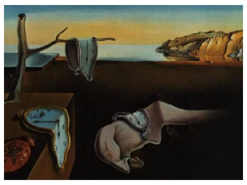

Journal of Statistical Software

December 2017, Volume 82, Issue 9. *doi: 10.18637/jss.v082.i09*

jmcm: An R Package for Joint Mean-Covariance Modeling of Longitudinal Data

Jianxin Pan
The University of Manchester

Yi Pan
The University of Manchester

Abstract

Longitudinal studies commonly arise in various fields such as psychology, social science, economics and medical research, etc. It is of great importance to understand the dynamics in the mean function, covariance and/or correlation matrices of repeated measurements. However, high-dimensionality (HD) and positive-definiteness (PD) constraints are two major stumbling blocks in modeling of covariance and correlation matrices. It is evident that Cholesky-type decomposition based methods are effective in dealing with HD and PD problems, but those methods were not implemented in statistical software yet, causing a difficulty for practitioners to use. In this paper, we first introduce recently developed Cholesky decomposition based methods for joint modeling of mean and covariance structures, namely modified Cholesky decomposition (MCD), alternative Cholesky decomposition (ACD) and hyperspherical parameterization of Cholesky factor (HPC). We then introduce our newly developed R package **jmcm** which is currently able to handle longitudinal data that follows a Gaussian distribution using the MCD, ACD and HPC methods. The use of package **jmcm** is illustrated and a comparison of those methods is made through the analysis of two real datasets.

Keywords: Cholesky decomposition, covariance matrix estimator, longitudinal data, joint mean-covariance models.

1. Introduction

A longitudinal study usually involves repeated observations of the same variables over a long period of time and is often used in psychology, sociology and medical research. The covariance matrix plays a prominent role in analyzing data from longitudinal studies since the components of collected measurements within the same subject are not independent. A good covariance modeling approach improves statistical inference of the mean of interest and the covariance structure itself may be of scientific interest in some circumstances (Diggle and Verbyla 1998).

However, modeling of covariance structure is challenging because the estimated covariance matrices in general should be positive definite and there are many parameters in covariance matrices. To overcome these two obstacles, Pourahmadi (1999) advocated a data-driven joint mean-covariance modeling method based on a modified Cholesky decomposition (MCD) of the marginal within-subject covariance matrix. The decomposition leads to a reparameterization where entries can be interpreted in terms of innovation variances and autoregressive coefficients. See Pan and Mackenzie (2003) for a related discussion. Another Cholesky-type decomposition (ACD) proposed by Chen and Dunson (2003) can be understood as modeling certain innovation variances and moving average parameters, and the method is compared with MCD in detail by Pourahmadi (2007). These two Cholesky-type approaches demonstrate parsimonious and effective strategies, but their corresponding variance functions cannot be directly interpreted as those of the repeated observations. Therefore additional efforts are needed for interpreting the variance and covariance functions. More recently, Zhang, Leng, and Tang (2015) considered a regression approach based on the standard Cholesky decomposition of the correlation matrix and the hyperspherical parameterization (HPC) of its Cholesky factor, where parameters are directly interpretable with respect to variance and correlation. A brief review of these approaches is presented in the following sections. Note this paper is not an exhaustive survey, and many other covariance structure modeling methods are also commonly used in the literature, see Fan, Liao, and Liu (2016) for a more general overview on the estimation of large covariance and precision matrices.

Software for analyzing longitudinal data using some conventional approaches has been implemented in the R environment for statistical computing and graphics (R Core Team 2017) for many years. For instance, several packages provide functions for determining maximum likelihood estimates of the parameters in a linear mixed-effect model (LMM) that incorporates both fixed effects and random effects in the linear predictor, such as the `lme` function in package **nlme** (Pinheiro, Bates, DebRoy, Sarkar, and R Core Team 2017) and the `lmer` function in package **lme4** (Bates, Mächler, Bolker, and Walker 2015). Similar commercial statistical programs are also available for LMM such as `PROC MIXED` in SAS (SAS Institute Inc. 2013), `MIXED` in SPSS (SPSS Inc. 2015) and `fitlme` in MATLAB (The MathWorks Inc. 2015). The method of generalized estimating equation (GEE; Liang and Zeger 1986) is widely used as it focuses on models for the mean of the correlated observations without fully specifying the joint distribution of the responses. Several implementations of GEE are available through packages **gee** (Carey, Lumley, and Ripley 2015) and **geepack** (Halekoh, Højsgaard, and Yan 2006). The Gaussian copula model provides a flexible general framework for marginal regression analysis of continuous, discrete and categorical responses, and is available through package **gcmr** (Masarotto and Varin 2012, 2017). However all of these procedures are based on specific model assumptions like existence of an expectation or homogeneous variances and are not intuitive for a joint mean-covariance modeling framework. In this paper we focus on the joint mean-covariance modeling for both balanced and unbalanced longitudinal data, and we present a user friendly R package **jmcm** (Pan and Pan 2017) which is freely available from the Comprehensive R Archive Network (CRAN) at `https://CRAN.R-project.org/package=jmcm` and that can be used to handle such joint models. For efficiency, the core part of package **jmcm** is implemented in compiled C++ code using **Rcpp** (Eddelbuettel and François 2011; Eddelbuettel 2013) and **RcppArmadillo** (Eddelbuettel and Sanderson 2014) for numerical linear algebra. All the implemented R functions are well documented with some examples. The main objective of this paper is to introduce the joint mean-covariance modeling approaches in package **jmcm** to a wide audience of statisticians and practitioners who need to analyze longitudinal data.

The rest of the paper is organized as follows. In Section 2 we present the joint mean-covariance modeling methods, and discuss different choices for modeling strategies of covariance structure mentioned above. Furthermore, we consider the maximum likelihood estimations for each type of the models, with particular emphasize on numerical optimization techniques. Section 3 provides a detailed overview on the implementation of the methods introduced in Section 2 using the package **jmcm** and gives a brief illustration of the computational tools and Section 4 concludes the paper with further discussions.

2. Joint mean-covariance modeling framework

2.1. Joint mean-covariance models

Denote longitudinal measurements by $y_i = (y_{i1}, y_{i2}, \ldots, y_{im_i})^\top$ $(i = 1, 2, \ldots, n)$ that are collected from n subjects and measured at time points $t_i = (t_{i1}, t_{i2}, \ldots, t_{im_i})^\top$. Here we assume the number of measurements m_i and time t_i are subject specific, so that unbalanced longitudinal datasets with observations taken at irregular time points can be modeled.

The basic linear model used in the joint mean-covariance modeling frame work of longitudinal data analysis can be described by the distribution of a vector-valued random response variable y_i, which is assumed to be multivariate normal,

$$y_i \sim N_{m_i}(\mu_i, \Sigma_i),$$

where $\mu_i = (\mu_{i1}, \mu_{i2}, \ldots, \mu_{im_i})^\top$ is an $m_i \times 1$ vector and Σ_i is an $m_i \times m_i$ within-subject covariance matrix. The mean μ_i of y_i is usually modeled by a linear regression,

$$\mu_i = X_i\beta, \tag{1}$$

where X_i denotes an $m_i \times (p+1)$ model matrix with an intercept on the first column followed by covariates of the ith subject, β is a $(p+1) \times 1$ regression coefficient vector. The subject-specific within-subject covariance matrix, Σ_i, may be modeled similarly based on different decomposition approaches, and will be discussed in detail in the following sections.

Estimates of the joint mean-covariance model parameters θ, including $\theta_1 = \beta$ in the mean model and other unspecified parameters θ_2 in the covariance matrices, can be obtained by maximum likelihood (ML) estimation. In particular, a maximum likelihood estimator (MLE) of the unknown parameter vector is defined as any solution $\hat{\theta}_n$ of:

$$\hat{\theta}_n = \arg\min_{\theta \in \Theta} \left\{-2l(\theta)\right\}, \tag{2}$$

where

$$-2l(\theta) = \sum_{i=1}^{n} \log |\Sigma_i| + \sum_{i=1}^{n} (y_i - \mu_i(\theta_1))^\top \Sigma_i^{-1}(\theta_2)(y_i - \mu_i(\theta_1)) \tag{3}$$

is minus twice the log-likelihood function without the constant term. Note that this is the form of the log-likelihood function that is implemented by default in the package, though the value of the full log-likelihood including the constant term can be easily obtained by setting a specific option before model fitting. After obtaining the score functions $U(\theta) = (U(\theta_1)^\top, U(\theta_2)^\top)^\top$ based on $f(\theta) = -2l(\theta)$ by direct calculations, we then estimate θ via the iterative quasi-Newton method (BFGS) that solves the score equations. More specifically, we apply the following quasi-Newton algorithm.

1. Select an initial value $\theta^{(0)} = ((\theta_1^{(0)})^\top, (\theta_2^{(0)})^\top)^\top$. Set the superscript $k = 0$ for the iteration number. A convenient initial value for $\theta_1^{(0)} = \beta^{(0)}$ is its ordinary least-squares estimate, $\beta^{(0)} = (\sum_{i=1}^n X_i^\top X_i)^{-1}(\sum_{i=1}^n X_i^\top y_i)$ while the initial value for $\theta_2^{(0)}$ is set to form a diagonal covariance matrix, and will be discussed in detail respectively with the choice of covariance structure models later.

2. Initialize score function $U^{(0)} = U(\theta^{(0)})$ and the inverse Hessian $H^{(0)}$ as identity matrix.

3. Update the search direction (Newton step) and compute the step size $\tilde{\lambda}$ by performing an approximate line minimization:
$$p^{(k)} = -H^{(k)}U^{(k)}, \quad \tilde{\lambda} = \arg\min_{0<\tilde{\lambda}\leq 1} f(\theta^{(k)} + \tilde{\lambda}p^{(k)}). \tag{4}$$

4. Update θ as
$$\theta^{(k+1)} = \theta^{(k)} + \tilde{\lambda}p^{(k)} \tag{5}$$
and then the new gradient
$$U^{(k+1)} = U(\theta^{(k+1)}). \tag{6}$$

5. Compute the difference $\theta^{(k+1)} - \theta^{(k)}$ and $U^{(k+1)} - U^{(k)}$ and update the inverse Hessian with the BFGS updating formula
$$\begin{aligned} H_{i+1} = H_i &+ \frac{(\theta^{(k+1)} - \theta^{(k)})(\theta^{(k+1)} - \theta^{(k)})^\top}{(\theta^{(k+1)} - \theta^{(k)})^\top(U^{(k+1)} - U^{(k)})} \\ &- \frac{[H^{(k)}(U^{(k+1)} - U^{(k)})][H^{(k)}(U^{(k+1)} - U^{(k)})]^\top}{(U^{(k+1)} - U^{(k)})^\top H^{(k)}(U^{(k+1)} - U^{(k)})} \\ &+ (U^{(k+1)} - U^{(k)})^\top H^{(k)}(U^{(k+1)} - U^{(k)})uu^\top, \end{aligned} \tag{7}$$
where u is defined as the following vector
$$u \equiv \frac{(\theta^{(k+1)} - \theta^{(k)})}{(\theta^{(k+1)} - \theta^{(k)})^\top(U^{(k+1)} - U^{(k)})} - \frac{H^{(k)}(U^{(k+1)} - U^{(k)})}{(U^{(k+1)} - U^{(k)})^\top H^{(k)}(U^{(k+1)} - U^{(k)})}. \tag{8}$$

6. Set $k = k + 1$ and repeat Steps 3 to 5 until a pre-specified criterion is met.

See Press, Teukolsky, Vetterling, and Flannery (2007) for a more detailed discussion of the BFGS optimization algorithm with line-search and its implementations. Note that currently only the BFGS algorithm is implemented in the package since it proves to be one of the best quasi-Newton methods for solving smooth unconstrained optimization problems and works very well in our problem. Other quasi-Newton algorithms will also be implemented as alternatives in the future. In practice, we find that estimation of the parameters θ of the joint mean-covariance model can be further improved by solving for the parameters one by one with the other parameters fixed in the optimization, and this will be discussed in more detail in the following sections.

2.2. Modified Cholesky decomposition (MCD)

The two major obstacles in modeling covariance matrices are high-dimensionality (HD) and positive-definiteness (PD). The HD problem can be largely reduced by introducing the regression techniques and the PD constraint can be potentially removed by employing the Cholesky decomposition in covariance structure modeling (Pourahmadi 2013). The standard Cholesky decomposition of an $m_i \times m_i$ positive definite covariance matrix is of the following form

$$\Sigma_i = C_i C_i^\top, \tag{9}$$

where $C_i = (c_{ijk})$ is a lower triangular matrix with positive diagonal elements and its entries are difficult to interpret (Pinheiro and Bates 1996). We will find that the task of statistical interpretation can be much easier by reducing C_i to a unit lower triangular matrix by post- or pre-multiplying with the inverse of $D_i = \text{diag}(c_{i11}, c_{i22}, \dots, c_{im_im_i})$.

Defining modified Cholesky decomposition (MCD)

The first case, post-multiplying C_i by the inverse of D_i, leads to the modified Cholesky decomposition (MCD) and keeps D_i inside (Zhang and Leng 2012),

$$\Sigma_i = (C_i D_i^{-1})(D_i D_i)(D_i^{-1} C_i^\top) = L_i D_i^2 L_i^\top, \tag{10}$$

or in another more commonly used form (Pourahmadi 1999),

$$T_i \Sigma_i T_i^\top = D_i^2, \tag{11}$$

where $T_i = L_i^{-1}$ and $L_i = C_i D_i^{-1}$ can be considered as a standardized version of C_i, dividing each column by its corresponding diagonal entry (Maadooliat, Pourahmadi, and Huang 2013). The below-diagonal entries of T_i are the negatives of the so-called generalized autoregressive parameters (GARPs), ϕ_{ijk}, in

$$y_{ij} = \mu_{ij} + \sum_{k=1}^{j-1} \phi_{ijk}(y_{ik} - \mu_{ik}) + \epsilon_{ij}, \tag{12}$$

the AR model for the actual measurements on subject i. The diagonal entries of D_i^2 are the innovation variance $\sigma_{ij}^2 = \mathsf{VAR}(\epsilon_{ij})$, see Pourahmadi (1999) for the details. It is helpful to invert Equation 12 by using $y_{i1} = \epsilon_{i1}$ and repeating substitution for y_{it} in terms of ϵ_{it} to obtain

$$y_{ij} - \mu_{ij} = \epsilon_{ij} + \sum_{k=1}^{j-1} \tilde{\phi}_{ijk}\epsilon_{ik} \quad (j = 2, \dots, m_i), \tag{13}$$

where the matrix form reveals $L_i = (\tilde{\phi}_{ijk})$ so that its entries on the jth row can be interpreted as regression parameters when y_{ij} is regressed on the present and past innovations $\epsilon_{ij}, \epsilon_{i(j-1)}, \dots, \epsilon_{i1}$. Then we can prove

$$\mathsf{COV}(y_{is}, y_{it}) = \sum_{k=1}^{\min(s,t)} \tilde{\phi}_{isk}\tilde{\phi}_{itk}\sigma_{ik}^2 \tag{14}$$

by setting $\tilde{\phi}_{ijj} = 1$ and $\tilde{\phi}_{ijk} = 0$ for $j < k$ and $1 \leq s, t \leq m_i$. Thus, the correlation coefficient between y_{is} and y_{it} depends on both the $\tilde{\phi}_{ijk}$'s and the σ^2_{ij}'s.

Maximum likelihood estimation of MCD

Using the idea of linear models and employing covariates as in Pan and Mackenzie (2003), the unconstrained parameters $\zeta_{ij} \equiv \log \sigma^2_{ij}$ and ϕ_{ijk} are modeled as

$$\zeta_{ij} = z_{ij}^\top \lambda, \quad \phi_{ijk} = w_{ijk}^\top \gamma, \tag{15}$$

where z_{ij} and w_{ijk} are $(d+1)\times 1$ and $(q+1)\times 1$ vectors of covariates, $\lambda = (\lambda_0, \lambda_1, \ldots, \lambda_d)^\top$ and $\gamma = (\gamma_0, \gamma_1, \ldots, \gamma_q)^\top$ are unknown parameters for the innovation variances and autoregressive coefficients, respectively.

Under the model in (15), minus twice the log-likelihood function, except for a constant, is given by

$$-2l = \sum_{i=1}^n \log |T_i^{-1} D_i^2 T_i^{-\top}| + \sum_{i=1}^n r_i^\top T_i^\top D_i^{-2} T_i r_i, \tag{16}$$

where $r_{ij} = y_{ij} - x_{ij}^\top \beta$ is the jth element of $r_i = y_i - X_i\beta$, the vector of residuals for the ith subject.

The maximum likelihood estimating equations for β, λ and γ become

$$\begin{cases} U_1(\beta) = \sum_{i=1}^n X_i^\top \Sigma_i^{-1}(y_i - X_i\beta), \\ U_2(\lambda) = \frac{1}{2}\sum_{i=1}^n Z_i^\top (D_i^{-2} e_i - 1_{m_i}), \\ U_3(\gamma) = \sum_{i=1}^n G_i^\top D_i^{-2}(r_i - G_i\gamma), \end{cases} \tag{17}$$

where the matrix G_i, of order $m_i \times (q+1)$, has typical row $g_{ij}^\top = \sum_{k=1}^{j-1} r_{ik} w_{ijk}^\top$. Also, $Z_i = (z_{i1}^\top, z_{i2}^\top, \ldots, z_{im_i}^\top)^\top$, $e_i = (e_{i1}, e_{i2}, \ldots, e_{im_i})^\top$ with $e_{ij} = (r_{ij} - \hat{r}_{ij})^2$ and $\hat{r}_{ij} = \sum_{k=1}^{j-1} \phi_{ijk} r_{ik}$, are the $m_i \times (d+1)$ matrix of covariates and the $m_i \times 1$ vector of squared fitted residuals respectively, and 1_{m_i} is the $m_i \times 1$ vector of 1's.

The initial guess $\beta^{(0)}$ can be set by employing a simple linear regression:

```
R> lm.obj <- lm.fit(X, Y)
R> bta0 <- coef(lm.obj)
```

After extracting the residuals from the linear model, the starting value $\lambda^{(0)}$ is obtained by fitting the linear regression model in (15) while $\gamma^{(0)}$ is simply assumed to be a vector of 0's so that T_i is constructed as an identity matrix:

```
R> res <- resid(lm.obj)
R> lmd0 <- coef(lm.fit(Z, log(res^2)))
R> gma0 <- rep(0, lgma)
```

We then estimate θ by minimizing expression in (16) via the iterative quasi-Newton algorithm, as explained in Section 2.1, after substitution of $U(\theta)$ by $(-2U_1(\beta)^\top, -2U_2(\lambda)^\top, -2U_3(\gamma)^\top)^\top$. Since the solutions satisfy Equation 17 and the parameters β, λ, γ are asymptotically independent (Ye and Pan 2006), the three parameters can also be sequentially solved one by one with the other parameters kept fixed. More specifically, we apply the following algorithm.

1. Initialize the parameters as $\theta^{(0)} = ((\beta^{(0)})^\top, (\lambda^{(0)})^\top, (\gamma^{(0)})^\top)^\top$. Set $k = 0$.
2. Compute Σ_i by using $\lambda^{(k)}$ and $\gamma^{(k)}$. Update β as
$$\beta = \left(\sum_{i=1}^{n} X_i^\top \Sigma_i^{-1} X_i\right)^{-1} \sum_{i=1}^{n} X_i^\top \Sigma_i^{-1} y_i.$$
3. Given $\beta = \beta^{(k+1)}$ and $\gamma = \gamma^{(k)}$, update λ via the iterative quasi-Newton algorithm after substitution of $f(\theta) = -2l(\theta)$ by $f(\lambda)$ and $U(\theta)$ by $-2U_2(\lambda)$ since there is no explicit form for the solution of λ.
4. Given $\beta = \beta^{(k+1)}$ and $\lambda = \lambda^{(k+1)}$, update γ as
$$\gamma = \left(\sum_{i=1}^{n} G_i^\top D_i^{-2} G_i\right)^{-1} \sum_{i=1}^{n} G_i^\top D_i^{-2} r_i.$$
5. Update the search direction as
$$p^{(k)} = \theta^{(k+1)} - \theta^{(k)}.$$
Compute step size $\tilde{\lambda}$ by performing an approximate line minimization
$$\tilde{\lambda} = \arg \min_{0<\tilde{\lambda}\leq 1} f(\theta^{(k)} + \tilde{\lambda} p^{(k)}).$$
6. Update $\theta^{(k+1)}$ again as
$$\theta^{(k+1)} = \theta^{(k)} + \tilde{\lambda} p^{(k)}.$$
7. Set $k = k + 1$ and repeat Steps 2 to 6 until a pre-specified criterion is met.

2.3. Alternative Cholesky decomposition (ACD)

Defining alternative Cholesky decomposition (ACD)

The second case, pre-multiplying C_i by the inverse of D_i, leads to alternative Cholesky decomposition (ACD; Chen and Dunson 2003) and keeps D_i outside,
$$\Sigma_i = D_i(D_i^{-1}C_i)(C_i^\top D_i^{-1})D_i = D_i \tilde{L}_i \tilde{L}_i^\top D_i,$$
where $\tilde{L}_i = D_i^{-1}C_i$ is obtained from a slightly different standardized C_i, dividing each row by its corresponding diagonal entry (Maadooliat *et al.* 2013).

For statistical interpretation of the below-diagonal entries of $\tilde{L}_i$, it is clear that $D_i^{-1}(y_i - \mu_i)$ has $\tilde{L}_i\tilde{L}_i^\top$ as the standard Cholesky decomposition of its covariance matrix and $\epsilon_i = (D_i\tilde{L}_i)^{-1}(y_i - \mu_i)$, its vector of innovations, has $\mathsf{COV}(\epsilon_i) = I_{m_i}$. Thus, with $\tilde{L}_i = (\tilde{\phi}_{ijk})$ and $D_i = (\sigma_{ij})$, we can obtain the variable order, MA representation for the standardized residual from $D_i^{-1}(y_i - \mu_i) = \tilde{L}_i\epsilon_i$ as

$$(y_{ij} - \mu_{ij})/\sigma_{ij} = \epsilon_{ij} + \sum_{k=1}^{j-1} \tilde{\phi}_{ijk}\epsilon_{ik}. \tag{18}$$

Then we can prove

$$\mathsf{COV}(y_{is}, y_{it}) = \sigma_{is}\sigma_{it} \sum_{k=1}^{\min(s,t)} \tilde{\phi}_{itk}\tilde{\phi}_{isk} \tag{19}$$

for any $1 \le s, t \le m_i$, so that the correlation coefficient between y_{is} and y_{it} given by

$$\mathsf{CORR}(y_{is}, y_{it}) = \frac{\sum_{k=1}^{\min(s,t)} \tilde{\phi}_{itk}\tilde{\phi}_{isk}}{\sqrt{(\sum_{k=1}^{s} \tilde{\phi}_{isk}^2 \sum_{k=1}^{t} \tilde{\phi}_{itk}^2)}} \tag{20}$$

is determined solely by $\tilde{\phi}_{ijk}$'s.

Maximum likelihood estimation of ACD

Following the similar approach in Pan and Mackenzie (2003), the log-innovation variance $\zeta_{ij} = \log\sigma_{ij}^2$ and moving average parameters $\tilde{\phi}_{ijk}$ in ACD are modeled as

$$\zeta_{ij} = z_{ij}^\top\lambda, \quad \tilde{\phi}_{ijk} = v_{ijk}^\top\gamma, \tag{21}$$

where z_{ij} and v_{ijk} are $(d+1)\times 1$ and $(q+1)\times 1$ vectors of covariates, $\lambda = (\lambda_0, \lambda_1, \ldots, \lambda_d)^\top$ and $\gamma = (\gamma_0, \gamma_1, \ldots, \gamma_q)^\top$ are unknown parameters for the innovation variances and moving average regression coefficients, respectively.

Under the model in (21), minus twice the log-likelihood function, except for a constant, is given by

$$-2l = \sum_{i=1}^{n} \log|D_i\tilde{L}_i\tilde{L}_i^\top D_i| + \sum_{i=1}^{n} r_i^\top D_i^{-1}\tilde{L}_i^{-\top}\tilde{L}_i^{-1}D_i^{-1}r_i, \tag{22}$$

where $r_{ij} = y_{ij} - x_{ij}^\top\beta$ is the jth element of $r_i = y_i - X_i\beta$, the vector of residuals for the ith subject.

The score functions can be obtained and simplified as

$$\begin{cases} U_1(\beta) = \sum_{i=1}^{n} X_i^\top\Sigma_i^{-1}(y_i - X_i\beta), \\ U_2(\lambda) = \dfrac{1}{2}\sum_{i=1}^{n} Z_i^\top(h_i - 1_{m_i}), \\ U_3(\gamma) = \sum_{i=1}^{n}(\epsilon_i^\top \otimes I_{m_i})\dfrac{\partial\tilde{L}_i^\top}{\partial\gamma}\tilde{L}_i^{-\top}\epsilon_i, \end{cases} \tag{23}$$

where $Z_i = (z_{i1}^\top, z_{i2}^\top, \ldots, z_{im_i}^\top)^\top$, $h_i = \mathrm{diag}(\tilde{L}_i^{-\top}\tilde{L}_i^{-1}D_i^{-1}r_i r_i^\top D_i^{-1})$, $\epsilon_i = (\epsilon_{i1}, \ldots, \epsilon_{im_i})^\top = \tilde{L}_i^{-1}D_i^{-1}r_i$, thus $\epsilon_{i1}, \ldots, \epsilon_{im_i}$ are independent standard normal random variables, and I_{m_i} is an $m_i \times m_i$ identity matrix.

Since the covariance structures of MCD and ACD are quite close, the initial guess of parameters $\beta^{(0)}$, $\lambda^{(0)}$ and $\gamma^{(0)}$ in ACD can be obtained using the same approach as described for determining the initial parameter setting of MCD:

```
R> lm.obj <- lm.fit(X, Y)
R> bta0 <- coef(lm.obj)
R> res <- resid(lm.obj)
R> lmd0 <- coef(lm.fit(Z, log(res^2)))
R> gma0 <- rep(0, lgma)
```

We then estimate θ by minimizing expression in (22) via the iterative quasi-Newton algorithm, as explained in Section 2.1, after substitution of $U(\theta)$ by $(-2U_1(\beta)^\top, -2U_2(\lambda)^\top, -2U_3(\gamma)^\top)^\top$.

Since the solutions satisfy Equation 23 and the parameters λ and γ are not asymptotically orthogonal (Maadooliat *et al.* 2013), the three parameters can be split into two groups, $\theta_1 = \beta$ and $\theta_2 = (\lambda^\top, \gamma^\top)^\top$ and can be sequentially solved one by one with the other parameters kept fixed. More specifically, we apply the following algorithm.

1. Initialize the parameters as $\theta^{(0)} = ((\theta_1^{(0)})^\top, (\theta_2^{(0)})^\top)^\top = ((\beta^{(0)})^\top, (\lambda^{(0)})^\top, (\gamma^{(0)})^\top)^\top$. Set $k = 0$.

2. Compute Σ_i by using $\lambda^{(k)}$ and $\gamma^{(k)}$. Update $\theta_1 = \beta$ as

$$\beta = \left(\sum_{i=1}^n X_i^\top \Sigma_i^{-1} X_i\right)^{-1} \sum_{i=1}^n X_i^\top \Sigma_i^{-1} y_i.$$

3. Given $\beta = \beta^{(k+1)}$, update θ_2 via the iterative quasi-Newton algorithm after substitution of $f(\theta) = -2l(\theta)$ by $f(\theta_2)$ and $U(\theta)$ by $(-2U_2(\lambda)^\top, -2U_3(\gamma)^\top)^\top$.

4. Update the search direction as

$$p^{(k)} = \theta^{(k+1)} - \theta^{(k)}.$$

   Compute the step size $\tilde{\lambda}$ by performing an approximate line minimization

$$\tilde{\lambda} = \arg\min_{0<\tilde{\lambda}\leq 1} f(\theta^{(k)} + \tilde{\lambda} p^{(k)}).$$

5. Update $\theta^{(k+1)}$ again as

$$\theta^{(k+1)} = \theta^{(k)} + \tilde{\lambda} p^{(k)}.$$

6. Set $k = k + 1$ and repeat Steps 2 to 5 until a pre-specified criterion is met.

2.4. Hyperspherical parameterization of the Cholesky factor (HPC)

Even though modified Cholesky decomposition (MCD; Pourahmadi 1999) and alternative Cholesky decomposition (ACD; Chen and Dunson 2003) provide parsimonious unconstrained and statistically interpretable parameterizations of a covariance matrix, the innovation variance is not the same as the marginal variances of the repeated measurements within the same subject.

Defining the hyperspherical parameterization of the Cholesky factor (HPC)

It is well known that the variance-correlation decomposition has the form

$$\Sigma_i = H_i R_i H_i, \tag{24}$$

where $H_i = \text{diag}(\sigma_{i1}, \sigma_{i2}, \ldots, \sigma_{im_i})$ with σ_{ij} being the standard deviation of the jth measurement for subject i and $R_i = (\rho_{ijk})_{j,k=1}^{m_i}$ is the correlation matrix of y_i with $\rho_{ijk} = \mathsf{CORR}(y_{ij}, y_{ik})$ being the correlation between the jth and kth observations of the ith subject. By using this decomposition, one can directly model the variances and correlations of observations separately.

Not surprisingly, the development of a regression method to model the correlation structure proves to be difficult. Specifically, a correlation matrix must be positive semi-definite and symmetric with 1's as the main diagonal entries and values between -1 and 1 as the off-diagonal entries. The new challenge is mitigated by employing the standard Cholesky decomposition on the correlation matrix R_i,

$$R_i = B_i B_i^\top \tag{25}$$

and parameterizing its Cholesky factor B_i via hyperspherical co-ordinates (HPC; Zhang *et al.* 2015),

$$B_i = \begin{bmatrix} 1 & 0 & 0 & \cdots & 0 \\ c_{i21} & s_{i21} & 0 & \cdots & 0 \\ c_{i31} & c_{i32}s_{i31} & s_{i32}s_{i31} & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ c_{im_i1} & c_{im_i2}s_{im_i1} & c_{im_i3}s_{im_i2}s_{im_i1} & \cdots & \prod_{l=1}^{m_i-1} s_{im_il} \end{bmatrix},$$

where $c_{ijk} = \cos(\theta_{ijk})$ and $s_{ijk} = \sin(\theta_{ijk})$.

Equivalently, the non-zeros entries in the lower triangular matrix $B_i = (b_{ijk})$ are given as $b_{i11} = 1$, $b_{ij1} = c_{ij1} = \cos(\theta_{ij1})$ for $j = 1, 2, \ldots, m_i$ and

$$b_{ijk} = \begin{cases} \cos(\theta_{ijk}) \prod_{l=1}^{k-1} \sin(\theta_{ijl}), & 2 \le k < j \le m_i, \\ \prod_{l=1}^{k-1} \sin(\theta_{ijl}), & k = j, j = 2, \ldots, m_i, \end{cases}$$

where θ_{ijk} are some angles in $[0, \pi)$ (Rapisarda, Brigo, and Mercurio 2007).

Maximum likelihood estimation of HPC

The correlation matrix R_i is guaranteed to be positive semi-definite since it is constructed by its corresponding standard Cholesky factor B_i and the angle parameters in B_i are unconstrained except that $\theta_{ijk} \in [0, \pi)$. We are free to model the log-variances and angles through regression by using some covariates

$$\log \sigma_{ij}^2 = z_{ij}^\top \lambda, \quad \theta_{ijk} = g_{ijk}^\top \gamma. \tag{26}$$

As for the range of θ_{ijk}, our experience from data analysis and simulation studies indicates all the estimated θ_{ijk}s fall in the range $[0, \pi)$. Transformation such as the inverse tangent transformation can be applied to ensure that θ_{ijk} definitely falls in $[0, \pi)$, and can be implemented in a future version. Under the model in (26), minus twice the log-likelihood function, except for a constant, is given by

$$-2l = \sum_{i=1}^n \log |H_i B_i B_i^\top H_i| + \sum_{i=1}^n r_i' H_i^{-1} B_i^{-\top} B_i^{-1} H_i^{-1} r_i, \tag{27}$$

where $r_{ij} = y_{ij} - x_{ij}^\top \beta$ is the jth element of $r_i = y_i - X_i\beta$, the vector of residuals for the ith subject.

The score functions can be obtained and simplified as

$$\begin{cases} U_1(\beta) = \displaystyle\sum_{i=1}^n X_i^\top \Sigma_i^{-1}(y_i - X_i\beta), \\ U_2(\lambda) = \dfrac{1}{2}\displaystyle\sum_{i=1}^n Z_i^\top (h_i - 1_{m_i}), \\ U_3(\gamma) = \displaystyle\sum_{i=1}^n \left((\epsilon_i^\top \otimes I_{m_i}) \frac{\partial B_i^\top}{\partial \gamma} B_i^{\top -1} \epsilon_i - \sum_{j=1}^{m_i} \frac{\partial \log B_{ijj}}{\partial \gamma} \right), \end{cases} \tag{28}$$

where $Z_i = (z_{i1}^\top, z_{i2}^\top, \ldots, z_{im_i}^\top)^\top$, $h_i = \mathrm{diag}(B_i^{-\top} B_i^{-1} H_i^{-1} r_i r_i^\top H_i^{-1})$, $\epsilon_i = (\epsilon_{i1}, \ldots, \epsilon_{im_i})^\top = B_i^{-1} H_i^{-1} r_i$, thus $\epsilon_{i1}, \ldots, \epsilon_{im_i}$ are independent standard normal random variables, and I_{m_i} is an $m_i \times m_i$ identity matrix.

The initial guess $\beta^{(0)}$ can be set by employing a simple linear regression:

```
R> lm.obj <- lm.fit(X, Y)
R> bta0 <- coef(lm.obj)
```

After extracting residuals from the linear model, the starting value $\lambda^{(0)}$ is obtained by fitting the linear regression model in (26) while $\gamma^{(0)}$ is simply assumed to be a vector whose first element is $\frac{1}{2}\pi$ and followed by 0's so that B_i is constructed as an identity matrix:

```
R> res <- resid(lm.obj)
R> lmd0 <- coef(lm.fit(Z, log(res^2)))
R> gma0 <- c(pi / 2, rep(0, lgma - 1))
```

We then estimate θ by minimizing the expression in (27) via the iterative quasi-Newton algorithm, as explained in Section 2.1, after substitution of $U(\theta)$ by $(-2U_1(\beta)^\top, -2U_2(\lambda)^\top, -2U_3(\gamma)^\top)^\top$.

Since the solutions satisfy Equation 28 and the parameters λ and γ are not asymptotically independent (Zhang *et al.* 2015), the three parameters can be split into two groups, $\theta_1 = \beta$ and $\theta_2 = (\lambda^\top, \gamma^\top)^\top$ and can be sequentially solved one by one with the other parameters kept fixed. More specifically, we apply the following algorithm.

1. Initialize the parameters as $\theta^{(0)} = ((\theta_1^{(0)})^\top, (\theta_2^{(0)})^\top)^\top = ((\beta^{(0)})^\top, (\lambda^{(0)})^\top, (\gamma^{(0)})^\top)^\top$. Set $k = 0$.

2. Compute Σ_i by using $\lambda^{(k)}$ and $\gamma^{(k)}$. Update $\theta_1 = \beta$ as
$$\beta = \left(\sum_{i=1}^n X_i^\top \Sigma_i^{-1} X_i\right)^{-1} \sum_{i=1}^n X_i^\top \Sigma_i^{-1} y_i.$$

3. Given $\beta = \beta^{(k+1)}$, update θ_2 via the iterative quasi-Newton algorithm after substitution of $f(\theta) = -2l(\theta)$ by $f(\theta_2)$ and $U(\theta)$ by $(-2U_2(\lambda)^\top, -2U_3(\gamma)^\top)^\top$.

4. Update the search direction as
$$p^{(k)} = \theta^{(k+1)} - \theta^{(k)}.$$
Compute the step size $\tilde{\lambda}$ by performing an approximate line minimization
$$\tilde{\lambda} = \arg \min_{0<\tilde{\lambda}\leq 1} f(\theta^{(k)} + \tilde{\lambda} p^{(k)}).$$

5. Update $\theta^{(k+1)}$ again as
$$\theta^{(k+1)} = \theta^{(k)} + \tilde{\lambda} p^{(k)}.$$

6. Set $k = k + 1$ and repeat Steps 2 to 5 until a pre-specified criterion is met.

2.5. Comparison of MCD, ACD and HPC

For modeling the covariance and correlation structure, the three discussed Cholesky-type decomposition-based approaches have been demonstrated to be effective in the sense that the estimated covariance and correlation are guaranteed positive (semi-)definite, and the number of parameters is considerably reduced through regression techniques.

It is clear that MCD and ACD have a closer relationship since they are constructed in a similar way through standardization of the Cholesky factor C_i, and the resulting unconstrained parameters have a nice statistical interpretation in terms of innovation variance, autoregressive and moving average parameters respectively. The main drawbacks of these two approaches are the potential need for a natural order (e.g., time series), which makes it difficult to find a reasonable statistical interpretation and may result in a different estimation of the covariance and correlation matrix with each single ordering. A recent application of the Cholesky-based approach for estimating the covariance matrix of multiple stocks within a portfolio and a more detailed discussion of the ordering problem can be found in Dellaportas and Pourahmadi (2012) and Pedeli, Fokianos, and Pourahmadi (2015). Additional effort and extra care are needed in practice for interpreting their corresponding variance and correlation functions. Moreover, owing to the decomposition, the resulting correlation function of MCD depends on both the innovation variance and autoregressive parameters, indicating that MCD is not robust against the misspecification of the innovation variance when the correlation is the main interest (Maadooliat *et al.* 2013). We also need to note that MCD is the most computationally efficient approach among the three approaches due to the fact that its Fisher information matrix is block diagonal (Ye and Pan 2006).

The parameterization of HPC is very attractive because the resulting parameters are unconstrained and directly interpretable with respect to the variances and correlations. The angles in the Cholesky factor of the correlation matrix have a geometric connection with correlations. However, modeling covariance and correlation using HPC can be computationally expensive since the problem of estimating the Cholesky factor is transformed into the problem that actually first estimates a matrix consisting of angles. For details see Zhang *et al.* (2015).

3. Examples of use

3.1. Analysis of a balanced longitudinal dataset

In this section, we provide our first example that illustrates how to apply joint mean-covariance models in analyzing a balanced longitudinal data by using **jmcm**. Kenward (1987) analyzed an experiment in which cattle were assigned randomly to two treatment groups A and B, and their weights were recorded. Thirty animals received treatment A and another thirty received treatment B. The animals were weighted $n = 11$ times over a 133-day period; the first 10 measurements on each animal were made at two-week intervals and the final measurement was made one week later. Since no observation was missing, it is considered to be a balanced longitudinal dataset. The data is loaded simply using the `data()` instruction:

```
R> library("jmcm")
R> data("cattle", package = "jmcm")
R> head(cattle)
```

```
  id day group weight
1  1   0     A    233
2  1  14     A    224
3  1  28     A    245
4  1  42     A    258
5  1  56     A    271
6  1  70     A    287
```

We present in Figure 1 the subject-specific longitudinal profiles of the cattle data using the following code:

```
R> library("lattice")
R> xyplot(weight ~ day | group, group = id, data = cattle, xlab = "days",
+    ylab = "weight", col = 1, type = "l")
```

and observe that in both groups the variability of the weights seems to increase over time with a severe weight loss on the final measurement in group B.

Following Pourahmadi (1999), Pan and Mackenzie (2003), Pan and MacKenzie (2006), Pan and MacKenzie (2007) and Zhang *et al.* (2015), we re-analyzed group A data by using a saturated mean model with the common measurement time rescaled to $t = 1, 2, \ldots, 10, 10.5$. The Bayesian information criterion (BIC), which is closely related to the Akaike information criterion (AIC) and introduces a larger penalty term for the number of parameters in the

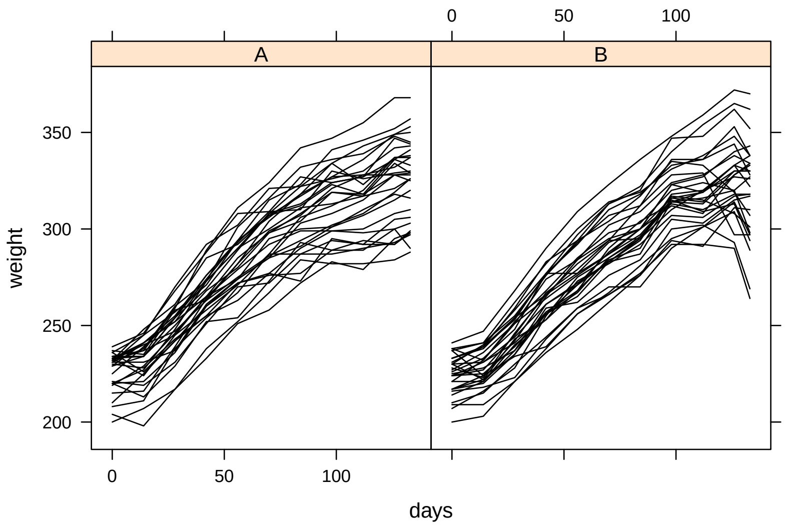

Figure 1: Subject-specific weight against time for groups A and B.

model than the AIC aiming to solve the problem of over-fitting, is used as the criterion to select the optimal model

$$\mathrm{BIC}(p, d, q) = -2\hat{l}_{\max}/n + (p + d + q + 3)\log(n)/n, \tag{29}$$

where p, d and q are respectively the orders of three polynomials and $\hat{l}_{\max}$ is the value of the maximum log-likelihood function for the given order. By default, the value of the likelihood does not include the constant term as defined in Equation 3 but it can be switched to the full likelihood containing the constant term easily by explicitly specifying `control = jmcmControl(ignore.const.term = FALSE)` in the `jmcm` function.

The basic use of `jmcm` is to indicate the model formula, data, choice of poly(p, d, q) and the covariance structure model. For example, a joint mean-covariance model based on the modified Cholesky decomposition (MCD) is estimated using:

```
R> cattleA <- subset(cattle, group == "A")
R> fit1 <- jmcm(weight | id | I(day / 14 + 1) ~ 1 | 1, data = cattleA,
+    triple = c(8, 3, 4), cov.method = "mcd")
R> fit1

Joint mean-covariance model based on MCD ['jmcmMod']
Formula: weight | id | I(day/14 + 1) ~ 1 | 1
Poly: c(8, 3, 4)
Data: cattleA

logLik: -771.0007
BIC: 53.4408

Mean Parameters:
[1]   1.832e+02   1.244e+02  -1.403e+02   7.881e+01  -2.362e+01   4.071e+00
[7]  -4.052e-01   2.162e-02  -4.787e-04
```

```
Innovation Variance Parameters:
[1]  5.366409 -0.878890  0.132427 -0.006371
Autoregressive Parameters:
[1]  0.297055  0.619888 -0.396189  0.069150 -0.003696
```

The R package **Formula** of Zeileis and Croissant (2010) is used to extract information from a two-sided linear formula object which is used to describe both longitudinal data and covariates of the model, with the response, subject id and observation time point on the left of a "`~`" operator separated by vertical bars ("`|`") and covariates for the mean model and innovation variance, also separated by a "`|`" operator, on the right. Here both covariates for mean model and innovation variance are marked as 1, and only time is used to construct design matrices. Optimal model selection involves identifying the best integer triple poly(p, d, q), specified by option `triple`, representing the degrees of three polynomial functions for the mean structure, log innovation variance and autoregressive coefficients respectively. To make the model fitting comparable with the results reported in the literature, in this paper we focus on the fitting using polynomials in time. The use of other covariates is also possible and will be demonstrated later. By default, the `jmcm` function uses the profile likelihood (i.e., estimating parameters one by one with other parameters fixed in each iteration) for having a better estimating result. Alternatively, the non-profile method can be applied by specifying `control = jmcmControl(profile = FALSE)`. When the estimation of the model has finished, an object of the S4 class '`jmcmMod`' is returned by the function and it automatically displays the basic information by calling the corresponding `print` method. The `getJMCM` function can be used to extract various objects (e.g., estimation of mean vector and covariance matrix) from a fitted joint mean-covariance model. In this example, the global optimal triple poly(8, 3, 4) reported in Pan and Mackenzie (2003) is modeled and produced a better result with $\hat{l}_{\max} = -771.0007$ and BIC $= 53.4408$.

Since it is a balanced longitudinal dataset, we produced the sample regressograms and fitted curves for the cattle data using the following function:

```
R> regressogram(fit1, time = 1:11)
```

By examining the log innovation variance versus time in Figure 2, it is clear that the curvature pattern is well captured by the fitted polynomial function curve. Figure 2 also indicates a good fit for autoregressive coefficients by examining the autoregressive coefficient versus time lag between measurements and the fitted curve.

The same triple poly(8, 3, 4), representing the degrees of three polynomial functions for the mean structure, the log innovation variance and moving average coefficients respectively, is used in joint mean-covariance model fitting based on ACD for the cattle data. The covariance structure option should be specified as `cov.method = "acd"`. By comparing this to the maximized log-likelihood and the BIC value for MCD modeling, we clearly see that the ACD method produces a larger log-likelihood $\hat{l}_{\max} = -747.6994$ and a smaller BIC value of 51.8873.

```
R> fit2 <- jmcm(weight | id | I(day / 14 + 1) ~ 1 | 1, data = cattleA,
+    triple = c(8, 3, 4), cov.method = "acd")
R> fit2

Joint mean-covariance model based on ACD ['jmcmMod']
Formula: weight | id | I(day/14 + 1) ~ 1 | 1
```

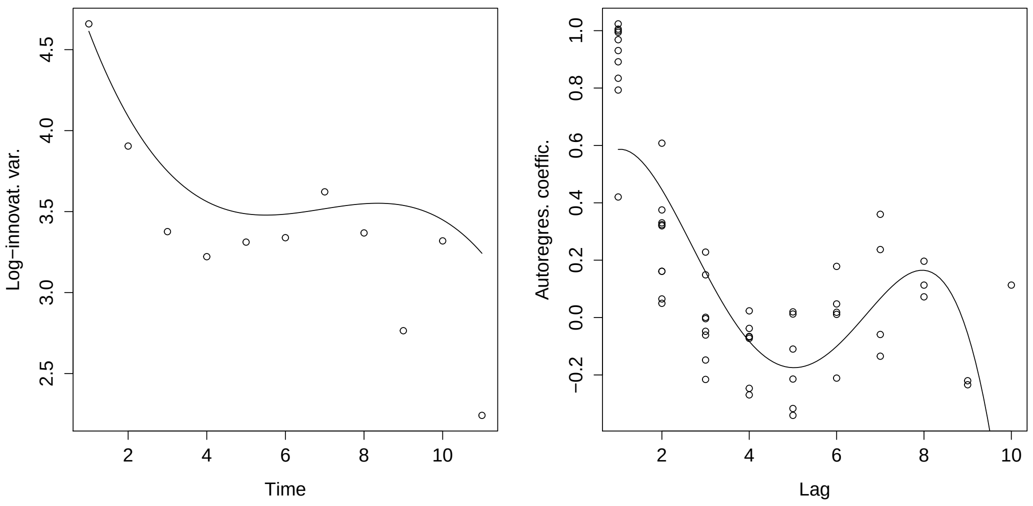

Figure 2: Group A analysis of Kenward's cattle data. Sample regressograms and MCD model fits based on the triple poly(8, 3, 4) for log innovation variances (left) and autoregressive coefficients (right).

```
Poly: c(8, 3, 4)
Data: cattleA

logLik: -747.6994
BIC: 51.8873

Mean Parameters:
[1]   1.784e+02   1.360e+02  -1.511e+02   8.400e+01  -2.503e+01   4.299e+00
[7]  -4.267e-01   2.272e-02  -5.020e-04
Innovation Variance Parameters:
[1]   4.959622  -0.625990   0.084538  -0.003963
Moving Average Parameters:
[1]   0.5216849   0.4795778  -0.1049448   0.0105871  -0.0003623
```

Regressograms for ACD can be produced by the same command:

```
R> regressogram(fit2, time = 1:11)
```

By examining the log innovation variance versus time (left) and the moving average coefficient versus time lag (right) in Figure 3, a similar conclusion can be drawn that the proposed polynomial model captures well the trend in the sample regressogram.

When the joint mean-covariance model approach based on HPC is applied to the cattle data, the covariance structure option should be specified as `cov.method = "hpc"`. The same two-sided linear formula object is used to describe both longitudinal data and covariates of the model, but on the right of the "`~`" operator, the covariates for the mean model and variance are specified on the right side of the operator instead of the mean model and innovation variance in MCD and ACD. The integer triple poly(p, d, q) is specified by option `triple`, representing the degrees of three polynomial functions for the mean structure, log variance

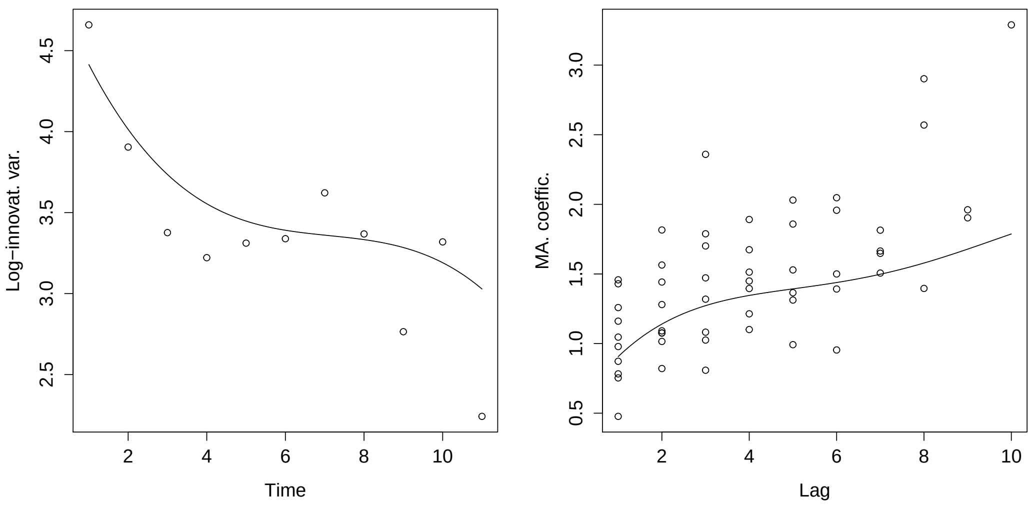

Figure 3: Group A analysis of Kenward's cattle data. Sample regressograms and ACD model fits based on the triple poly(8, 3, 4) for log innovation variances (left) and moving average coefficients (right).

and angles respectively. More specifically, the optimal triple poly(8, 2, 2) of the HPC model fitting reported in Zhang *et al.* (2015) can be reproduced using the following command:

```
R> fit3 <- jmcm(weight | id | I(day / 14 + 1) ~ 1 | 1, data = cattleA,
+    triple = c(8, 2, 2), cov.method = "hpc")
R> fit3

Joint mean-covariance model based on HPC ['jmcmMod']
Formula: weight | id | I(day/14 + 1) ~ 1 | 1
Poly: c(8, 2, 2)
Data: cattleA

logLik: -746.9001
BIC: 51.4939

Mean Parameters:
[1]   1.781e+02   1.373e+02  -1.528e+02   8.500e+01  -2.535e+01   4.359e+00
[7]  -4.330e-01   2.307e-02  -5.101e-04
Variance Parameters:
[1]   4.0263   0.3148  -0.0113
Angle Parameters:
[1]   0.729414   0.092111  -0.004424
```

A slightly better result with $\hat{l}_{\max} = -746.9001$ and BIC $= 51.4939$ is produced compared to the reported $\hat{l}_{\max} = -755.00$ and BIC $= 52.03$. Similarly, model fitting can be checked by plotting the two regressograms using:

```
R> regressogram(fit3, time = 1:11)
```

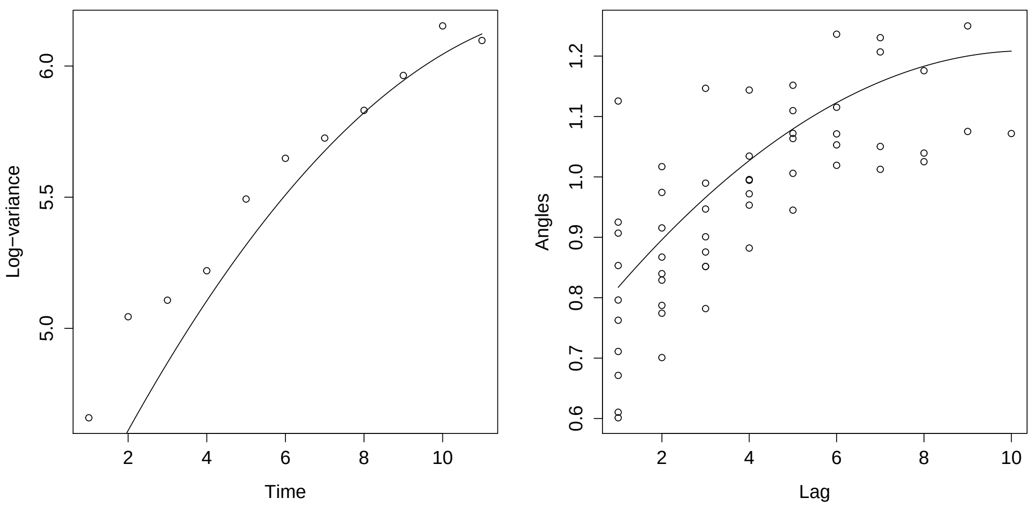

Figure 4: Group A analysis of Kenward's cattle data. Sample regressograms and HPC model fits based on the triple poly(8, 2, 2) for log variances (left) and angles (right).

We need to note that there is no general form for calculating angles. The corresponding angles ϕ_{ijk} of the empirical correlation matrix are calculated iteratively using expression:

$$\theta_{ijk} = \arccos(b_{ijk} / \prod_{l=1}^{k-1} \sin(\arccos(\theta_{ijl}))), \quad 1 \leq k < j \leq m_i, \tag{30}$$

where $\prod_1^0$ is taken as 1. By examining the log variance versus time (left) and angle versus time lag (right) in Figure 4, it is clear that the curvature patterns on the two sample regressograms are well captured by the two fitted models, indicating a good model fit.

The comparisons between MCD-, ACD- and HPC-based joint mean-covariance models on the cattle data are made in Table 1 with different choices of triple and execution time (in seconds) is measured for each fitted model. We find that the HPC-based model performs best in most cases with larger log-likelihood and smaller BIC values than compared to the MCD- and ACD-based models at the cost of a much longer execution time. From Table 1, we also find that MCD and ACD will produce quite close results in term of likelihood and BIC values while the MCD-based model is the most time efficient among the three approaches. Our tests were conducted under Windows 10 (64-bit version) on ThinkPad T410 equipped with an Intel(R) Core(TM) i5 M 480@2.67GHz with 4.00GB of RAM.

3.2. Analysis of an unbalanced longitudinal dataset

In this section, we apply the proposed joint mean-covariance modeling approach to an unbalanced *CD4+* cell dataset analyzed by Ye and Pan (2006) and Zhang *et al.* (2015). The dataset comprises a total of 2376 *CD4+* cell counts of 369 HIV-infected men covering a period of approximately eight and half years. The number of measurements m_i for each individual vary from 1 to 12 and the times are not equally spaced. The *CD4+* cell data are highly unbalanced and included in the package.

```
R> data("aids", package = "jmcm")
R> head(aids)
```

(p, d, q)	No. of	MCD			ACD			HPC		
	parms.	$l_{\max}$	BIC	Time	$l_{\max}$	BIC	Time	$l_{\max}$	BIC	Time
(8, 3, 4)	18	−771.0007	53.4408	0.53	−747.6994	51.8873	3.54	−745.2783	51.7259	6.61
(8, 2, 2)	15	−789.6174	54.3418	0.57	−750.8567	51.7577	2.78	−746.9001	51.4939	6.28
(10, 10, 10)	33	−739.1477	53.0178	4.77	−740.7159	53.1224	88.68	−739.6031	53.0482	128.10
(6, 1, 1)	11	−823.8421	56.1699	0.47	−763.5859	52.1528	1.33	−759.5982	51.8870	4.09
(3, 3, 3)	12	−825.3397	56.3831	1.05	−800.8213	54.7486	6.03	−798.1533	54.5707	16.69
(4, 4, 3)	14	−791.1545	54.3309	0.80	−760.6863	52.2996	3.21	−760.2976	52.2737	10.25
(7, 2, 2)	14	−791.7968	54.3737	0.47	−755.7579	51.9711	2.14	−751.8171	51.7084	6.17
(8, 7, 4)	22	−769.5302	53.7962	1.89	−745.1182	52.1688	4.97	−743.1843	52.0398	16.19
(9, 1, 3)	16	−794.7426	54.7968	0.38	−750.0146	51.8149	2.94	−746.7736	51.5989	9.39
(9, 4, 3)	19	−783.2143	54.3684	0.61	−746.3733	51.9123	5.41	−744.9879	51.8200	9.98
(9, 8, 5)	25	−754.3422	53.1238	2.09	−743.2145	52.3820	28.67	−741.6877	52.2802	37.91

Table 1: Kenward's cattle data. Comparison of MCD, ACD and HPC with different triples.

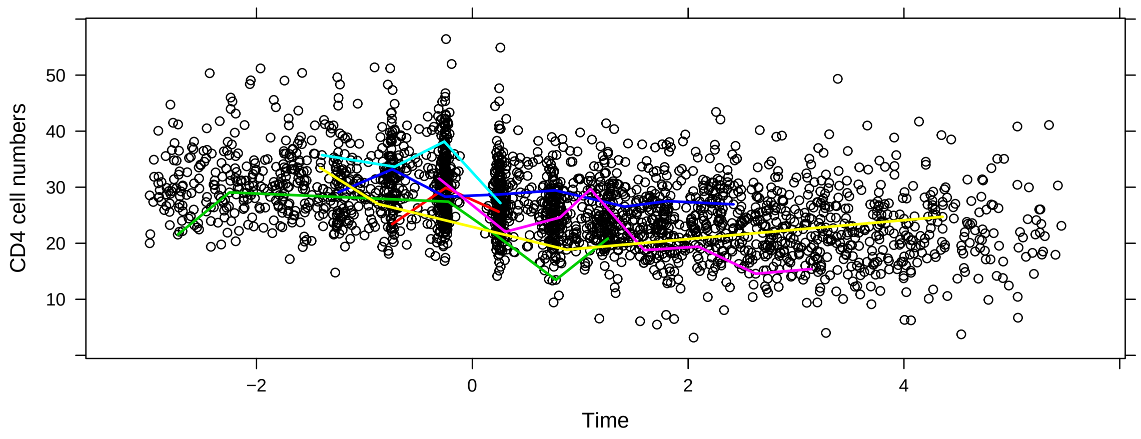

Figure 5: Scatter plot of *CD4+* cell counts against time, with the first six individual profiles superimposed.

```
       time cd4  age packs drugs sex cesd    id
1 -0.741958 548 6.57     0     0   5    8 10002
2 -0.246407 893 6.57     0     1   5    2 10002
3  0.243669 657 6.57     0     1   5   -1 10002
4 -2.729637 464 6.95     0     1   5    4 10005
5 -2.250513 845 6.95     0     1   5   -4 10005
6 -0.221766 752 6.95     0     1   5   -5 10005
```

We present in Figure 5 the scatter plot of *CD4+* cell counts against time, with the first six individual profiles superimposed:

```
R> library("lattice")
R> xyplot(sqrt(cd4) ~ time, data = aids, panel = function(x, y, ...) {
+    panel.xyplot(x, y, ...)
+    panel.lines(x[aids$id == 10002], y[aids$id == 10002], col = 2, lwd = 2)
+    panel.lines(x[aids$id == 10005], y[aids$id == 10005], col = 3, lwd = 2)
+    panel.lines(x[aids$id == 10029], y[aids$id == 10029], col = 4, lwd = 2)
+    panel.lines(x[aids$id == 10039], y[aids$id == 10039], col = 5, lwd = 2)
+    panel.lines(x[aids$id == 10048], y[aids$id == 10048], col = 6, lwd = 2)
+    panel.lines(x[aids$id == 10052], y[aids$id == 10052], col = 7, lwd = 2)
+  }, xlab = "Time", ylab = "CD4 cell numbers", col = 1)
```

and observe that the data is highly unbalanced with unclear profile patterns for each individual.

As in Zhang *et al.* (2015), square roots of the *CD4+* cell counts are used to make the response variable closer to the normal distribution. The optimal triplet poly(8, 1, 3) of the MCD method reported in Zhang *et al.* (2015) is fitted using the following command:

```
R> fit4 <- jmcm(I(sqrt(cd4)) | id | time ~ 1 | 1, data = aids,
+    triple = c(8, 1, 3), cov.method = "mcd")
R> fit4
```

```
Joint mean-covariance model based on MCD ['jmcmMod']
Formula: I(sqrt(cd4)) | id | time ~ 1 | 1
Poly: c(8, 1, 3)
Data: aids

logLik: -4979.193
BIC: 27.2278

Mean Parameters:
[1]  29.217447  -4.100596  -1.279396   1.073685   0.195578  -0.166439
[7]  -0.001842   0.009407  -0.001020
Innovation Variance Parameters:
[1]   3.2646  -0.0886
Autoregressive Parameters:
[1]   0.67990  -0.57684   0.17741  -0.01815
```

Here the *CD4+* data is again re-analyzed with MCD-based joint mean-covariance model using time as the main covariates and a value of $\hat{l}_{\max} = -4979.193$ and a smaller BIC value of 27.2278 are obtained. Note that the `jmcm` function does allow adding other covariates in the mean model and innovation variance model. For instance, the linear formula part of the `jmcm` function in this example can be replaced by `I(sqrt(cd4)) | id | time ~ age | age + packs`, which in turn generates the new vectors of covariates for the mean and innovation variance with the following form

$$x_{ij} = (1, t_{ij}, t_{ij}^2, \ldots, t_{ij}^p, age)^\top,$$
$$z_{ij} = (1, t_{ij}, t_{ij}^2, \ldots, t_{ij}^d, age, packs)^\top.$$

The joint mean-covariance model based on ACD and HPC approaches can also be fitted with other covariates in a similar way, and currently the fitted models can be compared with other model fittings using the log-likelihood and BIC values.

Since *CD4+* cell data are unbalanced, the sample covariance matrix cannot be obtained and using instruction `regressogram()` with the model fitting result will simply lead to an error message. Instead we produced the fitted curves and the 95% confidence intervals based on bootstrapping using the following function:

```
R> bootcurve(fit4, nboot = 1000)
```

where the number of bootstrap replications can be specified by option `nboot` and there are 1000 bootstrap samples in this example. Figure 6 shows the fitted curve of the mean, log innovation variance, autoregressive coefficient and their corresponding 95% confidence intervals. From Figure 6, we also observe the monotone-decreasing relationship of the fitted log innovation variance with the time, and a curvature pattern of the fitted autoregressive coefficient with the time lag.

The same triple poly(8, 1, 3) is used in joint mean-covariance model fitting based on ACD for the aids data, and we clearly see that the ACD method produces a larger likelihood $\hat{l}_{\max} = -4927.492$ and a smaller BIC value of 26.9476.

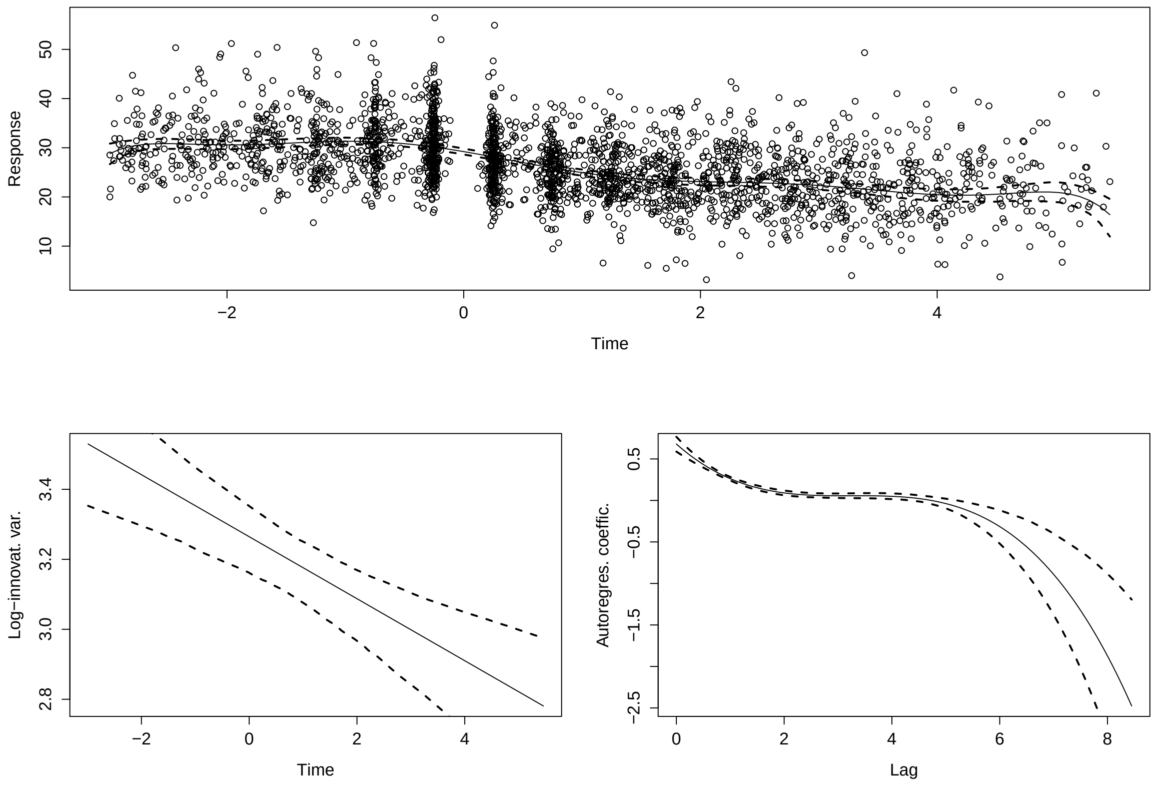

Figure 6: *CD4+* cell data. MCD model fits based on the triple poly(8, 1, 3). Fitted curves of the mean against time (top), the log innovation variance against time (left) and the autoregressive coefficient against lag (right): - - - - - -, 95% confidence intervals.

```
R> fit5 <- jmcm(I(sqrt(cd4)) | id | time ~ 1 | 1, data = aids,
+    triple = c(8, 1, 3), cov.method = "acd")
R> fit5

Joint mean-covariance model based on ACD ['jmcmMod']
Formula: I(sqrt(cd4)) | id | time ~ 1 | 1
Poly: c(8, 1, 3)
Data: aids

logLik: -4927.492
BIC: 26.9476

Mean Parameters:
[1]  29.0395978  -4.0577291  -1.0767686   0.9747427   0.1479554  -0.1416514
[7]  -0.0002373   0.0076762  -0.0008416
Innovation Variance Parameters:
[1]   3.2441  -0.1163
Moving Average Parameters:
[1]   0.580633  -0.151159   0.056772  -0.006433
```

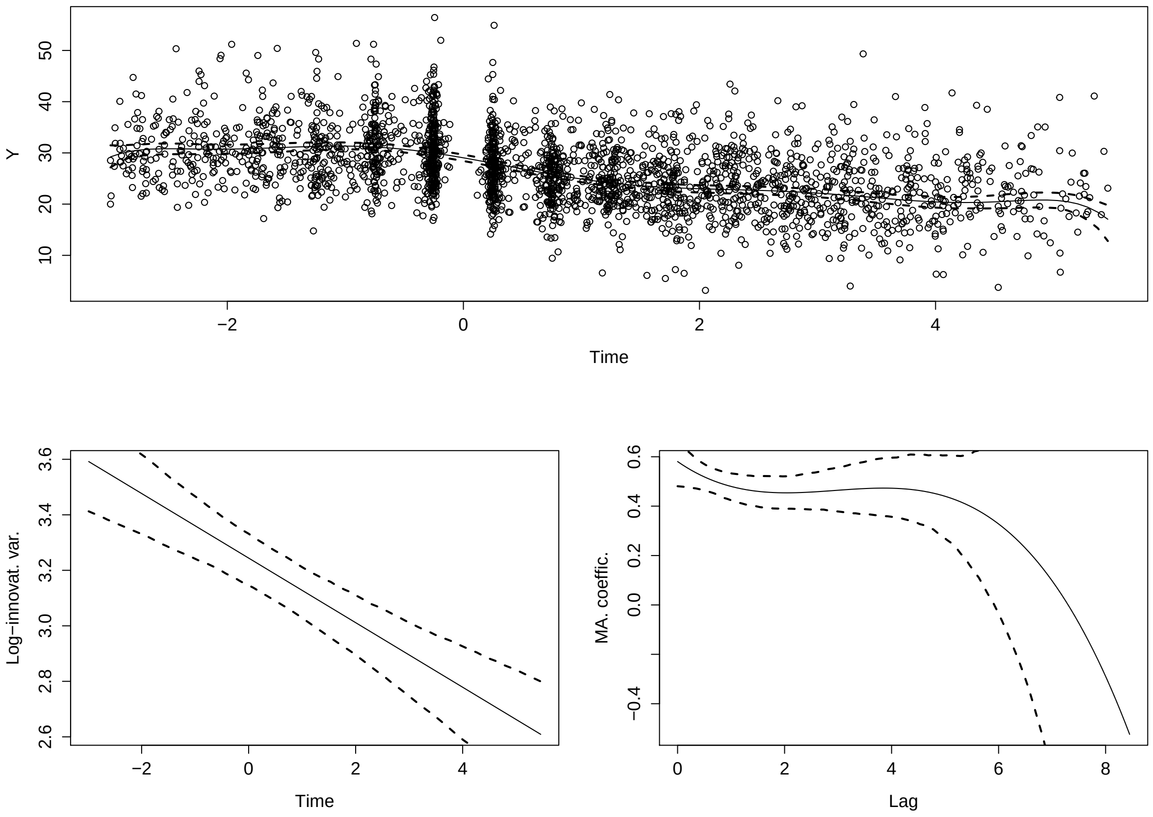

Figure 7: *CD4+* cell data. ACD model fits based on the triple poly(8, 1, 3). Fitted curves of the mean against time (top), the log innovation variance against time (left) and the moving average coefficient against lag (right): - - - - - -, 95% confidence intervals.

Fitted curves for ACD can be produced by the same command:

```
R> bootcurve(fit5, nboot = 1000)
```

From Figure 7, again we observe the monotone-decreasing relationship of the fitted log innovation variance with the time, and a curvature pattern of the fitted moving average coefficient with the time lag.

When the optimal triplet poly(8, 1, 1) of the HPC approach reported in Zhang *et al.* (2015) is fitted, the optimal BIC value turns out to be 26.7268, and $\hat{l}_{\max} = -4892.68$, producing the best model among the three proposed covariance and correlation structure modeling methods.

```
R> fit6 <- jmcm(I(sqrt(cd4)) | id | time ~ 1 | 1, data = aids,
+    triple = c(8, 1, 1), cov.method = "hpc")
R> fit6

Joint mean-covariance model based on HPC ['jmcmMod']
Formula: I(sqrt(cd4)) | id | time ~ 1 | 1
Poly: c(8, 1, 1)
Data: aids
```

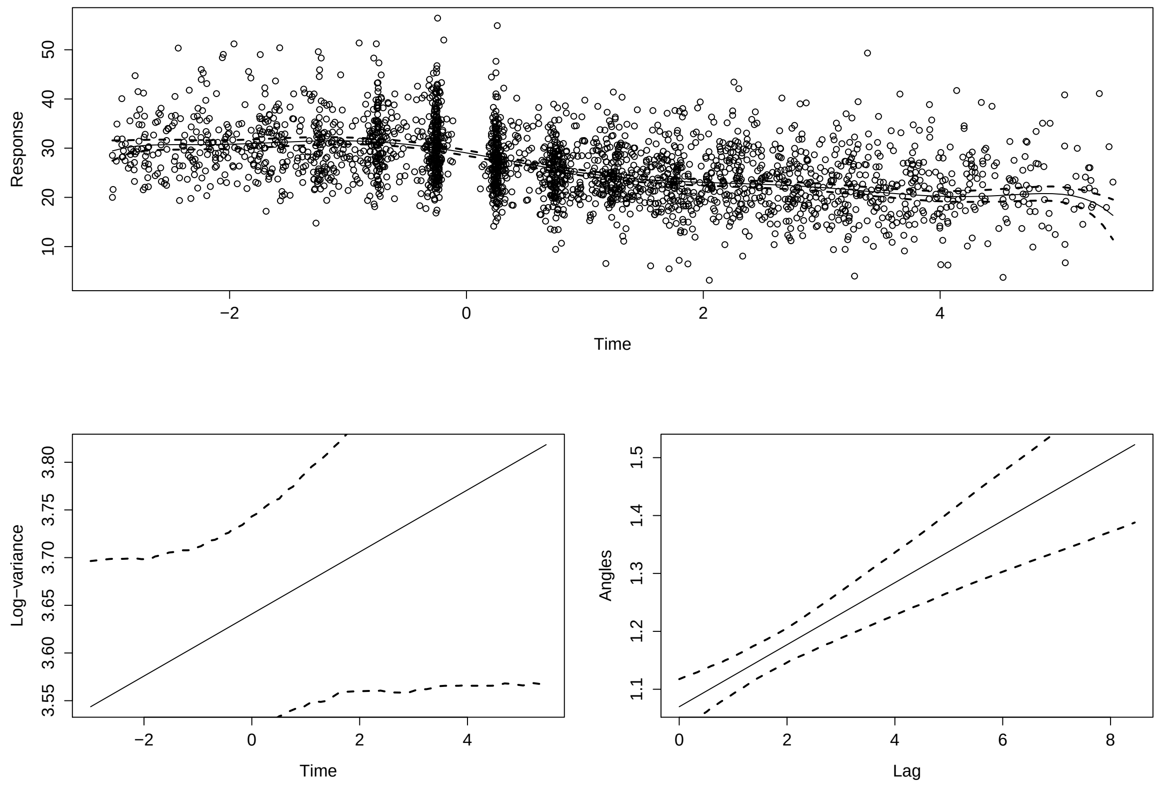

Figure 8: *CD4+* cell data. HPC model fits based on the triple poly(8, 1, 1). Fitted curves of the mean against time (top), the log variance against time (left) and the angle against lag (right): - - - - - -, 95% confidence intervals.

```
logLik: -4892.68
BIC: 26.7268

Mean Parameters:
[1]  29.0352214  -4.1553878  -0.9452119   0.9969254   0.1066325  -0.1394301
[7]   0.0026802   0.0072015  -0.0008294
Variance Parameters:
[1]  3.64089  0.03252
Angle Parameters:
[1]  1.06980  0.05357
```

Similarly, model fitting can be checked by plotting the fitted curves using:

```
R> bootcurve(fit6, nboot = 1000)
```

and from Figure 8, we observe the monotone-increasing relationship of the fitted log variance with the time, and the fitted angles with the time lag.

We also compared the MCD-, ACD- and HPC-based joint mean-covariance models on the *CD4+* cell data in Table 2. We find that the HPC-based model proves to be a better fitting

(p, d, q)	No. of	MCD			ACD			HPC		
	parms.	$l_{\max}$	BIC	Time	$l_{\max}$	BIC	Time	$l_{\max}$	BIC	Time
(8, 1, 1)	13	−5008.753	27.3560	2.11	−4928.924	26.9233	37.28	−4892.680	26.7268	134.64
(8, 1, 3)	15	−4979.193	27.2278	2.80	−4927.492	26.9476	39.72	−4890.396	26.7465	112.79
(6, 1, 1)	11	−5018.470	27.3766	1.88	−4937.227	26.9362	25.06	−4902.175	26.7463	79.32
(3, 3, 3)	12	−5006.176	27.3260	3.22	−4951.234	27.0282	40.57	−4919.522	26.8563	120.88
(4, 4, 3)	14	−4995.509	27.3002	3.36	−4934.265	26.9682	56.34	−4902.100	26.7939	177.25
(8, 3, 3)	17	−4974.683	27.2354	3.31	−4919.700	26.9374	58.68	−4886.337	26.7565	155.85
(8, 7, 4)	22	−4971.712	27.2994	7.16	−4914.223	26.9878	270.25	−4881.736	26.8117	763.83
(9, 1, 3)	16	−4974.104	27.2162	3.00	−4918.684	26.9158	63.18	−4881.266	26.7130	120.09
(9, 4, 3)	19	−4970.209	27.2432	5.06	−4909.363	26.9134	66.25	−4875.877	26.7319	212.51
(9, 8, 5)	25	−4962.655	27.2983	7.70	−4901.841	26.9687	221.10	−4871.577	26.8047	662.64

Table 2: *CD4*+ cell data. Comparison of MCD, ACD and HPC with different triples.

model in most cases with a larger log-likelihood values and smaller BIC values compared to the MCD- and ACD-based models at the cost of a much longer execution time. The ACD-based model slightly outperforms the MCD approach, and the MCD-based model again provides the most time-efficient model fitting.

4. Conclusion

In this article, we illustrated the capabilities of package **jmcm** for the joint mean-covariance modeling of both balanced and unbalanced longitudinal data using three popular covariance and correlation structure modeling approaches. In particular, we provide: functions for estimation of MCD-, ACD- and HPC-based joint mean-covariance models, devices for displaying regressograms and fitted model curves. By using these models, the estimated covariance and correlation are guaranteed to be positive (semi-)definite and the estimation of a high-dimensional covariance and correlation matrix is reduced to solving a series of regression problems. The likelihood-based estimation procedure permits extensions such as regularization-based model selection, so that the package can be compared with other likelihood-based R packages.

However, the package is currently limited to handle longitudinal data with a multivariate Gaussian distribution. It is worthwhile to develop methods further that are robust with non-normally distributed data by introducing the Cholesky-based covariance structure modeling methods to the GEE model and/or the Gaussian copula model. We plan to update the package **jmcm** on a regular basis with new statistical procedures available for the joint mean-covariance modeling approach.

Acknowledgments

We would like to thank the Editors and two anonymous reviewers for their constructive comments and helpful suggestions.

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.

Carey V, Lumley T, Ripley B (2015). ***gee**: Generalized Estimation Equation Solver.* R package version 4.13-19, URL http://CRAN.R-project.org/package=gee.

Chen Z, Dunson DB (2003). "Random Effects Selection in Linear Mixed Models." *Biometrics*, **59**(4), 762–769. doi:10.1111/j.0006-341x.2003.00089.x.

Dellaportas P, Pourahmadi M (2012). "Cholesky-GARCH Models with Applications to Finance." *Statistics and Computing*, **22**(4), 849–855. doi:10.1007/s11222-011-9251-2.

Diggle PJ, Verbyla AP (1998). "Nonparametric Estimation of Covariance Structure in Longitudinal Data." *Biometrics*, **52**(2), 401–415. doi:10.2307/3109751.

Eddelbuettel D (2013). *Seamless R and C++ Integration with* ***Rcpp***. Springer-Verlag, New York.

Eddelbuettel D, François R (2011). "**Rcpp**: Seamless R and C++ Integration." *Journal of Statistical Software*, **40**(8), 1–18. doi:10.18637/jss.v040.i08.

Eddelbuettel D, Sanderson C (2014). "**RcppArmadillo**: Accelerating R with High-Performance C++ Linear Algebra." *Computational Statistics & Data Analysis*, **71**, 1054–1063. doi:10.1016/j.csda.2013.02.005.

Fan J, Liao Y, Liu H (2016). "An Overview of the Estimation of Large Covariance and Precision Matrices." *The Econometrics Journal*, **19**(1), C1–C32. doi:10.1111/ectj.12061.

Halekoh U, Højsgaard S, 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.

Kenward MG (1987). "A Method for Comparing Profiles of Repeated Measurements." *Journal of the Royal Statistical Society C*, **36**(3), 296–308. doi:10.2307/2347788.

Liang KY, Zeger SL (1986). "Longitudinal Data Analysis Using Generalized Linear Models." *Biometrika*, **73**(1), 13–22. doi:10.2307/2336267.

Maadooliat M, Pourahmadi M, Huang JZ (2013). "Robust Estimation of the Correlation Matrix of Longitudinal Data." *Statistics and Computing*, **23**(1), 17–28. doi:10.1007/s11222-011-9284-6.

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.

Pan J, Mackenzie G (2003). "On Modelling Mean-Covariance Structures in Longitudinal Studies." *Biometrika*, **90**(1), 239–244.

Pan J, MacKenzie G (2006). "Regression Models for Covariance Structures in Longitudinal Studies." *Statistical Modelling*, **6**(1), 43–57. doi:10.1191/1471082x06st105oa.

Pan J, MacKenzie G (2007). "Modelling Conditional Covariance in the Linear Mixed Model." *Statistical Modelling*, **7**(1), 49–71. doi:10.1177/1471082x0600700104.

Pan J, Pan Y (2017). ***jmcm****: Joint Mean-Covariance Models using* ***Armadillo*** *and S4*. R package version 0.1.8.0, URL https://CRAN.R-project.org/package=jmcm.

Pedeli X, Fokianos K, Pourahmadi M (2015). "Two Cholesky-Log-GARCH Models for Multivariate Volatilities." *Statistical Modelling*, **15**(3), 233–255. doi:10.1177/1471082x14551246.

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 http://CRAN.R-project.org/package=nlme.

Pinheiro JC, Bates DM (1996). "Unconstrained Parametrizations for Variance-Covariance Matrices." *Statistics and Computing*, **6**(3), 289–296. doi:10.1007/bf00140873.

Pourahmadi M (1999). "Joint Mean-Covariance Models with Applications to Longitudinal Data: Unconstrained Parameterisation." *Biometrika*, **86**(3), 677–690.

Pourahmadi M (2007). "Cholesky Decompositions and Estimation of a Covariance Matrix: Orthogonality of Variance-Correlation Parameters." *Biometrika*, **94**(4), 1006–1013.

Pourahmadi M (2013). *High-Dimensional Covariance Estimation: With High-Dimensional Data*. John Wiley & Sons. doi:10.1002/9781118573617.

Press WH, Teukolsky SA, Vetterling WT, Flannery BP (2007). *Numerical Recipes: The Art of Scientific Computing, Third Edition (C++)*, volume 994. Cambridge University Press.

Rapisarda F, Brigo D, Mercurio F (2007). "Parameterizing Correlations: A Geometric Interpretation." *IMA Journal of Management Mathematics*, **18**(1), 55–73.

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/.

SAS Institute Inc (2013). *SAS/STAT Software, Version 9.4*. SAS Institute Inc., Cary. URL http://www.sas.com/.

SPSS Inc (2015). *IBM SPSS Statistics 23*. SPSS Inc., Chicago. URL http://www.spss.com/.

The MathWorks Inc (2015). *MATLAB – The Language of Technical Computing, Version R2015a*. Natick. URL http://www.mathworks.com/products/matlab/.

Ye H, Pan J (2006). "Modelling of Covariance Structures in Generalised Estimating Equations for Longitudinal Data." *Biometrika*, **93**(4), 927–941.

Zeileis A, Croissant Y (2010). "Extended Model Formulas in R: Multiple Parts and Multiple Responses." *Journal of Statistical Software*, **34**(1), 1–13. doi:10.18637/jss.v034.i01.

Zhang W, Leng C (2012). "A Moving Average Cholesky Factor Model in Covariance Modelling for Longitudinal Data." *Biometrika*, **99**(1), 141–150.

Zhang W, Leng C, Tang CY (2015). "A Joint Modelling Approach for Longitudinal Studies." *Journal of the Royal Statistical Society B*, **77**(1), 219–238. doi:10.1111/rssb.12065.

Affiliation:

Jianxin Pan
School of Mathematics
The University of Manchester
Oxford Road
Manchester, M13 9PL, United Kingdom
Telephone: +44/161/275-5864
Fax: +44/161/275-5819
E-mail: `jianxin.pan@manchester.ac.uk`
URL: `http://www.manchester.ac.uk/research/jianxin.pan/`

Journal of Statistical Software `http://www.jstatsoft.org/`
published by the Foundation for Open Access Statistics `http://www.foastat.org/`

December 2017, Volume 82, Issue 9 *Submitted:* 2016-02-11
`doi:10.18637/jss.v082.i09` *Accepted:* 2016-11-18