Journal of Statistical Software

April 2018, Volume 84, Issue 4. doi: 10.18637/jss.v084.i04

Multiple Response Variables Regression Models in R: The mcglm Package

Wagner Hugo Bonat
Paraná Federal University
University of Southern Denmark

Abstract

This article describes the R package **mcglm** implemented for fitting multivariate covariance generalized linear models (McGLMs). McGLMs provide a general statistical modeling framework for normal and non-normal multivariate data analysis, designed to handle multivariate response variables, along with a wide range of temporal and spatial correlation structures defined in terms of a covariance link function and a matrix linear predictor involving known symmetric matrices. The models take non-normality into account in the conventional way by means of a variance function, and the mean structure is modeled by means of a link function and a linear predictor. The models are fitted using an estimating function approach based on second-moment assumptions. This provides a unified approach to a wide variety of different types of response variables and covariance structures, including multivariate extensions of repeated measures, time series, longitudinal, genetic, spatial and spatio-temporal structures. The **mcglm** package allows a flexible specification of the mean and covariance structures, and explicitly deals with multivariate response variables, through a user friendly formula interface similar to the ordinary `glm` function. Illustrations in this article cover a wide range of applications from the traditional one response variable Gaussian mixed models to multivariate spatial models for areal data using the multivariate Tweedie distribution. Additional features, such as robust and bias-corrected standard errors for regression parameters, residual analysis, measures of goodness-of-fit and model selection using the score information criterion are discussed through six worked examples. The **mcglm** package is a full R implementation based on the **Matrix** package which provides efficient access to **BLAS** (basic linear algebra subroutines), **Lapack** (dense matrix), **TAUCS** (sparse matrix) and **UMFPACK** (sparse matrix) routines for efficient linear algebra in R.

Keywords: multivariate regression models, estimating functions, mixed outcomes, semi-parametric models, Tweedie distribution, Poisson-Tweedie distribution, R.

1. Introduction

The **mcglm** package (Bonat 2018) for R (R Core Team 2017) provides functions to fit and analyze multivariate covariance generalized linear models (McGLMs; Bonat and Jørgensen 2016). The package is designed to take full advantage of the modular specification of the models using a `glm` style interface. Additional features include robust and bias-corrected standard errors for regression parameters, measures of goodness-of-fit and residual analysis. The **mcglm** package is a full R implementation based on the **Matrix** (Bates and Maechler 2017) package which provides efficient access to C and Fortran 90 libraries such as **BLAS** (basic linear algebra subroutines), **Lapack** (dense matrix), **TAUCS** (sparse matrix) and **UMFPACK** (sparse matrix) routines for efficient linear algebra in R. Package **mcglm** is available from the Comprehensive R Archive Network (CRAN) at `https://CRAN.R-project.org/package=mcglm`.

McGLMs are fitted by means of quasi-likelihood and Pearson estimating functions, based on second-moment assumptions, and implemented in an efficient Newton scoring algorithm (Bonat and Jørgensen 2016; Jørgensen and Knudsen 2004). The mean structure for each response variable is specified in the orthodox way by means of a link function and a linear predictor. Similarly, the covariance structure for each response variable is defined in terms of a covariance link function combined with a matrix linear predictor involving known symmetric matrices. The models take non-normality into account by means of a variance function. Finally, the generalized Kronecker product (Martinez-Beneito 2013) is employed to specify the joint covariance matrix for all response variables involved in the model.

The main features of the McGLMs framework include the ability to deal with most common types of response variables, such as continuous, count, proportions and binary/binomial. Characteristics such as symmetry or asymmetry, excess zeros and overdispersion are easily handled by choosing a variance function. We can model many different types of dependence, such as those present in repeated measures, longitudinal, time series, genetic, spatial and spatio-temporal data by different specifications of the covariance link function and the matrix linear predictor. Furthermore, all these features extend to multivariate response variables using the generalized Kronecker product allowing to compute the correlation between response variables.

The analysis of non-normal multivariate data currently involves a choice between a considerable array of statistical modeling frameworks. Possible approaches include the generalized linear mixed models (GLMMs; Verbeke, Fieuws, Molenberghs, and Davidian 2014; Fieuws, Verbeke, and Molenberghs 2007), models based on copulas (Krupskii and Joe 2013) and in particular Gaussian copula marginal models (Masarotto and Varin 2012), hierarchical generalized linear models (Lee and Nelder 1996) and generalized estimating equations (Liang and Zeger 1986; Liang, Zeger, and Qaqish 1992) to cite a few.

Although all aforementioned methodologies can deal with multivariate response variables, currently software implementations have been focused on models for one dependent response variable. Furthermore, different types of dependence such as those present in repeated measures, longitudinal, genetic and spatial data may demand different software packages. It is interesting to note that well-established R packages such as **lme4** (Bates, Mächler, Bolker, and Walker 2015) and **nlme** (Pinheiro, Bates, DebRoy, Sarkar, and R Core Team 2017) do not deal with multivariate response variables. In the Bayesian context the flexible packages **INLA** (Lindgren and Rue 2015) and **MCMCpack** (Martin, Quinn, and Park 2011) do not

handle multivariate response variables, judging from the package documentation. The package **gcmr** (Masarotto and Varin 2017) implements Gaussian copula regression models in R. This package provides a rich set of models to model the covariance structure, but is limited to handle one response variable. Similarly, the package **hglm** (Rönnegård, Shen, and Alam 2010) provides functions to fit hierarchical generalized linear models along with a large set of models to describe the covariance structure, but is also limited to deal with one response variable. GEE models can be fitted in R by one of the following packages: **geeM** (McDaniel, Henderson, and Rathouz 2013), **gee** (Carey 2015) and **geepack** (Højsgaard, Halekoh, and Yan 2006). All of them are limited to deal with one response variable and the modeling of the covariance structure is currently restricted to making a selection from a short list of pre-specified covariance structures, such as auto-regression or compound symmetry. In R, there are at least two GLMMs packages that can deal with multivariate response variables, namely **MCMCglmm** (Hadfield 2010), which uses Markov chain Monte Carlo methods in the Bayesian framework, and the package **SabreR** (Crouchley 2012), which uses marginal likelihood but is limited to dealing with at most three response variables.

In SAS the **GLIMMIX** procedure for GLMMs deals with multivariate response variables but is limited to the exponential family of distributions and a few pre-determined covariance structures (SAS Institute 2011). Other software platforms for fitting generic random-effects models via Markov chain Monte Carlo methods, such as **JAGS** (Plummer 2003) **WinBUGS** (Lunn, Thomas, Best, and Spiegelhalter 2000) and **Stan** (Carpenter, Gelman, Hoffman, Lee, Goodrich, Betancourt, Brubaker, Guo, Li, and Riddell 2017), can deal with multivariate response variables but carry substantial overheads in terms of computational times and convergence checks, while being cumbersome to implement non-standard covariance structures and more general probability distributions, such as the Tweedie and Poisson-Tweedie distributions. These limitations of current software availability for joint mean-covariance modeling of multivariate response variables motivated us to develop an implementation of McGLMs and consequently the **mcglm** package.

The main goal of this article is to describe the functionalities of the **mcglm** package for fitting multivariate covariance generalized linear models. Through six worked examples, we show the flexibility of the package to deal with different types of dependence structures, arising for example in mixed models, repeated measures and longitudinal data analysis, as well as in spatial areal data. By using the score information criterion, we discuss the selection of the linear and matrix linear predictor components. More challenging examples, as the cases of mixed outcomes (Bonat 2017), multivariate Poisson-Tweedie models for count data (Jørgensen and Kokonendji 2016; Bonat, Olivero, Grande-Vega, Fárfan, and Fa 2017) and multivariate Tweedie models (Jørgensen 2013; Jørgensen and Lauritzen 2000) for spatial areal data are also discussed.

The article is organized as follows. Section 2 introduces the multivariate covariance generalized linear models with emphasis to the specification of the matrix linear predictor. Section 3 presents the fitting algorithm based on the estimating function approach. Section 4 provides some measures of goodness-of-fit. In particular, we discuss the generalized error sum of squares (`ESS`), the pseudo Gaussian log-likelihood (`plogLik`) and its extensions, the pseudo *Akaike* information criterion (`pAIC`) and the Kullback-Leibler (`pKLIC`) information criterion. Furthermore, the score information criterion (`SIC`) is presented for the selection of the components of the linear and matrix linear predictors. Section 5 introduces the R implementation discussing the main functions and methods available in the **mcglm** package. Section 6 illus-

trates the package usage through six worked examples. Finally, Section 7 presents a discussion and directions for future work on the improvement of the **mcglm** package.

2. Multivariate covariance generalized linear models

Let $\mathbf{Y}_{N\times R} = \{\boldsymbol{Y}_1, \ldots, \boldsymbol{Y}_R\}$ be an outcome matrix and let $\mathbf{M}_{N\times R} = \{\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_R\}$ denote the corresponding matrix of expected values. Let $\boldsymbol{\Sigma}_r$ denote the $N \times N$ covariance matrix within the outcome r for $r = 1, \ldots, R$. Similarly, let $\boldsymbol{\Sigma}_b$ be the $R \times R$ correlation matrix between outcomes. The McGLMs (Bonat and Jørgensen 2016) are defined by

$$\begin{aligned} \mathsf{E}(\mathbf{Y}) &= \mathbf{M} = \{g_1^{-1}(\boldsymbol{X}_1\boldsymbol{\beta}_1), \ldots, g_R^{-1}(\boldsymbol{X}_R\boldsymbol{\beta}_R)\} \\ \mathsf{VAR}(\mathbf{Y}) &= \boldsymbol{C} = \boldsymbol{\Sigma}_R \overset{G}{\otimes} \boldsymbol{\Sigma}_b \end{aligned}$$

where $\boldsymbol{\Sigma}_R \overset{G}{\otimes} \boldsymbol{\Sigma}_b = \text{Bdiag}(\tilde{\boldsymbol{\Sigma}}_1, \ldots, \tilde{\boldsymbol{\Sigma}}_R)(\boldsymbol{\Sigma}_b \otimes \boldsymbol{I})\text{Bdiag}(\tilde{\boldsymbol{\Sigma}}_1^\top, \ldots, \tilde{\boldsymbol{\Sigma}}_R^\top)$ is the generalized Kronecker product (Martinez-Beneito 2013). The matrix $\tilde{\boldsymbol{\Sigma}}_r$ denotes the lower triangular matrix of the Cholesky decomposition of $\boldsymbol{\Sigma}_r$. The operator Bdiag() denotes a block diagonal matrix and $\boldsymbol{I}$ denotes an $N \times N$ identity matrix. The functions $g_r()$ are orthodox link functions. Let $\boldsymbol{X}_r$ denote an $N \times k_r$ design matrix and $\boldsymbol{\beta}_r$ a $k_r \times 1$ regression parameter vector. For continuous, binary, binomial, proportions or indexes the covariance matrix within outcomes $\boldsymbol{\Sigma}_r$ is defined by

$$\boldsymbol{\Sigma}_r = \text{V}(\boldsymbol{\mu}_r; p_r)^{\frac{1}{2}}(\boldsymbol{\Omega}(\boldsymbol{\tau}_r))\text{V}(\boldsymbol{\mu}_r; p_r)^{\frac{1}{2}}.$$

Similarly, for count data the covariance matrix within outcomes takes the following form

$$\boldsymbol{\Sigma}_r = \text{diag}(\boldsymbol{\mu}_r) + \text{V}(\boldsymbol{\mu}_r; p_r)^{\frac{1}{2}}(\boldsymbol{\Omega}(\boldsymbol{\tau}_r))\text{V}(\boldsymbol{\mu}_r; p_r)^{\frac{1}{2}}, \tag{1}$$

where $\text{V}(\boldsymbol{\mu}_r; p_r) = \text{diag}(\vartheta(\boldsymbol{\mu}_r; p_r))$ denotes a diagonal matrix, whose main entries are given by the variance function $\vartheta(\cdot; p_r)$ applied element wise to the vector $\boldsymbol{\mu}_r$.

The variance function plays an important role in McGLMs, since different choices for $\vartheta(\cdot; p_r)$ imply different marginal outcome distributions. The **mcglm** package implements three set of variance functions. To deal with continuous outcomes the power variance function $\vartheta(\cdot; p_r) = \mu_r^{p_r}$ provides a flexible family of models, since it describes the Tweedie family of distributions that has as special cases the Gaussian ($p = 0$), Gamma ($p = 2$) and inverse Gaussian ($p = 3$) distributions (Jørgensen 1987, 1997). For handling binary, binomial or proportional data the extended binomial variance function defined by $\vartheta(\cdot; p_r) = \mu_r^{p_{r1}}(1 - \mu_r)^{p_{r2}}$ may be used. Finally, for modeling count outcomes the **mcglm** package implements the Poisson-Tweedie dispersion function (Jørgensen and Kokonendji 2016), i.e., $\vartheta(\cdot; p) = \mu + \tau\mu^p$, where τ is the dispersion parameter. Note that, since the dispersion parameter appears only in the second term, the covariance within outcomes takes the special form in Equation 1. The Poisson-Tweedie family of distributions provides a rich class of models to deal with count outcomes, since many important distributions appear as special cases, examples include the Hermite ($p = 0$), Neyman Type A ($p = 1$), negative binomial ($p = 2$) and Poisson-inverse Gaussian ($p = 3$) distributions.

We highlight that the dispersion function introduced by Jørgensen and Kokonendji (2016) is not a variance function in the sense of Jørgensen (1997), however, for practical data analysis both are completely analogous. In this paper we adopt the term variance function for both for

simplicity. It is justified, since in general all features attributed to variance functions extend easily to dispersion functions in our framework.

The power parameter p is important in the context of McGLMs, since for all variance functions discussed it is an index which distinguishes between important distributions. The algorithm we shall present in Section 3 allows us to estimate the power parameter, which works as an automatic distribution selection. It is important to highlight that given the second-moment specification of McGLMs the estimation of the power parameters requires that the mean vector varies, i.e., requires the presence of significant covariates in the linear predictor. The information for estimation of the power parameter comes from the relationship between the mean and variance, thus an enough variation in the mean vector is necessary for estimation of the power parameter.

The dispersion matrix $\boldsymbol{\Omega}(\boldsymbol{\tau}_r)$ describes the part of the covariance within outcomes that does not depend on the mean structure. Based on the ideas of Anderson (1973) and Pourahmadi (2000), Bonat and Jørgensen (2016) proposed to model the dispersion matrix using a matrix linear predictor combined with a covariance link function, i.e.,

$$h(\boldsymbol{\Omega}(\boldsymbol{\tau}_r)) = \tau_{r0} Z_{r0} + \cdots + \tau_{rD} Z_{rD}, \tag{2}$$

where $h()$ is the covariance link function, Z_{rd} with $d = 0, \ldots, D$ are known matrices reflecting the covariance structure within the response variable r, and $\boldsymbol{\tau}_r = (\tau_{r0}, \ldots, \tau_{rD})$ is a $(D+1)\times 1$ parameter vector. The **mcglm** package implements three covariance link functions, namely, the identity, inverse and exponential-matrix (Chiu, Leonard, and Tsui 1996).

An important feature of the McGLMs framework is that, whereas the regression parameter estimators depend relatively little on the form of the covariance structure, this is not so for the standard errors of the regression parameter estimators, which depend directly on the choice of covariance structure.

McGLMs are defined by three functions, namely, link, variance and covariance functions and a linear predictor and a matrix linear predictor for each response variable. Thus, there is no loss of generality to discuss the specification of the matrix linear predictor for the one response variable case. Demidenko (2013) showed that the covariance structure induced by the orthodox Gaussian linear mixed model is a linear covariance matrix, i.e., has the form of Equation 2. In this sense, McGLMs can been seen as an extension of Gaussian linear mixed models for handling non-Gaussian data. Furthermore, popular approaches to deal with longitudinal auto-correlated data, as the compound symmetry, moving average and first order auto-regressive approaches, are also covariance linear models. In what follows we discuss some of the possibilities for the specification of the matrix linear predictor in the context of longitudinal and spatial data analysis.

Often in the context of longitudinal data analysis the outcomes are collected for a set of independent groups or unit samples at multiple follow-up times. Bonat *et al.* (2017) discussed the analysis of multivariate longitudinal count models using McGLMs and following their notation, denote y_{go} an observation $o = 1, \ldots, O_g$ within the group $g = 1, \ldots, G$ and let $\boldsymbol{y}_g$ denote the O_g-dimensional vector of measurements from the gth group. The response variable vector is given by $\boldsymbol{Y} = (\boldsymbol{y}_1, \ldots, \boldsymbol{y}_G)^\top$. Let A_g denote an $O_g \times E$ design matrix composed of the values of E known covariates available to model the covariance structure. Furthermore, let $A_{g,\cdot e}$ denote the eth column of the matrix A_g. The main effect of the covariate e and the interaction effect between the covariates e and e' are included in the covariance model

through the symmetric matrices

$$A_g^e = A_{g,\cdot e} A_{g,\cdot e}^\top \quad \text{and} \quad A_g^{ee'} = A_{g,\cdot e} A_{g,\cdot e'}^\top + A_{g,\cdot e'} A_{g,\cdot e}^\top,$$

respectively. The matrices A_g^e and $A_g^{ee'}$ are group specific. To obtain the components of the matrix linear predictor for the entire response variable vector $\boldsymbol{Y}$, we assume independent groups. Thus, the components of the matrix linear predictor that measure the effect of the eth covariate and the interaction effect are respectively given by

$$Z_e = \text{Bdiag}(A_1^e, \ldots, A_G^e) \quad \text{and} \quad Z_{ee'} = \text{Bdiag}(A_1^{ee'}, \ldots, A_G^{ee'}), \tag{3}$$

where as before the operator Bdiag() denotes a block diagonal matrix. The matrices Z_e and $Z_{ee'}$ can be included as the Z_d's components in the matrix linear predictor, see Equation 2. These structures can be combined with any of the three covariance link functions available in the **mcglm** package. In particular, the combination with the identity covariance link function provides a straightforward extension of Gaussian linear mixed models to deal with non-Gaussian data. Note that these simple structures provide a way to introduce the effects of continuous and categorical covariates in the covariance model in a linear mixed models fashion. These structures can be easily constructed in R using the function `mc_mixed` of the **mcglm** package.

The compound symmetry or exchangeable structure is a popular choice in the analysis of repeated measures data, it is defined by a linear combination of an identity and a matrix of ones, i.e., for a particular group with three observations the matrix linear predictor is given by

$$\boldsymbol{\Omega}_g(\boldsymbol{\tau}) = \tau_0 \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} + \tau_1 \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}.$$

Note that, when the design matrix A_g contains the intercept, the first term $A_{g,\cdot 1} A_{g,\cdot 1}^\top$ corresponds to a matrix of ones, that combined with an identity matrix (`mc_id`) results in the compound symmetry model.

The moving average model of order p MA(p) is also a linear covariance model. The components of the matrix linear predictor associated with the MA(1) and MA(2) structures are given respectively by

$$A_1 = \begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix} \quad \text{and} \quad A_2 = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \end{bmatrix},$$

the components of these models can be created in R using the function `mc_ma` of the **mcglm** package. For longitudinal data analysis, we can use the inverse of the Euclidean distance or any other measure of distance between pairs of observations as a component of the matrix linear predictor, for example

$$A_1 = \begin{bmatrix} 0 & 1/d_{12} & 1/d_{13} \\ 1/d_{12} & 0 & 1/d_{23} \\ 1/d_{13} & 1/d_{23} & 0 \end{bmatrix},$$

where d_{ij} denotes the distance between the observations at time i and j. See, for example the function `mc_dist` for more details.

In the Bayesian context Gaussian Markov random fields (GMRF; Rue and Held 2005) are often employed for the analysis of times series and areal data. These models specify the precision matrix (i.e., the inverse of the covariance matrix) in the following way

$$\boldsymbol{\Omega}_r^{-1}(\tau_0, \rho) = \tau(\boldsymbol{D} - \rho \boldsymbol{W}), \tag{4}$$

where $\boldsymbol{W}$ is a neighborhood matrix and $\boldsymbol{D}$ is a diagonal matrix with the number of neighbors in the main diagonal. The model is parameterized by the precision τ and auto-correlation parameter ρ. The matrices $\boldsymbol{D}$ and $\boldsymbol{W}$ can model space, time and space-time interaction in a straightforward way, by using different neighborhood matrices. It is easy to see, that the model in Equation 4 is a linear covariance model using the inverse covariance link function, i.e.,

$$\boldsymbol{\Omega}_r^{-1}(\boldsymbol{\tau}) = \tau_0 \boldsymbol{D} + \tau_1 \boldsymbol{W},$$

where $\tau_0 = \tau$ and $\tau_1 = -\tau\rho$. In the Bayesian context, it is common to fix the auto-correlation parameter ρ at the value 1, which is the so-called intrinsic conditional auto-regressive model. For times series and longitudinal data the neighborhood structure is naturally specified by the time structure. For spatial data the function `tri2nb` of the **spdep** package (Bivand and Piras 2015; Bivand 2017) can be used to create the neighborhood matrix based on a vector of spatial coordinates. The functions `mc_rw` and `mc_car` provided by the **mcglm** package help to build the components of the matrix linear predictor associated with these models. Furthermore, the structures can be combined to deal with space-temporal data, see Bonat and Jørgensen (2016, Example 3).

Some additional fields for application of McGLMs include additive genetic models. Hadfield and Nakagawa (2010) showed that virtually all models used in the fields of quantitative genetic and phylogenetic in the Gaussian case are special specifications of the matrix linear predictor involving different types of known matrices, such as the additive genetic relatedness matrix. Examples include the phylogenetic meta-analysis, the taxonomic mixed model and the traditional animal models. In that case, the R package **nadiv** (Wolak 2012) can be useful for creating the components of the matrix linear predictor associated with these models. Furthermore, these components can easily be incorporated in the **mcglm** package, as for example in Bonat (2017). The analysis of twin and family data can be done and extended by a suitable specification of the matrix linear predictor (Rabe-Hesketh, Skrondal, and Gjessing 2008).

3. Estimation and inference

McGLMs are fitted based on the estimating function approach described in detail by Bonat and Jørgensen (2016) and Jørgensen and Knudsen (2004). In this section we present a general overview of the algorithm and the asymptotic distribution of the estimating function estimators. The second-moment assumptions of McGLMs motivate us to divide the set of parameters into two subsets $\boldsymbol{\theta} = (\boldsymbol{\beta}^\top, \boldsymbol{\lambda}^\top)^\top$. In this notation $\boldsymbol{\beta} = (\boldsymbol{\beta}_1^\top, \ldots, \boldsymbol{\beta}_R^\top)^\top$ and $\boldsymbol{\lambda} = (\rho_1, \ldots, \rho_{R(R-1)/2}, p_1, \ldots, p_R, \boldsymbol{\tau}_1^\top, \ldots, \boldsymbol{\tau}_R^\top)^\top$ denote a $K \times 1$ and $Q \times 1$ vector of all regression and dispersion parameters, respectively.

Let $\mathcal{Y} = (\boldsymbol{Y}_1^\top, \ldots, \boldsymbol{Y}_R^\top)^\top$ and $\mathcal{M} = (\boldsymbol{\mu}_1^\top, \ldots, \boldsymbol{\mu}_R^\top)^\top$ denote the $NR \times 1$ stacked vector of the response variable matrix $\mathbf{Y}_{N\times R}$ and expected values matrix $\mathbf{M}_{N\times R}$ by columns, respectively.

The quasi-score function (Liang and Zeger 1986) was adopted for the regression parameters

$$\psi_{\boldsymbol{\beta}}(\boldsymbol{\beta}, \boldsymbol{\lambda}) = \boldsymbol{D}^\top \boldsymbol{C}^{-1}(\mathcal{Y} - \mathcal{M}),$$

where $\boldsymbol{D} = \nabla_{\boldsymbol{\beta}}\mathcal{M}$ is an $NR \times K$ matrix, and $\nabla_{\boldsymbol{\beta}}$ denotes the gradient operator. The $K \times K$ *sensitivity* and *variability* matrices of $\psi_{\boldsymbol{\beta}}$ are respectively given by

$$\mathrm{S}_{\boldsymbol{\beta}} = \mathsf{E}(\nabla_{\boldsymbol{\beta}}\psi_{\boldsymbol{\beta}}) = -\boldsymbol{D}^\top \boldsymbol{C}^{-1}\boldsymbol{D} \quad \text{and} \quad \mathrm{V}_{\boldsymbol{\beta}} = \mathsf{VAR}(\psi_{\boldsymbol{\beta}}) = \boldsymbol{D}^\top \boldsymbol{C}^{-1}\boldsymbol{D}.$$

The Pearson estimating function, defined by the components

$$\psi_{\boldsymbol{\lambda}_i}(\boldsymbol{\beta}, \boldsymbol{\lambda}) = \mathrm{tr}(W_{\boldsymbol{\lambda}_i}(\boldsymbol{r}^\top \boldsymbol{r} - \boldsymbol{C})) \quad \text{for} \quad i = 1, \ldots, Q,$$

where $W_{\boldsymbol{\lambda}_i} = -\partial \boldsymbol{C}^{-1}/\partial \boldsymbol{\lambda}_i$ and $\boldsymbol{r} = \mathcal{Y} - \mathcal{M}$, was adopted for the dispersion parameters.
The entry (i, j) of the $Q \times Q$ sensitivity matrix of $\psi_{\boldsymbol{\lambda}}$ is given by,

$$\mathrm{S}_{\boldsymbol{\lambda}_{ij}} = \mathsf{E}\left(\frac{\partial}{\partial \boldsymbol{\lambda}_i}\psi_{\boldsymbol{\lambda}_j}\right) = -\mathrm{tr}\left(W_{\boldsymbol{\lambda}_i}\boldsymbol{C}W_{\boldsymbol{\lambda}_j}\boldsymbol{C}\right).$$

The entry (i, j) of the $Q \times Q$ variability matrix of $\psi_{\boldsymbol{\lambda}}$ is given by

$$\mathrm{V}_{\boldsymbol{\lambda}_{ij}} = \mathrm{Cov}(\psi_{\boldsymbol{\lambda}_i}, \psi_{\boldsymbol{\lambda}_j}) = 2\mathrm{tr}(W_{\boldsymbol{\lambda}_i}\boldsymbol{C}W_{\boldsymbol{\lambda}_j}\boldsymbol{C}) + \sum_{l=1}^{NR} k_l^{(4)}(W_{\boldsymbol{\lambda}_i})_{ll}(W_{\boldsymbol{\lambda}_j})_{ll},$$

where $k_l^{(4)}$ denotes the fourth cumulant of $\mathcal{Y}_l$, for which we use the empirical version.
To take into account the covariance between the vectors $\boldsymbol{\beta}$ and $\boldsymbol{\lambda}$, Bonat and Jørgensen (2016) computed the *cross-sensitivity* and *cross-variability* matrices. Denote these quantities by $\mathrm{S}_{\boldsymbol{\lambda}\boldsymbol{\beta}}$, $\mathrm{S}_{\boldsymbol{\beta}\boldsymbol{\lambda}}$ and $\mathrm{V}_{\boldsymbol{\lambda}\boldsymbol{\beta}}$, respectively. For details see Bonat and Jørgensen (2016, Section 3).
The joint sensitivity and variability matrices of $\psi_{\boldsymbol{\beta}}$ and $\psi_{\boldsymbol{\lambda}}$ are given by

$$\mathrm{S}_{\boldsymbol{\theta}} = \begin{pmatrix} \mathrm{S}_{\boldsymbol{\beta}} & \mathrm{S}_{\boldsymbol{\beta}\boldsymbol{\lambda}} \\ \mathrm{S}_{\boldsymbol{\lambda}\boldsymbol{\beta}} & \mathrm{S}_{\boldsymbol{\lambda}} \end{pmatrix} \quad \text{and} \quad \mathrm{V}_{\boldsymbol{\theta}} = \begin{pmatrix} \mathrm{V}_{\boldsymbol{\beta}} & \mathrm{V}_{\boldsymbol{\lambda}\boldsymbol{\beta}}^\top \\ \mathrm{V}_{\boldsymbol{\lambda}\boldsymbol{\beta}} & \mathrm{V}_{\boldsymbol{\lambda}} \end{pmatrix}.$$

Let $\hat{\boldsymbol{\theta}} = (\hat{\boldsymbol{\beta}}^\top, \hat{\boldsymbol{\lambda}}^\top)^\top$ be the estimating function estimator of $\boldsymbol{\theta}$. Then the asymptotic distribution of $\hat{\boldsymbol{\theta}}$ is

$$\hat{\boldsymbol{\theta}} \sim \mathrm{N}(\boldsymbol{\theta}, \mathrm{J}_{\boldsymbol{\theta}}^{-1}),$$

where $\mathrm{J}_{\boldsymbol{\theta}}^{-1}$ is the inverse of the Godambe information matrix,

$$\mathrm{J}_{\boldsymbol{\theta}}^{-1} = \mathrm{S}_{\boldsymbol{\theta}}^{-1}\mathrm{V}_{\boldsymbol{\theta}}\mathrm{S}_{\boldsymbol{\theta}}^{-\top},$$

where $\mathrm{S}_{\boldsymbol{\theta}}^{-\top} = (\mathrm{S}_{\boldsymbol{\theta}}^{-1})^\top$.
The *modified chaser* algorithm proposed by Jørgensen and Knudsen (2004) is employed to solve the system of equations $\psi_{\boldsymbol{\beta}} = \mathbf{0}$ and $\psi_{\boldsymbol{\lambda}} = \mathbf{0}$, defined by

$$\begin{aligned} \boldsymbol{\beta}^{(i+1)} &= \boldsymbol{\beta}^{(i)} - \mathrm{S}_{\boldsymbol{\beta}}^{-1}\psi_{\boldsymbol{\beta}}(\boldsymbol{\beta}^{(i)}, \boldsymbol{\lambda}^{(i)}), \\ \boldsymbol{\lambda}^{(i+1)} &= \boldsymbol{\lambda}^{(i)} - \alpha\mathrm{S}_{\boldsymbol{\lambda}}^{-1}\psi_{\boldsymbol{\lambda}}(\boldsymbol{\beta}^{(i+1)}, \boldsymbol{\lambda}^{(i)}). \end{aligned} \tag{5}$$

In the **mcglm** package, we introduce the extra *tuning constant* α to control the step length. Note that the modified chaser algorithm uses the insensitivity property (Jørgensen and Knudsen 2004), which allows us to use two separate equations to update $\boldsymbol{\beta}$ and $\boldsymbol{\lambda}$. Bonat and Jørgensen (2016) also proposed the *reciprocal likelihood* algorithm that replaces the second equation in Equation 5 by

$$\boldsymbol{\lambda}^{(i+1)} = \boldsymbol{\lambda}^{(i)} - [\alpha\psi_{\boldsymbol{\lambda}}(\boldsymbol{\beta}^{(i+1)}, \boldsymbol{\lambda}^{(i)})^{\top}\psi_{\boldsymbol{\lambda}}(\boldsymbol{\beta}^{(i+1)}, \boldsymbol{\lambda}^{(i)})\mathrm{V}_{\boldsymbol{\lambda}}^{-1}\mathrm{S}_{\boldsymbol{\lambda}} + \mathrm{S}_{\boldsymbol{\lambda}}]^{-1}\psi_{\boldsymbol{\lambda}}(\boldsymbol{\beta}^{(i+1)}, \boldsymbol{\lambda}^{(i)}),$$

where again we introduce the *tuning constant* α to control the step length. The **mcglm** package implements both the *modified chaser* and *reciprocal likelihood* algorithms as well as allows the user to control the step length through the argument `alpha` (see Section 5). The **mcglm** package also implements the bias-correction term proposed by Jørgensen and Knudsen (2004). Details about how to compute the derivatives in the weight matrices required in the Pearson estimating function can be found in Bonat and Jørgensen (2016).

4. Measures of goodness-of-fit

Model selection is an important issue in almost any practical data analysis. A common problem is variable selection in regression: given a large group of covariates (including some higher order terms), one needs to select a subset to be included in the regression model. To face this problem, Stoklosa, Gibb, and Warton (2014) in the context of generalized estimating equations (GEE) proposed the score information criterion (SIC) to be used with forward selection algorithms in the cases where we have a large number of covariates to compose the linear predictor. Bonat *et al.* (2017) extended the SIC to select the components of the matrix linear predictor. The SIC is based on the score statistics. This makes the use of this criterion convenient, since it can be computed for all candidate models without actually fitting them. In this section, we describe the SIC with the emphasis placed on the selection of the components of the matrix linear predictor. It is straightforward to apply the same arguments for the selection of the components of the linear predictor.

Suppose without loss of generality that $r = 1$ and that the power parameter is fixed. In that case, the vector of dispersion parameters simplifies to $\boldsymbol{\lambda} = \boldsymbol{\tau}$. For a given mean structure, suppose that the parameter vector $\boldsymbol{\tau}$ can be partitioned as $\boldsymbol{\tau} = (\boldsymbol{\tau}_1^{\top}, \boldsymbol{\tau}_2^{\top})^{\top}$, whose dimensions are $(Q - s) \times 1$ and $s \times 1$, respectively. The Pearson estimating function $\psi_{\boldsymbol{\lambda}}$ and its sensitivity and variability matrices, can also be partitioned to $\psi_{\boldsymbol{\lambda}}(\boldsymbol{\beta}, \boldsymbol{\tau}) = (\psi_{\boldsymbol{\lambda}_1}(\boldsymbol{\beta}, \boldsymbol{\tau}_1)^{\top}, \psi_{\boldsymbol{\lambda}_2}(\boldsymbol{\beta}, \boldsymbol{\tau}_2)^{\top})^{\top}$,

$$\mathrm{S}_{\boldsymbol{\lambda}} = \begin{pmatrix} \mathrm{S}_{\boldsymbol{\lambda}_{11}} & \mathrm{S}_{\boldsymbol{\lambda}_{12}} \\ \mathrm{S}_{\boldsymbol{\lambda}_{21}} & \mathrm{S}_{\boldsymbol{\lambda}_{22}} \end{pmatrix} \quad \text{and} \quad \mathrm{V}_{\boldsymbol{\lambda}} = \begin{pmatrix} \mathrm{V}_{\boldsymbol{\lambda}_{11}} & \mathrm{V}_{\boldsymbol{\lambda}_{12}} \\ \mathrm{V}_{\boldsymbol{\lambda}_{21}} & \mathrm{V}_{\boldsymbol{\lambda}_{22}} \end{pmatrix},$$

respectively. The null hypothesis H_0 is $\boldsymbol{\tau}_2 = \mathbf{0}$. Let $\tilde{\boldsymbol{\tau}} = (\hat{\boldsymbol{\tau}}_1^{\top}, \mathbf{0}^{\top})^{\top}$ be the vector of Pearson estimates under H_0. Note that only the base model containing $\hat{\boldsymbol{\tau}}_1$ parameters has to be fitted. In practical situations, this model can contain only a simple intercept. The Pearson estimating function takes the form

$$\psi_{\boldsymbol{\lambda}}(\beta, \tilde{\boldsymbol{\tau}}) = (\psi_{\boldsymbol{\lambda}_1}^{\top}(\boldsymbol{\beta}, \tilde{\boldsymbol{\tau}}), \psi_{\boldsymbol{\lambda}_2}^{\top}(\boldsymbol{\beta}, \tilde{\boldsymbol{\tau}}))^{\top} = (\mathbf{0}^{\top}, \psi_{\boldsymbol{\lambda}_2}^{\top}(\boldsymbol{\beta}, \tilde{\boldsymbol{\tau}}))^{\top}.$$

The generalized score statistic is given by

$$T_{\boldsymbol{\lambda}_2}(\boldsymbol{\beta}, \tilde{\boldsymbol{\tau}}) = \psi_{\boldsymbol{\lambda}_2}^{\top}(\boldsymbol{\beta}, \tilde{\boldsymbol{\tau}})\mathsf{VAR}(\psi_{\boldsymbol{\lambda}_2}(\boldsymbol{\beta}, \tilde{\boldsymbol{\tau}}))^{-1}\psi_{\boldsymbol{\lambda}_2}(\boldsymbol{\beta}, \tilde{\boldsymbol{\tau}}), \tag{6}$$

where

$$\mathsf{VAR}(\psi_{\boldsymbol{\lambda}_2}(\boldsymbol{\beta}, \tilde{\boldsymbol{\tau}})) = \mathrm{V}_{\boldsymbol{\lambda}_{22}} - \mathrm{S}_{\boldsymbol{\lambda}_{21}}\mathrm{S}_{\boldsymbol{\lambda}_{11}}^{-1}\mathrm{V}_{\boldsymbol{\lambda}_{12}} - \mathrm{V}_{\boldsymbol{\lambda}_{12}}\mathrm{S}_{\boldsymbol{\lambda}_{11}}^{-1}\mathrm{S}_{\boldsymbol{\lambda}_{12}} + \mathrm{S}_{\boldsymbol{\lambda}_{21}}\mathrm{S}_{\boldsymbol{\lambda}_{11}}^{-1}\mathrm{V}_{\boldsymbol{\lambda}_{11}}\mathrm{S}_{\boldsymbol{\lambda}_{11}}^{-1}\mathrm{S}_{\boldsymbol{\lambda}_{12}}$$

is the variance of the sub-vector $\psi_{\boldsymbol{\lambda}_2}(\boldsymbol{\beta}, \tilde{\boldsymbol{\tau}})$. Under the null hypothesis, $T_{\boldsymbol{\lambda}_2}(\boldsymbol{\beta}, \tilde{\boldsymbol{\tau}})$ has a chi-square distribution with s degrees of freedom. In practice, all quantities in Equation 6 are evaluated at the Pearson estimates under the null hypothesis. If H_0 were true, then $\psi_{\boldsymbol{\lambda}_2}(\boldsymbol{\beta}, \tilde{\boldsymbol{\tau}})$, that is the Pearson estimating function for $\boldsymbol{\tau}_2$, would be close to zero when evaluated under the null. Large values of $T_{\boldsymbol{\lambda}_2}(\boldsymbol{\beta}, \tilde{\boldsymbol{\tau}})$ would argue against H_0. The main idea behind the SIC is to use Equation 6 as a quadratic approximation to the log-likelihood ratio statistic. The SIC is defined by

$$\mathrm{SIC}^{(1)}(\boldsymbol{\beta}, \boldsymbol{\tau}) = -T_{\boldsymbol{\lambda}_2}(\boldsymbol{\beta}, \tilde{\boldsymbol{\tau}}) + \delta|\boldsymbol{\tau}|.$$

Note that this criterion is a function of $\tilde{\boldsymbol{\tau}}$ only, thus only the base model needs to be fitted. The SIC can be combined with the Wald statistic in a stepwise procedure, as described in Bonat *et al.* (2017). The **mcglm** package implements the SIC for selection of the linear and matrix linear predictors components, by the functions `mc_sic` and `mc_sic_covariance`. The user can use the penalty $\delta = 2$, as it is analogous to the *Akaike* information criterion. It is also possible to use $\delta = \log(N)$ to have an analogous to the *Bayesian* information criterion or any other penalty specified by the user.

The SIC is an important tool to assist with the selection of the linear and matrix linear predictors components, but it is less useful for comparing models fitted using different link, variance or covariance functions. Furthermore, the application of the SIC to compare non-nested models can be difficult, because of the requirement to have a base model for comparison. Thus, it is useful to have measures of goodness-of-fit similar to the *Akaike* information criterion in the context of likelihood analysis.

Carey and Wang (2011) proposed the Gaussian pseudo log-likelihood (`plogLik`) given by

$$\mathrm{plogLik}(\boldsymbol{\theta}) = -\frac{NR}{2}\log(2\pi) - \frac{1}{2}\log|\hat{\boldsymbol{C}}| - (\mathcal{Y} - \hat{\mathcal{M}})^{\top}\hat{\boldsymbol{C}}^{-1}(\mathcal{Y} - \hat{\mathcal{M}}),$$

where NR is the total number of observations, $\hat{\mathcal{M}}$ and $\hat{\boldsymbol{C}}$ denote the estimated vector of expected values and covariance matrix, respectively.

We combined three penalty terms with the Gaussian pseudo log-likelihood in order to have analogs to the *Akaike*, *Bayesian* and Kullback-Leibler information criterion. Thus, the pseudo *Akaike* information criterion (`pAIC`) is given by

$$\mathrm{pAIC}(\boldsymbol{\theta}) = 2(P + Q) - 2\mathrm{plogLik}(\boldsymbol{\theta}),$$

where P is the number of regression parameters, and Q is the number of dispersion parameters. Similarly, the pseudo *Bayesian* information criterion (`pBIC`) is obtained by

$$\mathrm{pBIC}(\boldsymbol{\theta}) = \log NR(P + Q) - 2\mathrm{plogLik}(\boldsymbol{\theta}).$$

Finally, the pseudo Kullback-Leibler information criterion (`pKLIC`) is given by

$$\mathrm{pKLIC}(\boldsymbol{\theta}) = 2\mathrm{tr}(\mathrm{S}_{\boldsymbol{\theta}}^{-1}\mathrm{V}_{\boldsymbol{\theta}}) - 2\mathrm{plogLik}(\boldsymbol{\theta}),$$

where $\mathrm{S}_{\boldsymbol{\theta}}$ and $\mathrm{V}_{\boldsymbol{\theta}}$ are the joint sensitivity and variability matrices defined in Section 3. These measures of goodness-of-fit are implemented in the **mcglm** package and can be accessed together using the function `gof` or individually by the functions indicated in brackets.

5. Implementation in R

The main function of the **mcglm** package is `mcglm()` which allows to fit multivariate covariance generalized linear models using quasi-score and Pearson estimating functions. The synopsis of the `mcglm()` function is:

```
mcglm(linear_pred, matrix_pred, link, variance, covariance, offset, Ntrial,
  power_fixed, data, control_initial = "automatic", contrasts = NULL,
  control_algorithm = list())
```

The argument `linear_pred` specifies the linear predictor for each response variable using a list of '`formula`' objects. For example, in a bivariate case with response variables `y1` and `y2` and a covariate `x1`, the `linear_pred` argument is specified as `linear_pred = c(y1 ~ x1, y2 ~ x1)`. Similarly, the argument `matrix_pred` specifies the matrix linear predictor for each response variable, but now using a list of known matrices. The **mcglm** package provides some pre-specified covariance structures (see Section 2), but the users can use any symmetric matrix as a component of the matrix linear predictor. Consider a simple longitudinal structure with two unit samples (`id`) and two replications (`time`) as follows:

```
R> data <- data.frame("id" = gl(2, 2), "time" = rep(1:2, 2))
```

The components of the matrix linear predictor, for example, for a moving average first order model MA(1) are given by:

```
R> Z0 <- mc_id(data)
R> Z1 <- mc_ma(id = "id", time = "time", data = data, order = 1)
```

In this way, for the bivariate case considered above the `matrix_pred` argument is specified by:

```
R> matrix_pred <- list(c(Z0, Z1), c(Z0, Z1))
```

The arguments `link`, `variance` and `covariance` specify, respectively, the link, variance and covariance link functions, for each response variable. The following link functions are implemented in the **mcglm** package: `"logit"`, `"probit"`, `"cauchit"`, `"cloglog"`, `"loglog"`, `"identity"`, `"log"`, `"sqrt"` and `"inverse"`. For the variance functions the options are: `"constant"`, `"tweedie"`, `"binomialP"`, `"binomialPQ"` and `"poisson_tweedie"`. Finally, the **mcglm** package implements three covariance link functions, namely, `"identity"`, `"inverse"` and `"expm"` (exponential-matrix). For the bivariate case under consideration, the specification can be as follows:

```
R> link <- c("log", "logit")
R> variance <- c("poisson_tweedie", "binomialP")
R> covariance <- c("identity", "inverse")
```

In that case, we are using for the first response variable the logarithm, Poisson-Tweedie and identity functions as the link, variance and covariance functions. Similarly, for the second response variable the logit, binomial and inverse are the link, variance and covariance functions. This specification is suitable, for example, for the mix of count and binomial outcomes.

Functions	Description
`print()`	Simple printed display of model features.
`summary()`	Standard regression output.
`fitted()`	Fitted values for observed data.
`residuals()`	Pearson, raw and standardized residuals.
`coef()`	Coefficient estimates.
`vcov()`	Variance-covariance matrix of coefficient estimates.
`confint()`	Confidence intervals.
`anova()`	Analysis of variance tables for fitted models.
`plot()`	Diagnostic plots of Pearson residuals and algorithm check.

Table 1: Methods available for objects of class '`mcglm`'.

The users can specify an offset for continuous or count data as well as the number of trials for binomial data through the arguments `offset` and `Ntrial`, respectively. In our bivariate case, suppose we have an offset available for the count response variable and also the number of trials for the binomial response variable. The specification of the arguments `offset` and `Ntrial` should be as follows:

```
R> Ntrial <- list(NULL, data$Ntrial)
R> offset <- list(data$offset, NULL)
```

For variance functions that depend on a power parameter, the users can either fix it using `power_fixed = TRUE` or estimate it using `power_fixed = FALSE`. For more than one response variable the usage extends easily to `power_fixed = c(TRUE, FALSE)`, where we fix the power parameter for the first response variable while estimating it for the second one. The value of the power parameter is fixed at the initial value supplied to the argument `control_initial`. If no initial values are supplied the function `mc_initial_values` is called. For the power parameters the automatic initial values are 1 for all variance functions. For more details, see `?mc_initial_values`. The data set is supplied through the argument `data`. The argument `contrast` is passed to the `model.matrix` function for specifying different contrasts for categorical covariates.

The argument `control_algorithm` should be a named list controlling features of the fitting algorithm. The options are: `correct`, a logical indicating if the corrected Pearson estimating function should be used; `max_iter`, the maximum number of iterations; `tol`, the convergence tolerance; `method`, a string indicating which algorithm should be used, i.e., *modified chaser* (`"chaser"`) or *reciprocal likelihood* (`"rc"`). The tuning constant is specified by the option `tuning`. This should in general be a number smaller than 1. Finally, the logical `verbose` specifies if the trace of the algorithm iterations should be printed.

The returned fitted model object of class '`mcglm`' is a list that contains, among others, the estimating function estimates, the sensitivity and the variability matrices. A set of standard methods is available to extract information from the fitted model, see Table 5. Most of the functions and methods use a standard syntax as implemented in other R packages oriented towards regression analysis.

The **mcglm** package implements some extra features and facilities useful for data analysis. For repeated measures and longitudinal data analysis the bias-corrected and robust standard

errors for regression parameters presented by Nuamah, Qu, and Amini (1996) are available through the functions `mc_bias_corrected_std` and `mc_robust_std`. The measures of goodness-of-fit discussed in Section 4 are available in the function `gof`. The selection of the components of linear and matrix linear predictors can be done using the functions `mc_sic` and `mc_sic_covariance`. Finally, conditional hypothesis tests can be performed with the function `mc_conditional_test`.

6. Examples

In this section, the usage of the **mcglm** package is illustrated with six worked examples covering various forms of dependent and multivariate data frequently arising in real applications.

6.1. Gaussian mixed models

The first example corresponds to Gaussian mixed models. We use the `sleepstudy` data set available in the **lme4** package. The data set contains the average reaction time per day for subjects in a sleep deprivation study. On day 0 the subjects had their normal amount of sleep. Starting that night they were restricted to three hours of sleep per night. The response variable is the average reaction time on a series of tests given each day to each subject. The analysis goal is to understand how sleep deprivation affects the average reaction times. The dependence structure is introduced by observations taken for the same subject (repeated measures) and on sequential days (longitudinal). The analysis using the **lme4** package is given by:

```
R> library("lme4")
R> data("sleepstudy", package = "lme4")
R> fit1.lme4 <- lmer(Reaction ~ Days + (Days | Subject), sleepstudy,
+    REML = FALSE)
```

Note that the fitted model has random intercepts and slopes associated with the covariate `Days`. The argument `REML = FALSE` indicates that we are using the maximum likelihood method instead of the restricted maximum likelihood method for estimation.

The marginal specification of this model can be fitted by the **mcglm** package. The first step is to specify the linear predictor.

```
R> form_ex1 <- Reaction ~ Days
```

The second step corresponds to building the components of the matrix linear predictor. Note that in that case we have to take into account the repeated measures effect (`subject`) as well as the effect of the covariate `Days` and perhaps the interaction term (correlation between the random intercept and slope) in a mixed models fashion, see Equation 3. The components of the matrix linear predictor are obtained by:

```
R> library("mcglm")
R> Z_ex1 <- mc_mixed(~ 0 + Subject / Days, data = sleepstudy)
```

We also include an intercept matrix corresponding to the residual variance.

```
R> Z0_ex1 <- mc_id(sleepstudy)
```

Finally, we fit the model.

```
R> fit_ex1 <- mcglm(linear_pred = c(form_ex1),
+    matrix_pred = list(resp = c(Z0_ex1, Z_ex1)),
+    control_algorithm = list(correct = FALSE), data = sleepstudy)
```

In order to have estimates comparable with the ones obtained by the **lme4** package we are using the option `correct = FALSE` that corresponds in the Gaussian case to the maximum likelihood method.

```
R> summary(fit_ex1)

Call: Reaction ~ Days

Link function: identity
Variance function: constant
Covariance function: identity
Regression:
            Estimates Std.error Z value
(Intercept) 251.40510  6.632277 37.9063
Days         10.46729  1.502237  6.9678

Dispersion:
  Estimates Std.error   Z value
1 654.94103  70.62388 9.2736486
2 565.51537 264.67950 2.1366044
3  32.68220  13.55974 2.4102375
4  11.05543  42.94762 0.2574166

Algorithm: chaser
Correction: FALSE
Number iterations: 3
```

The `summary()` output presents the formula of the linear predictor, the link, the variance and the covariance functions specified for fitting the model. The parameter estimates are presented in two separate sets, `Regression` and `Dispersion`, with their associated standard errors and Z statistics. Finally, the fitting algorithm, if the bias correction term was used or not and the number of iterations required to reach convergence are shown.

We can compare the results obtained by the **mcglm** and **lme4** packages using the maximized value of the log-likelihood function.

```
R> logLik(fit1.lme4)

'log Lik.' -875.9697 (df=6)

R> plogLik(fit_ex1)
```

```
Pseudo log Lik. -875.97 (df=6)
```

The two packages provide virtually the same estimates for regression and dispersion parameters. However, the standard errors associated with the dispersion parameters (variance components) from the **mcglm** package are in general larger than the ones obtained by the **lme4** package. This is due to the robust specification of McGLMs and the use of a not fully efficient estimating function for estimation of the dispersion parameters. The interested reader can obtain these quantities using the `coef` function.

We now turn to the selection of the matrix linear predictor components using the SIC as presented in Section 4. In this case, we have four components to compose the matrix linear predictor. The initial or basic model can be the one assuming independent observations:

```
R> fit0_ex1 <- mcglm(linear_pred = c(form_ex1),
+    matrix_pred = list(resp = c(Z0_ex1)),
+    control_algorithm = list(correct = FALSE), data = sleepstudy)
```

The components of the matrix linear predictor associated with the two main effects are obtained by:

```
R> Z.sic <- list("Z1" = Z_ex1[[1]], "Z2" = Z_ex1[[2]])
```

Computing the SIC for these components is performed by:

```
R> mc_sic_covariance(fit0_ex1, scope = Z.sic, idx = c(1, 2),
+    data = sleepstudy, response = 1)

        SIC df df_total       Tu    Chisq
1 -264.7654  1        2 268.7654 3.841459
2 -285.8309  1        2 289.8309 3.841459
```

The SIC values indicate that both components should be included in the model (SIC < 0). Thus,

```
R> fit1_ex1 <- mcglm(linear_pred = c(form_ex1),
+    matrix_pred = list(resp = c(Z0_ex1, Z.sic)),
+    control_algorithm = list(correct = FALSE), data = sleepstudy)
```

Now, we compute the SIC for the interaction term.

```
R> Z3_ex1 <- list("Z3" = Z_ex1[[3]])
R> mc_sic_covariance(fit1_ex1, scope = Z3_ex1, idx = c(1), response = 1)

       SIC df df_total         Tu    Chisq
1 7.938022  1        4 0.06197811 3.841459
```

The SIC shows that the interaction effect is not required, which agrees with the first fitted model (`fit_ex1`) where the interaction effect is not significant.

6.2. Longitudinal data analysis

The second example regards a fairly common longitudinal data analysis. We use the `dietox` data set available in package **geepack**. The analysis goal is to investigate the effect of dietary copper (`Cu`) and vitamin E (`Evit`) in diets containing 6% rapeseed oil on the performance status of growing pigs. In this application, for composing the linear predictor we have three covariates `Evit`, `Cu` and `Time`. We consider interaction terms up to the second order between the three main effects. In order to select the components of the linear predictor we are going to use the SIC. The longitudinal or repeated measures structure is introduced by the observations taken for the same animal. Thus, the matrix linear predictor is composed of an identity matrix combined with a compound symmetry structure, see Section 2.

```
R> data("dietox", package = "geepack")
R> Z0_ex2 <- mc_id(dietox)
R> Z1_ex2 <- mc_mixed(~ 0 + Pig, data = dietox)
```

To start the selection of the components of the linear predictor, we fit a simple intercept model, but also consider the repeated measures structure.

```
R> fit0_ex2 <- mcglm(linear_pred = c(Weight ~ 1),
+    matrix_pred = list(resp = c(Z0_ex2, Z1_ex2)), data = dietox)
```

The candidate terms to enter into the linear predictor are:

```
R> scope <- c("poly(Time, 2)", "Evit", "Cu", "Evit * poly(Time, 2)",
+    "Cu * poly(Time, 2)", "Evit * Cu")
```

Now, we compute the SIC for all candidate terms. In that case, we decide to use the penalty $\log(N)$ to have an analog to the Bayesian information criterion.

```
R> N <- dim(dietox)[1]
R> mc_sic(fit0_ex2, scope = scope, data = dietox, response = 1,
+    penalty = log(N))

         SIC           Covariates df df_total         Tu     Chisq
1 -687.30632        poly(Time, 2)  2        3 707.318622  5.991465
2   17.17333                 Evit  2        3   2.838969  5.991465
3   17.96460                   Cu  2        3   2.047695  5.991465
4 -650.74637 Evit * poly(Time, 2)  8        9 710.783268 15.507313
5 -649.52533   Cu * poly(Time, 2)  8        9 709.562228 15.507313
6   54.68074            Evit * Cu  8        9   5.356152 15.507313
```

Clearly, the first term to be included in the model is the one associated with the `Time` trend.

```
R> fit1_ex2 <- mcglm(linear_pred = c(Weight ~ poly(Time, 2)),
+    matrix_pred = list(resp = c(Z0_ex2, Z1_ex2)), data = dietox)
```

Updating the candidate terms and computing the SIC are performed using:

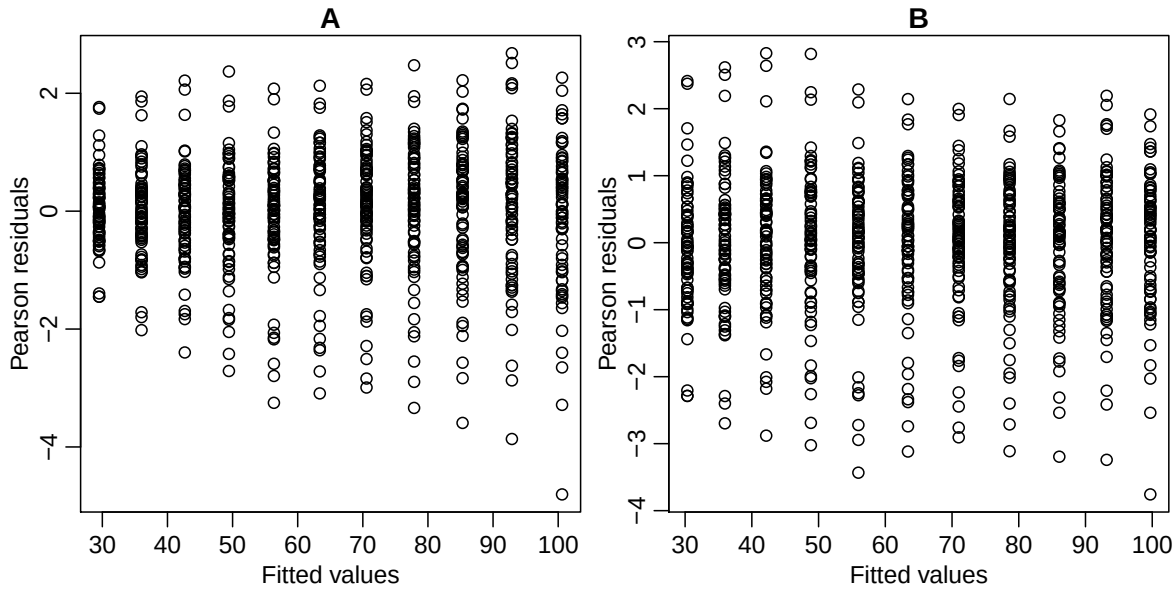

Figure 1: Pearson residuals against fitted values for models using constant (A) and Tweedie (B) variance functions.

```
R> scope <- c("Evit", "Cu", "Evit * poly(Time, 2)", "Cu * poly(Time, 2)",
+    "Evit * Cu")
R> mc_sic(fit1_ex2, scope = scope, data = dietox, response = 1,
+    penalty = log(N))
```

```
       SIC           Covariates df df_total        Tu     Chisq
1 30.21801                 Evit  2        5  3.135819  5.991465
2 31.12764                   Cu  2        5  2.226191  5.991465
3 39.70085 Evit * poly(Time, 2)  6        9 20.336051 12.591587
4 53.96472   Cu * poly(Time, 2)  6        9  6.072174 12.591587
5 67.59189            Evit * Cu  8       11  5.786537 15.507313
```

The SIC values show that the other terms are not required. Thus, we reached the final model. Figure 1(A) hints at the presence of variance heterogeneity. It indicates that a more general variance function, such as the Tweedie variance function can provide a better fit.

```
R> fit2_ex2 <- mcglm(linear_pred = c(Weight ~ poly(Time, 2)),
+    link = "log", variance = "tweedie", power_fixed = FALSE,
+    control_algorithm = list(tuning = 0.85),
+    matrix_pred = list(resp = c(Z0_ex2, Z1_ex2)), data = dietox)
```

The results in Figure 1(B) show that the `tweedie` variance function eases the variance heterogeneity detected in Figure 1(A). We can also use the measures of goodness-of-fit to decide about the best fit.

```
R> rbind(gof(fit1_ex2), gof(fit2_ex2))
```

```
   plogLik Df    pAIC    pKLIC     pBIC
1 -2134.58  5 4279.16 4387.053 4302.514
2 -2041.61  6 4095.22 4214.939 4123.245
```

The measures of goodness-of-fit employed show that the model using the `"tweedie"` variance function provides the best fit for the `dietox` data set. In general the estimation of the power parameter inflates the standard errors associated with the dispersion parameter estimates. Thus, we suggest to use a conditional hypothesis test where the value of the power parameter is fixed at its estimate instead of the orthodox marginal test. For the `dietox` data the results of the marginal test for the dispersion parameter estimates associated with the repeated measures structure indicate that this effect is not significant. On the other hand, the conditional test shows that the repeated measures effect is significant as expected. The marginal test is obtained using:

```
R> summary(fit2_ex2, verbose = TRUE, print = "Dispersion")
```

```
Dispersion:
  Estimates  Std.error  Z value
1 0.1143034 0.06409703 1.783286
2 0.6759078 0.39697332 1.702653

Algorithm: chaser
Correction: TRUE
Number iterations: 8
```

The conditional test is obtained with:

```
R> mc_conditional_test(fit2_ex2,
+    parameters = c("power11", "tau11", "tau12"), test = 2:3, fixed = 1)
```

```
  Estimates  Std.error  Z.value
1 0.1143034 0.02996828 3.814144
2 0.6759078 0.11439159 5.908720
```

6.3. Spatial areal data analysis

This example regards the malaria prevalence in children recorded at 65 villages in Gambia. We follow Masarotto and Varin (2017) who consider the aggregated data at village level. The data set is available in the **gcmr** package. The response variable is the number of sampled children with malaria in each village. Possible covariates include the frequency of sampled children who regularly sleep under a bed-net in each village (`netuse`). A measure of vegetation greenness in the immediate vicinity of the village (`green`) and an indicator variable denoting the presence or absence of a health center in the village (`phd`). Following Masarotto and Varin (2017) we specify the following linear predictor using:

```
R> library("spdep")
R> data("malaria", package = "gcmr")
R> malaria$Y <- malaria$cases / malaria$size
R> form_ex3 <- Y ~ netuse + I(green / 100) + phc
```

In this example the response variable is binomial, thus we first compute the proportion of children with malaria in each village. To model the spatial effect, we apply the conditional auto-regressive model, see Equation 4.

```
R> list_neigh <- tri2nb(coords = malaria[, 1:2])
R> Z_ex3 <- mc_car(list_neigh = list_neigh)
```

The following call of the `mcglm` function fits the spatial model. This model is then summarized:

```
R> fit1_ex3 <- mcglm(linear_pred = c(form_ex3), matrix_pred = list(Z_ex3),
+    link = "logit", variance = "binomialP", covariance = "inverse",
+    control_algorithm = list(verbose = FALSE, max_iter = 100,
+      tuning = 0.1, method = "rc"), Ntrial = list(malaria$size),
+    data = malaria)
R> summary(fit1_ex3)

Call: Y ~ netuse + I(green / 100) + phc

Link function: logit
Variance function: binomialP
Covariance function: inverse
Regression:
                Estimates Std.error   Z value
(Intercept)    -1.2974098 0.9301458 -1.394846
netuse         -0.9862835 0.2909367 -3.390028
I(green / 100)  3.2369222 1.7879445  1.810415
phc            -0.3449150 0.1827457 -1.887404

Dispersion:
   Estimates  Std.error  Z value
1 0.05421655 0.01018959 5.320778
2 0.04141227 0.01448750 2.858483

Algorithm: rc
Correction: TRUE
Number iterations: 23
```

Note that in this example we are using the *reciprocal likelihood* algorithm (`method = "rc"`) and the tuning constant (`tuning = 0.1`). Since, we have a binomial response variable, we specify the `logit` and `binomialP` as the link and variance functions, respectively. Finally, to obtain the conditional auto-regressive model we are using the `inverse` covariance link function. Concerning the covariate effects only the covariate `netuse` shows a significant effect. The components of the matrix linear predictor show a significant spatial effect. The function `mc_compute_rho` computes the spatial auto-correlation estimates based on a fitted model.

```
R> mc_compute_rho(fit1_ex3)

        rho       std  Conf.Min Conf.Max
1 0.7638308 0.1883931 0.3945871 1.133074
```

An alternative model is obtained by using the inverse of the Euclidean distance combined with the exponential-matrix covariance link function.

```
R> Z0_ex3 <- mc_id(malaria)
R> Z1_ex3 <- as.matrix(1 / dist(malaria[, 1:2] / 1000, upper = TRUE,
+    diag = TRUE))
R> Z1_ex3 <- list("Z1" = Z1_ex3)
R> fit2_ex3 <- mcglm(linear_pred = c(form_ex3),
+    matrix_pred = list(c(Z0_ex3, Z1_ex3)), link = "logit",
+    variance = "binomialP", covariance = "expm",
+    Ntrial = list(malaria$size), data = malaria)
```

We compare the fitted models using the measures of goodness-of fit presented in Section 4.

```
R> rbind(gof(fit1_ex3), gof(fit2_ex3))

  plogLik Df   pAIC     pKLIC      pBIC
1   23.64  6 -35.28 -31.03616 -22.23368
2   23.84  6 -35.68 -34.19290 -22.63368
```

In that case, the model using the inverse of the Euclidean distance and the `expm` covariance link function provides a fit slightly better than the conditional auto-regressive model.

6.4. Mixed response variables

The experiment analyzed in this section was carried out in a vegetation house with soybeans. The experiment has two plants by plot with three levels of the factor corresponding to amount of water in the soil (`water`) and five levels of potassium fertilization (`pot`). The plots were arranged in five blocks (`block`). Three response variables are of the interest, namely, grain yield, number of seeds and number of viable peas per plant. This experiment is particularly interesting because we have three response variables of mixed types, i.e., grain yield is a continuous outcome while number of seeds and number of viable peas per plant are examples of count and binomial response variables, respectively. The data set is available in the **mcglm** package. We started this data analysis by fitting three independent models.

```
R> data("soya", package = "mcglm")
R> form.grain <- grain ~ block + water * pot
R> form.seed <- seeds ~ block + water * pot
R> soya$viablepeasP <- soya$viablepeas / soya$totalpeas
R> form.peas <- viablepeasP ~ block + water * pot
R> Z0_ex4 <- mc_id(soya)
R> fit.grain <- mcglm(linear_pred = c(form.grain),
+    matrix_pred = list(Z0_ex4), data = soya)
R> fit.seed <- mcglm(linear_pred = c(form.seed), matrix_pred = list(Z0_ex4),
+    link = c("log"), variance = c("tweedie"), power_fixed = TRUE,
+    data = soya)
R> fit.peas <- mcglm(linear_pred = c(form.peas), matrix_pred = list(Z0_ex4),
+    link = "logit", variance = "binomialP", Ntrial = list(soya$totalpeas),
+    data = soya)
```

Now, we can easily fit the multivariate model and compare the results.

```
R> fit.joint <- mcglm(linear_pred = c(form.grain, form.seed, form.peas),
+    matrix_pred = list(Z0_ex4, Z0_ex4, Z0_ex4), link = c("identity",
+      "log", "logit"), variance = c("constant", "tweedie", "binomialP"),
+    Ntrial = list(NULL, NULL, soya$totalpeas), data = soya)
R> rbind(gof(list(fit.grain, fit.seed, fit.peas)), gof(fit.joint))

  plogLik Df   pAIC    pKLIC      pBIC
1 -339.54 60 799.08 833.0493 1004.0460
2 -319.73 63 765.46 847.7463  980.6743
```

The function `gof` allows to compute the measures of goodness-of-fit combining one response variable models, assuming that the correlation between the response variables is zero. In that case, the multivariate model provides a better fit than the one response variable models, at least judging by the `plogLik`, `pAIC` and `pBIC` values. The `pKLIC` tends to strongly penalize models with extra correlation parameters. Thus, we do not recommend to use this measure to compare covariance structures with different number of parameters. On the other hand, the `pKLIC` is a good choice for comparing covariance structures with a similar number of parameters, as shown in the example in Section 6.3.

The summary output of the model is similar to the one for the one response variable model already presented. The extra feature shown for the multivariate model is the correlation matrix between response variables.

```
R> summary(fit.joint, verbose = TRUE, print = "Correlation")

Correlation matrix:
  Parameters  Estimates Std.error   Z value
1      rho12 0.63767011 0.1417184 4.4995591
2      rho13 0.07034904 0.1156794 0.6081377
3      rho23 0.08882875 0.1152445 0.7707851

Algorithm: chaser
Correction: TRUE
Number iterations: 10
```

Furthermore, in this data analysis where a large number of regression coefficients are estimated including interaction effects, the `anova` function is a convenient way to check the effect significance.

```
R> anova(fit.joint)

Wald test for fixed effects
Call: grain ~ block + water * pot

      Covariate Chi.Square Df p.value
1       blockII    23.9952  4  0.0001
```

```
2       water50     4.0222  2  0.1338
3         pot30   107.3124  4  0.0000
4 water50:pot30    51.0550  8  0.0000

Call: seeds ~ block + water * pot

      Covariate Chi.Square Df p.value
1       blockII    19.5163  4  0.0006
2       water50     6.6050  2  0.0368
3         pot30    32.1902  4  0.0000
4 water50:pot30    21.3563  8  0.0063

Call: viablepeasP ~ block + water * pot

      Covariate Chi.Square Df p.value
1       blockII     7.4417  4  0.1143
2       water50     8.8128  2  0.0122
3         pot30   119.2957  4  0.0000
4 water50:pot30    33.6562  8  0.0000
```

6.5. Bivariate repeated measures models for count data

In this section we consider the bivariate longitudinal count data analyzed in Bonat *et al.* (2017) concerning data of animals hunted in the village of Basile Fang, Bioko Norte Province, Bioko Island, Equatorial Guinea. Monthly numbers of blue duikers (`BD`) and other small animals (`OT`) shot or snared were collected for a random sample of 52 commercial hunters from August 2010 to September 2013. Covariates available to compose the linear predictors include: `ALT`, a factor with five levels indicating the altitude where the animal was caught; `SEX`, a factor with two levels (Female and Male); `METHOD`, a factor with two levels (Firearm and Snare) and `MONTH`. The number of hunting days per month is also available in variable `OFFSET` and should be used as an offset in the models. Here, we present a simplified analysis considering only the repeated measures structure introduced by the observations taken for the same hunter and month (variable `HUNTER.MONTH`). For a more detailed description of the data and complete analysis we refer to Grande-Vega, Farfán, Ondo, and Fa (2016) and Bonat *et al.* (2017).

Following Bonat *et al.* (2017) the linear predictors for `BD` and `OT` are specified by:

```
R> data("Hunting", package = "mcglm")
R> form.OT <- OT ~ METHOD * ALT + SEX + ALT * poly(MONTH, 4)
R> form.BD <- BD ~ METHOD * ALT + SEX + ALT * poly(MONTH, 3)
```

The matrix linear predictor is specified by a linear combination between an identity and a compound symmetry matrix for the `HUNTER.MONTH` effect.

```
R> Z0_ex5 <- mc_id(Hunting)
R> Z1_ex5 <- mc_mixed(~ 0 + HUNTER.MONTH, data = Hunting)
```

Finally, we fit the bivariate model using the `poisson_tweedie` variance function.

```
R> fit1_ex5 <- mcglm(linear_pred = c(form.BD, form.OT),
+    matrix_pred = list(c(Z0_ex5, Z1_ex5), c(Z0_ex5, Z1_ex5)),
+    link = c("log", "log"), variance = c("poisson_tweedie",
+      "poisson_tweedie"), power_fixed = c(FALSE, FALSE),
+    offset = list(log(Hunting$OFFSET), log(Hunting$OFFSET)),
+    control_algorithm = list(max_iter = 200, verbose = FALSE),
+    data = Hunting)
```

We assess the significance of the regression coefficients using the `anova` function.

```
R> anova(fit1_ex5)

Wald test for fixed effects
Call: BD ~ METHOD * ALT + SEX + ALT * poly(MONTH, 3)

             Covariate Chi.Square Df p.value
1         METHODTrampa     3.8902  1  0.0486
2                 ALT2   162.0670  4  0.0000
3              SEXMale   401.4957  1  0.0000
4      poly(MONTH, 3)1    32.3174  3  0.0000
5    METHODTrampa:ALT2    63.2427  4  0.0000
6 ALT2:poly(MONTH, 3)1    71.2614 12  0.0000

Call: OT ~ METHOD * ALT + SEX + ALT * poly(MONTH, 4)

             Covariate Chi.Square Df p.value
1         METHODTrampa     1.1885  1  0.2756
2                 ALT2   158.9866  4  0.0000
3              SEXMale    67.5059  1  0.0000
4      poly(MONTH, 4)1     7.4804  4  0.1126
5    METHODTrampa:ALT2    32.6643  4  0.0000
6 ALT2:poly(MONTH, 4)1   100.8068 16  0.0000
```

The power, dispersion and correlation parameter estimates are provided by the summary.

```
R> summary(fit1_ex5, print = c("power", "Dispersion", "Correlation"))

Power:
  Estimates Std.error  Z value
1  1.460708 0.1653851 8.832164

Dispersion:
  Estimates Std.error  Z value
1 0.1116979 0.0527109 2.119067
2 0.4577956 0.1386405 3.302034
```

```
Power:
  Estimates Std.error  Z value
1  1.799323 0.2482945 7.246727

Dispersion:
  Estimates  Std.error  Z value
1 0.2052374 0.11454898 1.791700
2 0.5166482 0.09456719 5.463292

Correlation matrix:
  Parameters   Estimates  Std.error   Z value
1      rho12 -0.06845904 0.02904716 -2.356824

Algorithm: chaser
Correction: TRUE
Number iterations: 113
```

The results show that the Pólya-Aepli ($p = 1.5$) and negative binomial ($p = 2$) distributions can be suggested to the response variables `BD` and `OT`, respectively. The dispersion structures show the significance of the repeated measures structure for both response variables. The correlation between response variables is weak, but still significantly different from 0.

6.6. Multivariate Tweedie models for spatial areal data

The last example analyzes the spatial distribution of soil chemistry properties measured on a regular grid with 10×25 points spaced by 5 meters. Three continuous response variables are of interest, namely, calcium (`CA`), magnesium (`MG`) and potassium (`PT`) contents. Covariates include the spatial coordinates (`COORD.X` and `COORD.Y`), soil pH at water (`PHWATER`) and the portions of sand (`SAND`), silt (`SILT`) and clay (`CLAY`). The original data set is available in the **geoR** package (Ribeiro Jr and Diggle 2016). The simplified version used here is available in the **mcglm** package.

We model the covariance structure within response variables by combining the `tweedie` variance function and the conditional auto-regressive model (see Equation 4). The linear and matrix linear predictors for each response variable are specified as follows:

```
R> data("soil", package = "mcglm")
R> form.ca <- CA ~ COORD.X * COORD.Y + SAND + SILT + CLAY + PHWATER
R> form.mg <- MG ~ COORD.X * COORD.Y + SAND + SILT + CLAY + PHWATER
R> form.k <- K ~ COORD.X * COORD.Y + SAND + SILT + CLAY + PHWATER
R> list_neigh <- tri2nb(coords = soil[, 1:2])
R> Z_ex6 <- mc_car(list_neigh = list_neigh)
```

The estimation of multivariate models is a complex computational task. The choice of good initial values in general helps to reach convergence. A simple strategy to get good initial values consists of fitting separate models for each response variable and then use their estimates as initial values for the multivariate model. In this data analysis, we follow this strategy as shown by the code below.

```
R> fit.ca <- mcglm(linear_pred = c(form.ca), matrix_pred = list(Z_ex6),
+    link = "log", variance = "tweedie", covariance = "inverse",
+    power_fixed = FALSE, data = soil,
+    control_algorithm = list(max_iter = 500, tuning = 0.1))
R> fit.mg <- mcglm(linear_pred = c(form.mg), matrix_pred = list(Z_ex6),
+    link = "log", variance = "tweedie", covariance = "inverse",
+    power_fixed = FALSE, data = soil,
+    control_algorithm = list(max_iter = 500, tuning = 0.05))
R> fit.k <- mcglm(linear_pred = c(form.k), matrix_pred = list(Z_ex6),
+    link = "log", variance = "tweedie", covariance = "inverse",
+    power_fixed = FALSE, data = soil,
+    control_algorithm = list(max_iter = 500, tuning = 0.05))
```

The initial values for the `mcglm` function are supplied through a named list, with elements for the `regression`, `power`, `tau` and `correlation` parameters.

```
R> ini <- list()
R> ini$regression <- list("CA" = coef(fit.ca, type = "beta")$Estimates,
+    "MG" = coef(fit.mg, type = "beta")$Estimates,
+    "K" = coef(fit.k, type = "beta")$Estimates)
R> ini$power <- list("CA" = coef(fit.ca, type = "power")$Estimates,
+    "MG" = coef(fit.mg, type = "power")$Estimates,
+    "K" = coef(fit.k, type = "power")$Estimates)
R> ini$tau <- list("CA" = coef(fit.ca, type = "tau")$Estimates,
+    "MG" = coef(fit.mg, type = "tau")$Estimates,
+    "K" = coef(fit.k, type = "tau")$Estimates)
R> ini$rho <- c(0, 0, 0)
```

In the following the multivariate Tweedie model is fitted.

```
R> fit_ex6 <- mcglm(linear_pred = c(form.ca, form.mg, form.k),
+    matrix_pred = list(Z_ex6, Z_ex6, Z_ex6), link = c("log", "log", "log"),
+    variance = c("tweedie", "tweedie", "tweedie"),
+    covariance = c("inverse", "inverse", "inverse"),
+    power_fixed = c(FALSE, FALSE, FALSE), control_initial = ini,
+    data = soil, control_algorithm = list(max_iter = 100))
```

The users can summarize the model using the `summary` function or extract parameter estimates using the `coef` function as well as use all methods available in the **mcglm** package for analyzing the model results. In particular, for this application the functions `mc_compute_rho` and `mc_conditional_test` are tailored tools to compute the spatial auto-correlation and perform conditional hypotheses tests for the dispersion parameters associated with the spatial structure.

7. Discussion

This article described the R implementation of multivariate covariance generalized linear models in the **mcglm** package. The discussed examples illustrate the capability of the package

to deal with various types of data and dependence structures. The models are fitted by an estimating function approach combining quasi-score and Pearson estimating functions for estimation of the regression and dispersion parameters, respectively. The main technical advantage of this approach is the simplicity of the fitting method, which amounts to finding the root for a set of non-linear equations. In this direction, we presented the *modified chaser* and *reciprocal likelihood* methods. A careful choice of initial values is required for both methods. The fitting process while relying on a simple Newton scoring algorithm also involves computational demanding tasks such as the Cholesky decomposition and multiplication of large matrices. Thus, in the case of large data sets, consisting of several thousands of observations, and models involving many components in the matrix linear predictor, the computational cost may prevent routine use of the **mcglm** package.

The package was designed to take full advantage of the modular specification of the models using a `glm` style interface. The output of the functions is provided in a way that is easy to interpret for people familiar with standard `lm()`, `glm()` or `gam()` output. Furthermore, a set of methods and auxiliary functions are available for analyzing the model results. The covariance structure within response variable is modeled by combining a matrix linear predictor and a covariance link function. A set of functions for constructing the components of the matrix linear predictor is made available. Furthermore, the users can easily extend the package with new components.

The estimation of the power parameter is, in general, a difficult task. The estimation of the power parameter is particularly challenging when the distribution of the response variable is symmetric for continuous data using the Tweedie variance function. The power parameter is an index that distinguishes between some well known distributions such as the Gamma and inverse Gaussian. All these distributions can provide a good fit for symmetric data. Thus, we do not have enough information in the data to distinguish between these distributions. Therefore, we suggest opting for the simplest possibility i.e., the Gaussian model. Similarly, for count data using the Poisson-Tweedie dispersion function the estimation of the dispersion and power parameters are difficult for equidispersed count data. In such a case the dispersion parameter in the Poisson-Tweedie dispersion function should be zero, which in turn is a value close to the border of the parameter space. Therefore, we suggest again opting for the simplest possibility i.e., the Poisson model obtained by using the Tweedie variance function with power parameter fixed at 1.

Possible topics for further investigation and extensions include facilities for censored data in survival analysis and other special types of data and improvements regarding the residual analysis. In particular, the investigation of how the McGLMs framework can be used to deal with multivariate spatio-temporal data is an interesting topic. Further possible extensions could cover the incorporation of penalized splines in the mean and covariance structures, and the use of regularization for high dimensional data, which would have important applications in genetics. Also of interest would be the construction of new estimating functions to deal with data that are not missing at random and to make prediction using the best linear unbiased predictor (BLUP). Further points worth investigation would be a thorough comparison of the McGLMs framework with more established statistical modeling frameworks such as GEE (generalized estimating equations) models, Gaussian copula regression models, hierarchical generalized linear models and other.

References

Anderson TW (1973). "Asymptotically Efficient Estimation of Covariance Matrices with Linear Structure." *The Annals of Statistics*, **1**(1), 135–141. doi:10.1214/aos/1193342389.

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.

Bates D, Maechler M (2017). ***Matrix**: Sparse and Dense Matrix Classes and Methods*. R package version 1.2-12, URL https://CRAN.R-project.org/package=Matrix.

Bivand R (2017). ***spdep**: Spatial Dependence: Weighting Schemes, Statistics and Models*. R package version 0.7-4, URL https://CRAN.R-project.org/package=spdep.

Bivand R, Piras G (2015). "Comparing Implementations of Estimation Methods for Spatial Econometrics." *Journal of Statistical Software*, **63**(18), 1–36. doi:10.18637/jss.v063.i18.

Bonat WH (2017). "Modelling Mixed Outcomes in Additive Genetic Models." *The International Journal of Biostatistics*, **13**(2), 1–17. doi:10.1515/ijb-2017-0001.

Bonat WH (2018). ***mcglm**: Multivariate Covariance Generalized Linear Models*. R package version 0.4.0, URL https://CRAN.R-project.org/package=mcglm.

Bonat WH, Jørgensen B (2016). "Multivariate Covariance Generalized Linear Models." *Journal of the Royal Statistical Society C*, **65**(5), 649–675. doi:10.1111/rssc.12145.

Bonat WH, Olivero J, Grande-Vega M, Fárfan MA, Fa JE (2017). "Modelling the Covariance Structure in Marginal Multivariate Count Models." *Journal of Agricultural, Biological and Environmental Statistics*, **22**(4), 446–464. doi:10.1007/s13253-017-0284-7.

Carey VJ (2015). ***gee**: Generalized Estimation Equation Solver*. R package version 4.13-19, URL https://CRAN.R-project.org/package=gee.

Carey VJ, Wang Y (2011). "Working Covariance Model Selection for Generalized Estimating Equations." *Statistics in Medicine*, **30**(26), 3117–3124. doi:10.1002/sim.4300.

Carpenter B, Gelman A, Hoffman M, Lee D, Goodrich B, Betancourt M, Brubaker M, Guo J, Li P, Riddell A (2017). "**Stan**: A Probabilistic Programming Language." *Journal of Statistical Software*, **76**(1), 1–32. doi:10.18637/jss.v076.i01.

Chiu TYM, Leonard T, Tsui K (1996). "The Matrix-Logarithmic Covariance Model." *Journal of the American Statistical Association*, **91**(433), 198–210. doi:10.1080/01621459.1996.10476677.

Crouchley R (2012). ***sabreR**: Multivariate Generalized Linear Mixed Models*. R package version 2.0.

Demidenko E (2013). *Mixed Models: Theory and Applications with R*. John Wiley & Sons. doi:10.1002/0471728438.

Fieuws S, Verbeke G, Molenberghs G (2007). "Random-Effects Models for Multivariate Repeated Measures." *Statistical Methods in Medical Research*, **16**(5), 387–397. doi:10.1177/0962280206075305.

Grande-Vega M, Farfán MÁ, Ondo A, Fa JE (2016). "Decline in Hunter Offtake of Blue Duikers in Bioko Island, Equatorial Guinea." *African Journal of Ecology*, **54**(1), 49–58. doi:10.1111/aje.12260.

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.

Hadfield JD, Nakagawa S (2010). "General Quantitative Genetic Methods for Comparative Biology: Phylogenies, Taxonomies and Multi-Trait Models for Continuous and Categorical Characters." *Journal of Evolutionary Biology*, **23**(3), 494–508. doi:10.1111/j.1420-9101.2009.01915.x.

Højsgaard S, Halekoh U, Yan J (2006). "The R Package **geepack** for Generalized Estimating Equations." *Journal of Statistical Software*, **15**(2), 1–11. doi:10.18637/jss.v015.i02.

Jørgensen B (1987). "Exponential Dispersion Models." *Journal of the Royal Statistical Society B*, **49**(2), 127–162.

Jørgensen B (1997). *The Theory of Dispersion Models*. Chapman & Hall, London.

Jørgensen B (2013). "Construction of Multivariate Dispersion Models." *Brazilian Journal of Probability and Statistics*, **27**(3), 285–309. doi:10.1214/11-bjps171.

Jørgensen B, Knudsen SJ (2004). "Parameter Orthogonality and Bias Adjustment for Estimating Functions." *Scandinavian Journal of Statistics*, **31**(1), 93–114. doi:10.1111/j.1467-9469.2004.00375.x.

Jørgensen B, Kokonendji CC (2016). "Discrete Dispersion Models and Their Tweedie Asymptotics." *AStA Advances in Statistical Analysis*, **100**(1), 43–78. doi:10.1007/s10182-015-0250-z.

Jørgensen B, Lauritzen SL (2000). "Multivariate Dispersion Models." *Journal of Multivariate Analysis*, **74**(2), 267–281. doi:10.1006/jmva.1999.1885.

Krupskii P, Joe H (2013). "Factor Copula Models for Multivariate Data." *Journal of Multivariate Analysis*, **120**, 85–101. doi:10.1016/j.jmva.2013.05.001.

Lee Y, Nelder JA (1996). "Hierarchical Generalized Linear Models." *Journal of the Royal Statistical Society B*, **58**(4), 619–678.

Liang KY, Zeger SL (1986). "Longitudinal Data Analysis Using Generalized Linear Models." *Biometrika*, **73**(1), 13–22. doi:10.1093/biomet/73.1.13.

Liang KY, Zeger SL, Qaqish B (1992). "Multivariate Regression Analyses for Categorical Data." *Journal of the Royal Statistical Society B*, **54**(1), 3–40.

Lindgren F, Rue H (2015). "Bayesian Spatial Modelling with R-**INLA**." *Journal of Statistical Software*, **63**(19), 1–25. doi:10.18637/jss.v063.i19.

Lunn DJ, Thomas A, Best N, Spiegelhalter D (2000). "**WinBUGS** – A Bayesian Modelling Framework: Concepts, Structure, and Extensibility." *Statistics and Computing*, **10**(4), 325–337. doi:10.1023/a:1008929526011.

Martin AD, Quinn KM, Park JH (2011). "**MCMCpack**: Markov Chain Monte Carlo in R." *Journal of Statistical Software*, **42**(9), 1–22. doi:10.18637/jss.v042.i09.

Martinez-Beneito MA (2013). "A General Modelling Framework for Multivariate Disease Mapping." *Biometrika*, **100**(3), 539–553. doi:10.1093/biomet/ast023.

Masarotto G, Varin C (2012). "Gaussian Copula Marginal Regression." *Electronic Journal of Statistics*, **6**, 1517–1549. doi:10.1214/12-ejs721.

Masarotto G, Varin C (2017). "Gaussian Copula Regression in R." *Journal of Statistical Software*, **77**(8), 1–26. doi:10.18637/jss.v077.i08.

McDaniel LS, Henderson NC, Rathouz PJ (2013). "Fast Pure R Implementation of GEE: Application of the **Matrix** Package." *The R Journal*, **5**(1), 181–187.

Nuamah IF, Qu Y, Amini SB (1996). "A SAS Macro for Stepwise Correlated Binary Regression." *Computer Methods and Programs in Biomedicine*, **49**(3), 199–210. doi:10.1016/0169-2607(96)01718-x.

Pinheiro J, Bates D, DebRoy S, Sarkar D, R Core Team (2017). ***nlme**: Linear and Nonlinear Mixed Effects Models*. R package version 3.1-131, URL https://CRAN.R-project.org/package=nlme.

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*, pp. 1–10. Technische Universität Wien, Vienna, Austria.

Pourahmadi M (2000). "Maximum Likelihood Estimation of Generalised Linear Models for Multivariate Normal Covariance Matrix." *Biometrika*, **87**(2), 425–435. doi:10.1093/biomet/87.2.425.

Rabe-Hesketh S, Skrondal A, Gjessing HK (2008). "Biometrical Modeling of Twin and Family Data Using Standard Mixed Model Software." *Biometrics*, **64**(1), 280–288. doi:10.1111/j.1541-0420.2007.00803.x.

R Core Team (2017). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria. URL https://www.R-project.org/.

Ribeiro Jr PJ, Diggle PJ (2016). ***geoR**: Analysis of Geostatistical Data*. R package version 1.7-5.2, URL https://CRAN.R-project.org/package=geoR.

Rönnegård L, Shen X, Alam M (2010). "Fitting Hierarchical Generalized Linear Models." *The R Journal*, **2**(2), 20–28. URL https://journal.R-project.org/archive/2010/RJ-2010-009.

Rue H, Held L (2005). *Gaussian Markov Random Fields: Theory and Applications*. Chapman & Hall, London. doi:10.1201/9780203492024.

SAS Institute (2011). *SAS/STAT 9.3 User's Guide: The* ***GLIMMIX*** *Procedure*. Carey, NC.

Stoklosa J, Gibb H, Warton DI (2014). "Fast Forward Selection for Generalized Estimating Equations with a Large Number of Predictor Variables." *Biometrics*, **70**(1), 110–120. doi:10.1111/biom.12118.

Verbeke G, Fieuws S, Molenberghs G, Davidian M (2014). "The Analysis of Multivariate Longitudinal Data: A Review." *Statistical Methods in Medical Research*, **23**(1), 42–59. doi:10.1177/0962280212445834.

Wolak ME (2012). "**nadiv**: An R Package to Create Relatedness Matrices for Estimating Non-Additive Genetic Variances in Animal Models." *Methods in Ecology and Evolution*, **3**(5), 792–796. doi:10.1111/j.2041-210x.2012.00213.x.

Affiliation:

Wagner Hugo Bonat
Laboratory of Statistics and Geoinformation
Department of Statistics
Paraná Federal University
Centro Politécnico
Curitiba 81531980, CP 19081, Paraná, Brazil.
E-mail: wbonat@ufpr.br
URL: http://www.leg.ufpr.br/

Journal of Statistical Software http://www.jstatsoft.org/
published by the Foundation for Open Access Statistics http://www.foastat.org/
April 2018, Volume 84, Issue 4 *Submitted:* 2016-06-13
doi:10.18637/jss.v084.i04 *Accepted:* 2017-06-02