_i$ and the model will not provide the correct estimates if this is not accounted for. This can be handled by adding a `weight` argument to the variance structure naming the variable in the data which records the weights.

As an example, suppose that we had only a data set called `pfdataByDay` recording the total number of visits to the nest per day for each bird (`nvisit`), the average of the load returned on these visits `avgLoad`, and the average inter visit interval (`avgIVI`). It is not practical to include random effects in this model because almost no birds were observed for more than one day. So, for illustration, we fit a model to the summarized including only the fixed effects on the mean and dispersion. Here we define the new model including the number of visits as the weights in the dispersion component:

```
R> mymean <- list(fixed = list(name = "alpha",
+      formula = ~ log(avgIVI) + broodsize + sex,
+      priors = list(c("dnorm", 0, 0.001))))
R> mydisp <- list(fixed = list(name = "psi", formula = ~ broodsize + sex),
+    link = "log", weights = "nvisit")
```

The model with the new response variable can then be run with the following code:

```
R> pfdataByDay$logAvgLoad <- log(pmax(pfdataByDay$avgLoad, 0.001))
R> jm.args3 <- list(file = "pied_flycatchers_3.jags", n.adapt = 1000)
R> jm.args$inits = list(
+    list(".RNG.name" = "base::Wichmann-Hill", ".RNG.seed" = 34),
+    list(".RNG.name" = "base::Marsaglia-Multicarry", ".RNG.seed" = 4181),
+    list(".RNG.name" = "base::Super-Duper", ".RNG.seed" = 514229))
R> pfresults3 <- dalmatian(df = pfdataByDay,
+    mean.model = mymean, dispersion.model = mydisp,
+    jags.model.args = jm.args3, coda.samples.args = cs.args,
+    response = "logAvgLoad")
```

Numerical summaries are presented in Appendix C.

8. Comparison with existing packages

Several existing packages for R overlap with **dalmatian**. In the Bayesian framework, generalized linear mixed effects models (GLMM) that incorporate random effects on the mean alone may be fit via the packages **MCMCglmm** which implements the MCMC sampling algorithms with custom code and also allows for multivariate responses (Hadfield 2010), **glmmBUGS** (Brown and Zhou 2010) which acts as an interface to **JAGS**, **OpenBUGS**, or **WinBUGS** in much the same way as **dalmatian**, **brms** which acts as a wrapper for the recently developed Stan software implementing Hamiltonian Monte Carlo (Bürkner 2017), and **blme** which performs inference based on the posterior mode and normal approximation to avoid MCMC sampling (Chung, Rabe-Hesketh, Dorie, Gelman, and Liu 2013). The **bamlss** package (Umlauf, Klein, Simon, and Zeileis 2021) extends this framework further by fitting Bayesian generalized additive models that may capture variation in any parameter for a broad range of response distributions. The linear predictors may be very complex and can include fixed effects, random effects, or smooth terms (in one or more dimensions) defined through basis functions expansions.

In the classical, likelihood based framework, GLMM can be fit with the `glmer()` function from **lme4** which applies quadrature to integrate over the random effects and compute the value of the likelihood (Bates, Mächler, Bolker, and Walker 2015), the **glmm** package which approximates the likelihood by importance sampling of the random effects (Knudson 2020), **glmmsr** which provides sequential reduction approximation in addition to both quadrature (through **lme4**) and importance sampling (Ogden 2019), and **glmmTMB** which acts as a wrapper for the **TMB** (template model builder) package and integrates over the random effects via Laplace approximation (Kristensen, Nielsen, Berg, Skaug, and Bell 2016; Brooks *et al.* 2017). The **dglm** package fits double generalized linear models in which both the mean and variance may be functions of fixed covariates (Dunn and Smyth 2020) and the **glmmLasso** fits GLMM with variable selection performed automatically via the LASSO (L1-penalization) (Groll 2017).

The package that provides most similar functionality to **dalmatian** is **dhglm** (Lee and Noh 2018). Similar to our package, the main function, `dhglmfit()` accepts descriptions of the mean and dispersion models including both fixed and random effects as input. The package does not allow for censored observations, but random effects that may follow the normal, gamma, inverse-gamma, or beta distributions. The main difference, however, is that estimation is conducted in a completely different paradigm based on the h-likelihood – the joint density of the observed data and random effects (see Chapter 4 of Lee, Nelder, and Pawitan 2017, for details). This would be called a complete data likelihood in the Bayesian context. In fact, **dalmatian** makes use of this likelihood to remove the random effects through Monte Carlo integration. In the framework of the h-likelihood, the random effects are estimated by maximizing the h-likelihood and estimates of fixed effects are obtained from the marginal likelihood. Laplace approximations are applied to simplify computation of the marginal likelihood when it cannot be computed analytically, and inference about the dispersion is obtained from the REML likelihood. Complete details of the h-likelihood approach are given in (Lee *et al.* 2017). Jin and Lee (2020) provides a shorter review and also introduces a new, online software package for fitting DHGLM called **Albatross Analytics** (`http://cheoling.snu.ac.kr:3838/DHGLM/`). A Some authors have argued against the use of the h-likelihood (e.g., Meng 2009), but these arguments seem to have been based on the misconception that estimates are computed by direct maximization of extended likelihood, a technique long known to be problematic (Little and Rubin 1983). Our experience with a limited simulation study is that the methods appear to perform well from a frequentist perspective.

9. Conclusion

The package we have presented provides a simple interface that will allow researchers in a broad range of areas to harness the power of DHGLM via MCMC sampling without having to implement the necessary methods. The package relies on general software for MCMC sampling, currently **JAGS** and **nimble**, and we plan to extend the package to interface with Stan as well. This has several advantages including that these packages have a long history and are now very stable, it is simple to implement new models without needing to write custom samplers, and users familiar with these packages may easily edit the code produced by **dalmatian** themselves to make changes to the models. The disadvantage is that MCMC sampling is often slow, particularly for models with many unobserved nodes as when the response is rounded, and generic samplers are usually slower than custom implementations.

As we continue to develop **dalmatian**, we plan to work with the developers of these packages to improve the computational efficiency. We also plan to implement more features, including allowing for non-normally distributed random effects and correlation of the random effects within and between the linear predictors of the mean and variance. We encourage users to contact us with questions and suggestions by e-mail or through the code repository (`https://github.com/sjbonner/dalmatian`), and hope that users will contribute new examples that will help us to illustrate and extend **dalmatian**'s functionality.

Acknowledgments

DFW, AM, and SB were supported by the US National Science Foundation grant IOS1257718. SB was further supported by NSERC Grant 493024-2016, and JW was partly supported by the Research Council of Norway through its Centers of Excellence funding scheme, project number 223257. We would like to thank Duncan Brown and (the then) CCW for access to the pied flycatcher field site at Abergwyngregyn NNR, as well as Gabrielle Archard, Christian Both, Matt Davey, Kim Denny, Ioan Fazey, Camilla Hinde, Adam Morrey, Roberta Spears, Jane Stott, Richard Yarnell and Yoram and Shlomith Yom-Tov for assistance in the field.

References

Bates D, Mächler M, Bolker B, Walker S (2015). "Fitting Linear Mixed-Effects Models Using **lme4**." *Journal of Statistical Software*, **67**(1), 1–48. doi:10.18637/jss.v067.i01.

Bjørnstad ON, Ims RA, Lambin X (1999). "Spatial Population Dynamics: Analyzing Patterns and Processes of Population Synchrony." *Trends in Ecology & Evolution*, **14**(11), 427–432. doi:10.1016/s0169-5347(99)01677-8.

Bonner S, Kim H (2021). ***dalmatian**: Automating the Fitting of Double Linear Mixed Models in **JAGS** and **nimble***. R package version 1.0.0, URL https://CRAN.R-project.org/package=dalmatian.

Brooks ME, Kristensen K, Van Benthem KJ, Magnusson A, Berg CW, Nielsen A, Skaug HJ, Mächler M, Bolker B (2017). "**glmmTMB** Balances Speed and Flexibility among Packages for Zero-inflated Generalized Linear Mixed Modeling." *The R Journal*, **9**(2), 378–400. doi:10.32614/RJ-2017-066.

Brooks SP, Gelman A (1998). "General Methods for Monitoring Convergence of Iterative Simulations." *Journal of Computational and Graphical Statistics*, **7**, 435–455. doi:10.1080/10618600.1998.10474787.

Brown P, Zhou L (2010). "MCMC for Generalized Linear Mixed Models with **glmmBUGS**." *The R Journal*, **2**(1), 13–17. doi:10.32614/rj-2010-003.

Bürkner PC (2017). "**brms**: An R Package for Bayesian Multilevel Models Using Stan." *Journal of Statistical Software*, **80**(1), 1–28. doi:10.18637/jss.v080.i01.

Chung Y, Rabe-Hesketh S, Dorie V, Gelman A, Liu J (2013). "A Nondegenerate Penalized Likelihood Estimator for Variance Parameters in Multilevel Models." *Psychometrika*, **78**(4), 685–709. doi:10.1007/s11336-013-9328-2.

de Valpine P, Paciorek C, Turek D, Michaud N, Anderson-Bergman C, Obermeyer F, Wehrhahn Cortes C, Rodríguez A, Temple Lang D, Paganin S (2020). "**nimble** User Manual." doi:10.5281/zenodo.1211190. R package manual version 0.9.1.

de Valpine P, Turek D, Paciorek C, Anderson-Bergman C, Temple Lang D, Bodik R (2017). "Programming with Models: Writing Statistical Algorithms for General Model Structures with **nimble**." *Journal of Computational and Graphical Statistics*, **26**, 403–413. doi:10.1080/10618600.2016.1172487.

Dunn PK, Smyth GK (2020). ***dglm**: Double Generalized Linear Models.* R package version 1.8.4, URL https://CRAN.R-project.org/package=dglm.

Gelman A, Rubin DB (1992). "Inference From Iterative Simulation Using Multiple Sequences." *Statistical Science*, **7**, 457–511. doi:10.1214/ss/1177011136.

Groll A (2017). ***glmmLasso**: Variable Selection for Generalized Linear Mixed Models by L1-Penalized Estimation.* R package version 1.5.1, URL https://CRAN.R-project.org/package=glmmLasso.

Hadfield JD (2010). "MCMC Methods for Multi-Response Generalized Linear Mixed Models: The **MCMCglmm** R Package." *Journal of Statistical Software*, **33**(2), 1–22. doi:10.18637/jss.v033.i02.

i Marín XF (2016). "**ggmcmc**: Analysis of MCMC Samples and Bayesian Inference." *Journal of Statistical Software*, **70**(9), 1–20. doi:10.18637/jss.v070.i09.

Jin S, Lee Y (2020). "A Review of H-Likelihood and Hierarchical Generalized Linear Model." *WIREs Computational Statistics*, p. e1527. doi:10.1002/wics.1527.

Knudson C (2020). ***glmm**: Generalized Linear Mixed Models via Monte Carlo Likelihood Approximation.* R package version 1.4.2, URL https://CRAN.R-project.org/package=glmm.

Kristensen K, Nielsen A, Berg CW, Skaug H, Bell BM (2016). "**TMB**: Automatic Differentiation and Laplace Approximation." *Journal of Statistical Software*, **70**(5), 1–21. doi:10.18637/jss.v070.i05.

Lee Y, Nelder JA, Pawitan Y (2017). *Generalized Linear Models with Random Effects: Unified Analysis via H-Likelihood.* 2nd edition. Chapman & Hall/CRC, New York.

Lee Y, Noh M (2018). ***dhglm**: Double Hierarchical Generalized Linear Models.* R package version 2.0, URL https://CRAN.R-project.org/package=dhglm.

Liebhold A, Koenig WD, Bjørnstad ON (2004). "Spatial Synchrony in Population Dynamics." *Annual Review of Ecology and Evolution*, **35**, 467–490. doi:10.1146/annurev.ecolsys.34.011802.132516.

Little RJA, Rubin DB (1983). "On Jointly Estimating Parameters and Missing Data by Maximizing the Complete Data Likelihood." *The American Statistician*, **37**, 218–220. doi:10.1080/00031305.1983.10483106.

Meng XL (2009). "Decoding the H-Likelihood." *Statistical Science*, **24**(3), 280–293. ISSN 08834237. doi:10.1214/09-sts277c.

Ogden H (2019). ***glmmsr**: Fit a Generalized Linear Mixed Model.* R package version 0.2.3, URL https://CRAN.R-project.org/package=glmmsr.

Plummer M (2003). "**JAGS**: A Program for Analysis of Bayesian Graphical Models Using Gibbs Sampling." In K Hornik, F Leisch, A Zeileis (eds.), *Proceedings of the 3rd International Workshop on Distributed Statistical Computing (DSC 2003)*. Technische Universität Wien, Vienna, Austria. URL https://www.R-project.org/conferences/DSC-2003/Proceedings/Plummer.pdf.

Plummer M (2017). ***JAGS** Version 4.3.0 User Manual.* URL `https://sourceforge.net/projects/mcmc-jags/files/Manuals/4.x/jags_user_manual.pdf`.

Plummer M (2021). ***rjags**: Bayesian Graphical Models Using MCMC.* R package version 4-12, URL `https://CRAN.R-project.org/package=rjags`.

Plummer M, Best N, Cowles K, Vines K (2006). "**coda**: Convergence Diagnosis and Output Analysis for MCMC." *R News*, **6**(1), 7–11. URL `https://CRAN.R-project.org/doc/Rnews/`.

Raftery AE, Lewis SM (1995). "The Number of Iterations, Convergence Diagnostics and Generic Metropolis Algorithms." In WR Gilks, DJ Spiegelhalter, S Richardson (eds.), *Markov Chain Monte Carlo in Practice*. Champan and Hall, London.

R Core Team (2021). *R: A Language and Environment For Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria. URL `https://www.R-project.org/`.

Umlauf N, Klein N, Simon T, Zeileis A (2021). "**bamlss**: A Lego Toolbox for Flexible Bayesian Regression (and Beyond)." *Journal of Statistical Software*, **100**(4), 1–53. `doi:10.18637/jss.v100.i04`.

Westneat DF, Mutzel A, Bonner S, Wright J (2017). "Experimental Manipulation of Brood Size Affects Several Levels of Phenotypic Variance in Offspring and Parent Pied Flycatchers." *Behavioral Ecology and Sociobiology*, **71**(6), 91. ISSN 1432-0762. `doi:10.1007/s00265-017-2315-3`.

Westneat DF, Schofield M, Wright J (2012). "Parental Behavior Exhibits Among-Individual Variance, Plasticity, and Heterogeneous Residual Variance." *Behavioral Ecology*, **24**(3), 598–604. `doi:10.1093/beheco/ars207`.

Westneat DF, Wright J, Dingemanse NJ (2015). "The Biology Hidden Inside Residual Within-Individual Phenotypic Variation." *Biological Reviews*, **90**(3), 729–743. `doi:10.1093/beheco/ars207`.

Wickham H (2014). "Tidy Data." *Journal of Statistical Software*, **59**(10), 1–23. `doi:10.18637/jss.v059.i10`.

Wickham H (2016). ***ggplot2**: Elegant Graphics for Data Analysis.* Springer-Verlag.

Wickham H, François R, Henry L, Müller K (2021). ***dplyr**: A Grammar of Data Manipulation.* R package version 1.0.7, URL `https://CRAN.R-project.org/package=dplyr`.

Willig MR, Kaufman DM, Stevens RD (2003). "Latitudinal Gradients of Biodiversity: Pattern, Process, Scale, and Synthesis." *Review of Ecology, Evolution, and Systematics*, **34**(1), 273–309. `doi:10.1146/annurev.ecolsys.34.012103.144032`.

A. Results 1

A.1. Trace plots

This section includes the trace plots for the first model of load delivery fit to the pied fly-catchers data set in Section 5. In particular, Figure 1 shows the trace plots of the fixed effects for the mean, Figure 2 shows trace plots for the fixed effects of the dispersion, and Figure 3 shows trace plots of the standard deviations of the random effects included in both the mean and dispersion components.

A.2. Convergence diagnostics

Convergence diagnostics of the models fit in in Section 5. The values of the Brooks-Gelman-Rubin potential scale reduction factors obtained are:

```
R> pfconvergence$gelman

Potential scale reduction factors:

                          Point est. Upper C.I.
alpha.(Intercept)               1.00       1.00
alpha.log(IVI)                  1.00       1.00
alpha.broodsizeIncreased        1.00       1.00
alpha.sexMale                   1.00       1.00
```

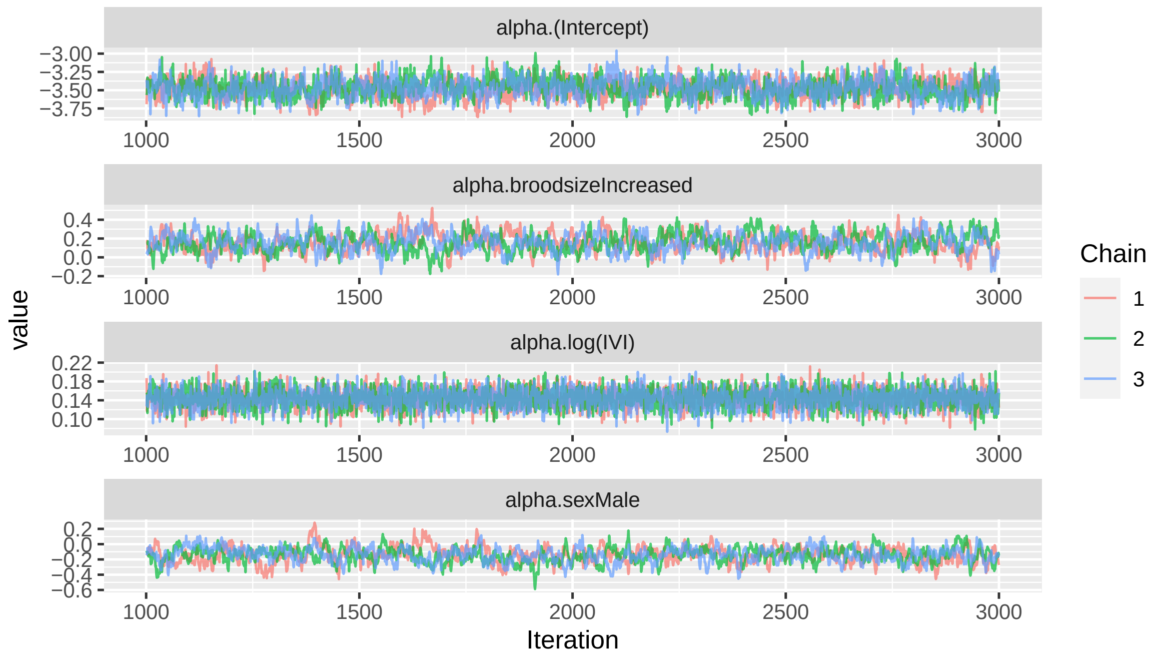

Figure 1: Trace plots for the fixed effects of the mean for the model of load delivery by the pied flycatcher parents.

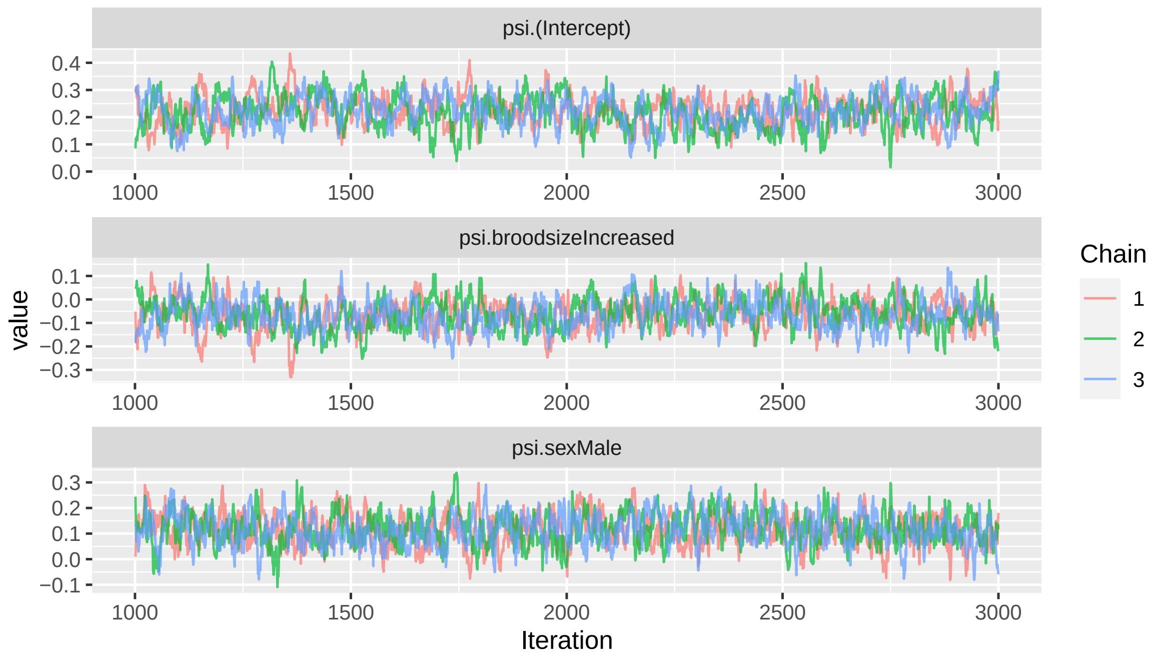

Figure 2: Trace plots for the fixed effects of the dispersion for the model of load delivery by the pied flycatcher parents.

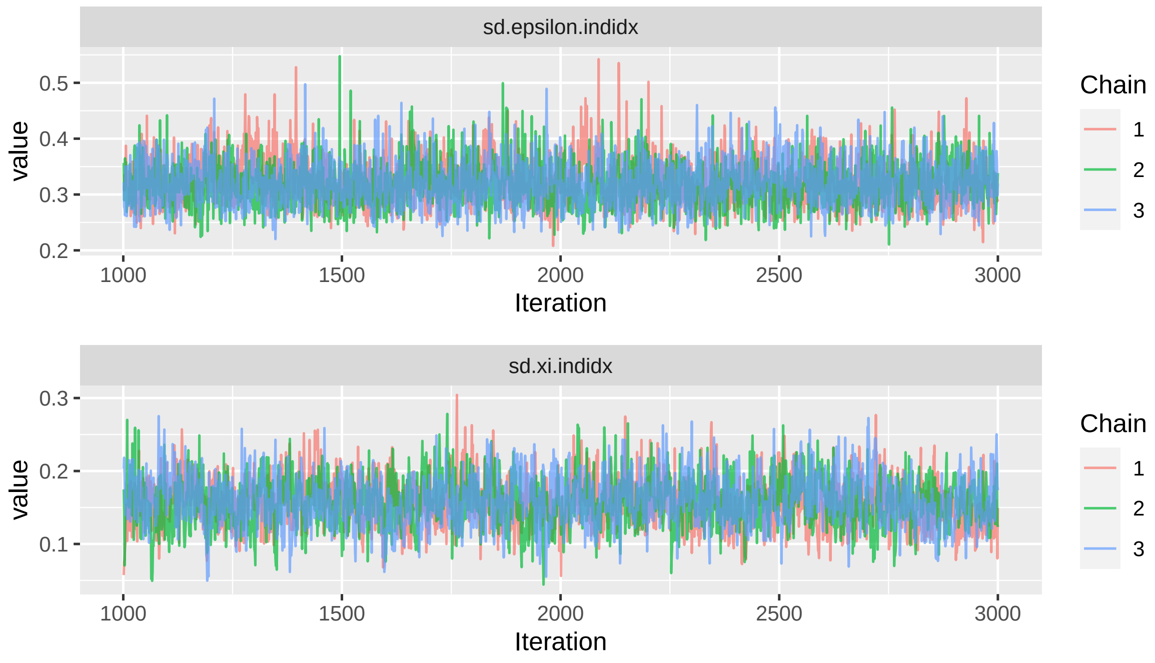

Figure 3: Trace plots for the standard deviations of the random effects of the mean (top) and dispersion (bottom) for the model of load delivery by the pied flycatcher parents.

```
psi.(Intercept)               1.01       1.02
psi.broodsizeIncreased        1.00       1.01
psi.sexMale                   1.00       1.01
sd.epsilon.indidx             1.00       1.01
sd.xi.indidx                  1.00       1.01
```

```
R> pfconvergence$raftery

Quantile (q) = 0.025
Accuracy (r) = +/- 0.005
Probability (s) = 0.95

                          Burn-in  Total Lower bound  Dependence
                          (M)      (N)   (Nmin)       factor (I)
 alpha.(Intercept)        4        5314  3746         1.42
 alpha.log(IVI)           2        3761  3746         1.00
 alpha.broodsizeIncreased 12       12620 3746         3.37
 alpha.sexMale            18       19347 3746         5.16
 psi.(Intercept)          18       20316 3746         5.42
 psi.broodsizeIncreased   22       23840 3746         6.36
 psi.sexMale              16       17218 3746         4.60
 sd.epsilon.indidx        3        4197  3746         1.12
 sd.xi.indidx             10       11968 3746         3.19
```

```
R> round(pfconvergence$effectiveSize)

       alpha.(Intercept)           alpha.log(IVI)
                     775                     4808
alpha.broodsizeIncreased            alpha.sexMale
                     416                      374
         psi.(Intercept)   psi.broodsizeIncreased
                     331                      422
             psi.sexMale        sd.epsilon.indidx
                     499                     2044
            sd.xi.indidx
                    1329
```

A.3. Numerical summaries

Numerical summaries of the model parameters for the model fit in Section 5:

```
Iterations = 1001:3000
Thinning interval = 1
Number of chains = 3
Sample size per chain = 2000
```

```
Posterior Summary Statistics for Each Model Component

Mean Model: Fixed Effects
                    Mean Median   SD Lower 95% Lower 50% Upper 50%
(Intercept)        -3.47  -3.47 0.13     -3.72     -3.56     -3.39
log(IVI)            0.14   0.14 0.02      0.11      0.13      0.16
broodsizeIncreased  0.16   0.15 0.10     -0.04      0.08      0.21
sexMale            -0.14  -0.14 0.10     -0.35     -0.20     -0.08
                   Upper 95%
(Intercept)            -3.21
log(IVI)                0.18
broodsizeIncreased      0.35
sexMale                 0.04

Mean Model: Random Effects
       Mean Median   SD Lower 95% Lower 50% Upper 50% Upper 95%
indidx 0.32   0.32 0.04      0.25      0.29      0.34       0.4

Dispersion Model: Fixed Effects
                    Mean Median   SD Lower 95% Lower 50% Upper 50%
(Intercept)         0.22   0.22 0.06      0.10      0.19      0.27
broodsizeIncreased -0.06  -0.06 0.06     -0.19     -0.10     -0.02
sexMale             0.12   0.11 0.06      0.00      0.08      0.16
                   Upper 95%
(Intercept)             0.32
broodsizeIncreased      0.06
sexMale                 0.24

Dispersion Model: Random Effects
       Mean Median   SD Lower 95% Lower 50% Upper 50% Upper 95%
indidx 0.16   0.16 0.03       0.1      0.13      0.17      0.23
```

A.4. Graphical summaries

This section includes graphical summaries of the model parameters obtained from the first model of load delivery fit to the pied flycatchers data set in Section 5. In particular, Figure 4 shows the caterpillar plots of the fixed effects of the mean and Figure 5 shows the fixed effects of the variance for the model of load delivery.

A.5. Fitted values

This section includes the graphical summaries of the fitted values produced by the model fit to the pied flycatchers data set in Section 5. Figure 6 shows the fitted values as a function of gender and brood size.

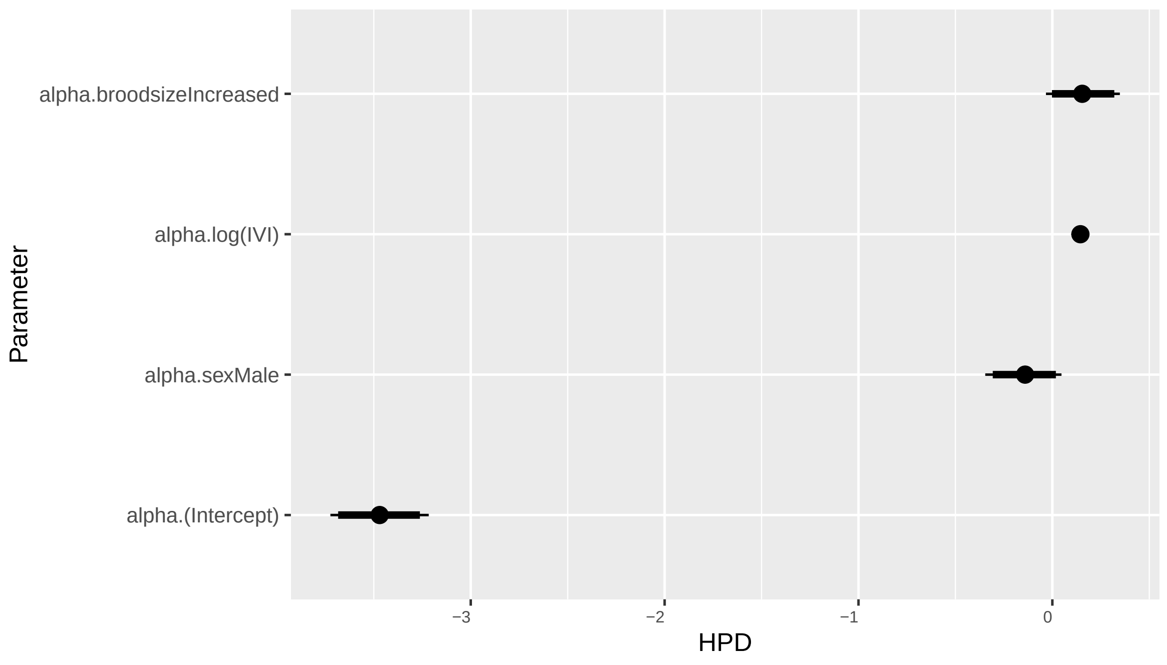

Figure 4: Caterpillar plots for the fixed effects of the mean for the model of load delivery by the pied flycatcher parents.

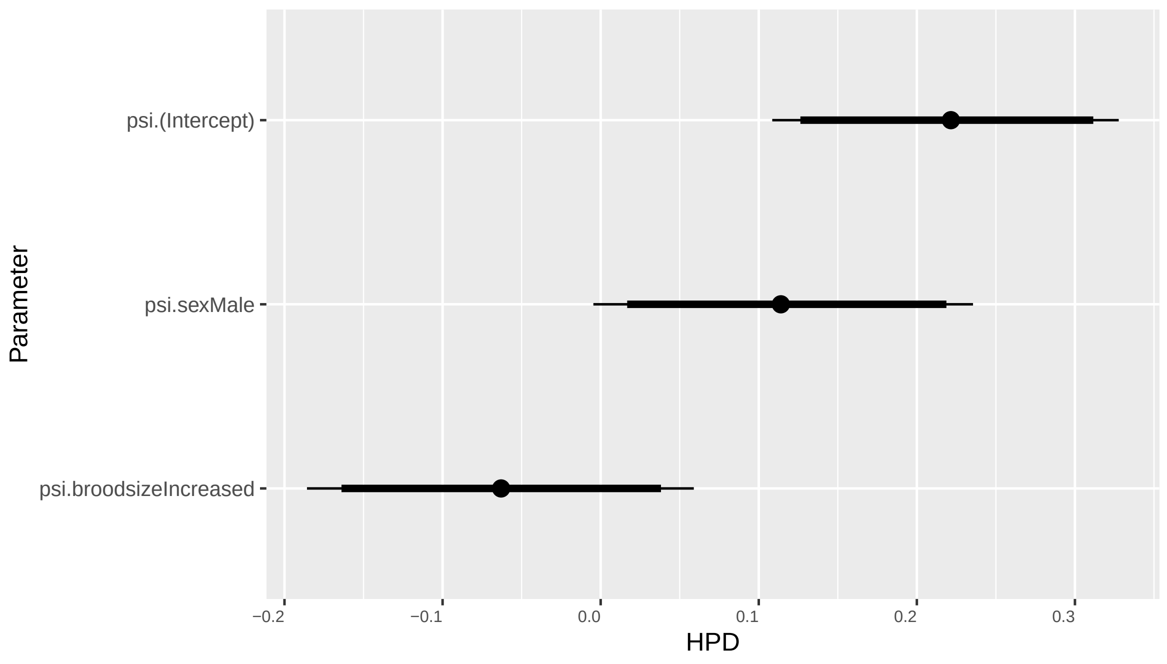

Figure 5: Caterpillar plots for the fixed effects of the variance for the model of load delivery by the pied flycatcher parents.

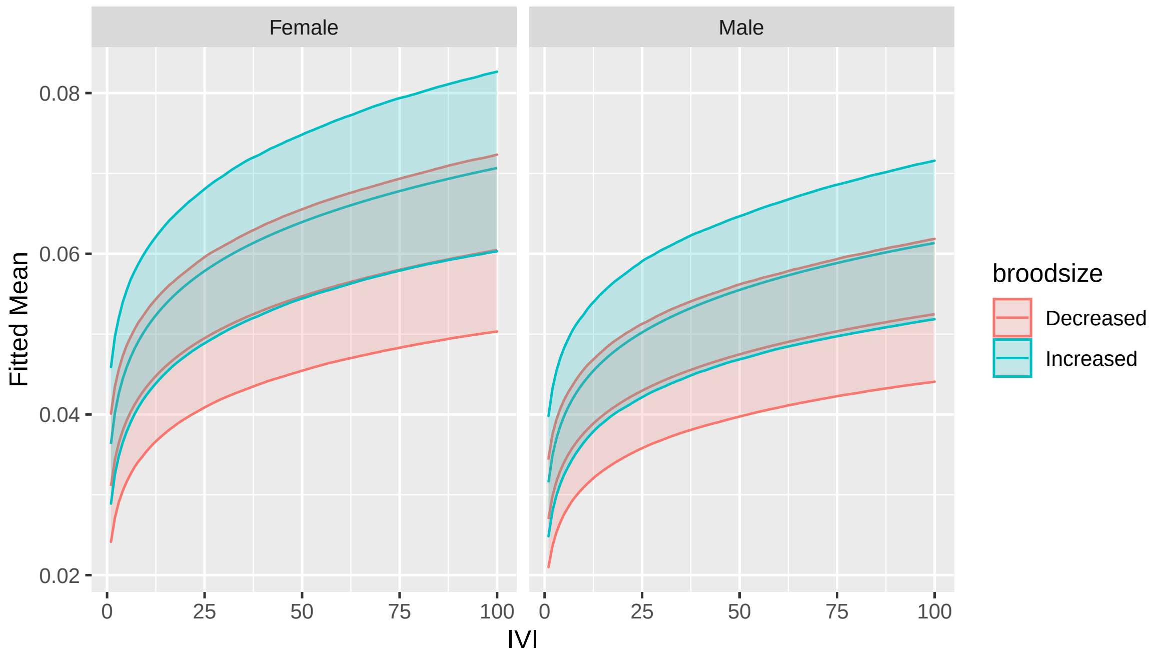

Figure 6: Fitted mean load as a function of IVI male and pied flycatchers with increased and decreased brood size.

B. Results 2: Accounting for rounding

Numerical summaries of the parameters for the model fit to censored response variables fit in Section 7.1 are as follows:

```
Iterations = 1001:3000
Thinning interval = 1
Number of chains = 3
Sample size per chain = 2000

Posterior Summary Statistics for Each Model Component

Mean Model: Fixed Effects
                    Mean Median   SD Lower 95% Lower 50% Upper 50%
(Intercept)        -3.08  -3.08 0.09     -3.26     -3.15     -3.03
log(IVI)            0.11   0.11 0.01      0.09      0.10      0.12
broodsizeIncreased  0.10   0.10 0.07     -0.03      0.06      0.15
sexMale            -0.11  -0.10 0.08     -0.25     -0.15     -0.05
                   Upper 95%
(Intercept)            -2.90
log(IVI)                0.14
broodsizeIncreased      0.24
sexMale                 0.04

Mean Model: Random Effects
```

```
       Mean Median   SD Lower 95% Lower 50% Upper 50% Upper 95%
indidx 0.23   0.23 0.03      0.18      0.21      0.24      0.29

Dispersion Model: Fixed Effects
                    Mean Median   SD Lower 95% Lower 50% Upper 50%
(Intercept)        -0.81  -0.81 0.08     -0.95     -0.86     -0.75
broodsizeIncreased  0.03   0.03 0.08     -0.13     -0.01      0.11
sexMale             0.10   0.10 0.08     -0.07      0.04      0.16
                   Upper 95%
(Intercept)            -0.64
broodsizeIncreased      0.19
sexMale                 0.25

Dispersion Model: Random Effects
       Mean Median   SD Lower 95% Lower 50% Upper 50% Upper 95%
indidx 0.21   0.21 0.05      0.12      0.17      0.23      0.31
```

C. Results 3: Load averaged by day

Numerical summaries of the parameters for the model fit using a weighted regression in Section 7.2 are as follows:

```
Iterations = 1001:3000
Thinning interval = 1
Number of chains = 3
Sample size per chain = 2000

Posterior Summary Statistics for Each Model Component

Mean Model: Fixed Effects
                    Mean Median   SD Lower 95% Lower 50% Upper 50%
(Intercept)        -5.03  -5.04 0.66     -6.32     -5.50     -4.61
log(avgIVI)         0.52   0.52 0.13      0.26      0.43      0.60
broodsizeIncreased  0.25   0.25 0.08      0.10      0.20      0.31
sexMale            -0.09  -0.09 0.07     -0.22     -0.14     -0.05
                   Upper 95%
(Intercept)            -3.70
log(avgIVI)             0.77
broodsizeIncreased      0.40
sexMale                 0.04

Dispersion Model: Fixed Effects
                    Mean Median   SD Lower 95% Lower 50% Upper 50%
(Intercept)         1.30   1.29 0.35      0.58      1.03      1.50
broodsizeIncreased  0.83   0.84 0.39      0.04      0.59      1.11
sexMale            -0.48  -0.48 0.41     -1.26     -0.77     -0.21
```

```
                        Upper 95%
(Intercept)                  1.99
broodsizeIncreased           1.58
sexMale                      0.34
```

Affiliation:

Simon Bonner, Han-Na Kim
University of Western Ontario
Department of Statistical and Actuarial Sciences
London, ON N6A 5B7, Canada
E-mail: sbonner6@uwo.ca, hkim787@uwo.ca

David Westneat
University of Kentucky
Department of Biology
101 T.H. Morgan Building Lexington KY 40506-022, United States of America
E-mail: biodfw@email.uky.edu

Ariane Mutzel
University of Kentucky
Department of Biology
101 T.H. Morgan Building Lexington KY 40506-022, United States of America
E-mail: ariane.mutzel@uky.edu

Jonathan Wright
NTNU
Department of Biology
Realfagbygget, DU1-184
Gløshaugen, Høgskoleringen 5, Norway
E-mail: jonathan.wright@ntnu.no

Matthew Schofield
University of Otago
Department of Mathematics & Statistics
PO Box 56
Dunedin 9054, New Zealand
E-mail: matthew.schofield@otago.ac.nz

Journal of Statistical Software http://www.jstatsoft.org/
published by the Foundation for Open Access Statistics http://www.foastat.org/
November 2021, Volume 100, Issue 10 *Submitted:* 2018-10-06
doi:10.18637/jss.v100.i10 *Accepted:* 2021-02-08