PGEE: An R Package for Analysis of Longitudinal Data with High-Dimensional Covariates

by Gul Inan and Lan Wang

Abstract We introduce an R package **PGEE** that implements the penalized generalized estimating equations (GEE) procedure proposed by Wang et al. (2012) to analyze longitudinal data with a large number of covariates. The **PGEE** package includes three main functions: `CVfit`, `PGEE`, and `MGEE`. The `CVfit` function computes the cross-validated tuning parameter for penalized generalized estimating equations. The function `PGEE` performs simultaneous estimation and variable selection for longitudinal data with high-dimensional covariates; whereas the function `MGEE` fits unpenalized GEE to the data for comparison. The R package **PGEE** is illustrated using a yeast cell-cycle gene expression data set.

Introduction

Longitudinal data arises from repeated measurements on the same subjects over time. A popular approach to analyzing longitudinal data is generalized estimating equations (GEE), which were proposed by Liang and Zeger (1986) and Zeger and Liang (1986). The GEE approach fits a marginal regression model to the longitudinal data. Instead of specifying the full joint likelihood, it only requires to specify the first two marginal moments. This is particularly attractive when the responses are discrete as specifying a joint distribution for multivariate discrete distribution is known to be challenging. Furthermore, although the GEE procedure relies on a working correlation model, it produces a consistent and asymptotically normal estimator even if the working correlation structure is misspecified. If the specified working correlation structure is close to the true correlation structure, further efficiency gain can be expected. Some commonly used working correlation structures include the exchangeable (Exch), first-order autoregressive (Ar(1)), stationary-1-dependent (MV_1) and so on. The generalized estimating equations are now implemented in two nice R packages: the **gee** package (Carey, 2015) and the **geepack** package (Halekoh et al., 2006).

With the advent of technology in data-collection, longitudinal data with a large number of covariates, in other words, high-dimensional longitudinal data, have now been commonly observed in fields such as health and genomic studies, economics and behavioral sciences. Including redundant covariates in model results in loss of accuracy in both estimation and inference. In the modern "large n, diverging p" framework, Wang (2011) studied the consistency and asymptotic normality of GEE regression estimators and verified the validity of the sandwich variance formula of GEE estimators and the large-sample Wald test under regularity conditions. Wang et al. (2012) further proposed penalized GEE for simultaneous variable selection and estimation for the cases where the number of covariates in the model is large. Similarly as GEE, the penalized GEE procedure only requires to specify the first two marginal moments and a working correlation matrix and assumes that missing data is valid only under missing completely at random, which means that missingness is independent of both observed and unobserved data. It leads to consistent variable selection performance even if the working correlation structure is misspecified, that is, with probably approaching one, the true model is selected if it is one of the candidate models. Recently, there has been growing interest in high-dimensional longitudinal data analysis, see for example Lian et al. (2014) and Wang et al. (2014).

In this paper, we present the R package **PGEE** (Inan et al., 2017) which implements the penalized generalized estimating equations procedure in Wang et al. (2012) to analyze the longitudinal data with high-dimensional covariates. The package **PGEE** is available on CRAN at `https://cran.r-project.org/web/packages/PGEE`. The rest of the paper is organized as follows. Section 2.2 provides a brief overview for both GEE and PGEE. Section 2.3 describes the main features of the functions in the **PGEE** package. Section 2.4 illustrates the use of **PGEE** via a yeast cell-cycle gene expression data set. Section 2.5 concludes the paper.

An overview for penalized generalized estimating equations

Data structure

Consider a longitudinal study where for the ith ($i = 1, 2, \ldots, N$) subject at time t ($t = 1, 2, \ldots, n_i$) we observe a response variable Y_{it} and a $p \times 1$ vector of covariates $\mathbf{X}_{it}$. Let $\mathbf{Y}_i = \left(Y_{i1}, \ldots, Y_{in_i}\right)^T$ and

$\mathbf{X}_i = \left(\mathbf{X}_{i1}, \ldots, \ldots, \mathbf{X}_{in_i}\right)^T$ denote $n_i \times 1$ vector of responses and $n_i \times p$ matrix of covariates for the ith subject, respectively. The observations obtained from the same subject are correlated whereas those obtained from different subjects are assumed to be independent.

Background on generalized estimating equations

Liang and Zeger (1986) assume that the first two marginal moments of Y_{ij} are $\mu_{it}(\boldsymbol{\beta}) := \mathbb{E}(Y_{it}|\mathbf{X}_{it}) = \mu(\theta_{it})$ and $\sigma^2_{it}(\boldsymbol{\beta}) := \text{Var}(Y_{it}|\mathbf{X}_{it}) = \dot{\mu}(\theta_{it})$, where $\theta_{it} = \mathbf{X}_{it}^T\boldsymbol{\beta}$, and $\boldsymbol{\beta} = (\beta_1, \ldots, \beta_p)^T$ is $p \times 1$ vector of unknown regression coefficients of interest, $i = 1, 2, \ldots, N, t = 1, 2, \ldots, n_i$. These moment assumptions would follow when the marginal response variable has a canonical exponential family distribution with scaling parameter one.

Let $\mathbf{V}_i = \text{Var}(\mathbf{Y}_i|\mathbf{X}_{it})$ be the $n_i \times n_i$ covariance matrix of the ith subject, $i = 1, \ldots, N$. In practice, Liang and Zeger (1986) suggest to estimate $\mathbf{V}_i$ via a working correlation structure. Specifically, let

$$\mathbf{V}_i = \mathbf{A}_i^{1/2} R_{n_i}(\boldsymbol{\alpha}) \mathbf{A}_i^{1/2}, \tag{1}$$

where $\mathbf{A}_i$ is an $n_i \times n_i$ diagonal matrix with the marginal variance of responses on the diagonals, and $R_{n_i}(\boldsymbol{\alpha})$ represents an $n_i \times n_i$ working correlation matrix indexed by a vector of parameters $\boldsymbol{\alpha}$. From now on, we will use $R(\boldsymbol{\alpha})$ rather than $R_{n_i}(\boldsymbol{\alpha})$ for simplicity. The popular choices for $R(\boldsymbol{\alpha})$ may be independence (Indep), exchangeable (Exch), first-order autoregressive (Ar(1)), stationary-m-dependent (MV_m), and non-stationary-m-dependent (NMV_m) (m denotes the lag order), and unstructured (UN) working correlation structure. A good review of the commonly used working correlation matrices is given in Horton and Lipsitz (1999).

Let $\mathbf{A}_i(\boldsymbol{\beta}) = \text{diag}(\sigma^2_{i1}(\boldsymbol{\beta}), \ldots, \sigma^2_{in_i}(\boldsymbol{\beta}))$ and $\boldsymbol{\mu}_i(\boldsymbol{\beta}) = (\mu_{i1}(\boldsymbol{\beta}), \ldots, \mu_{in_i}(\boldsymbol{\beta}))^T$. Liang and Zeger (1986) proposed to estimate the regression parameters $\boldsymbol{\beta}$ by solving the following set of estimating equations

$$\mathbf{S}(\boldsymbol{\beta}) = \sum_{i=1}^{N} \mathbf{X}_i^T \mathbf{A}_i^{1/2} \hat{R}^{-1}(\boldsymbol{\alpha}) \mathbf{A}_i^{-1/2} (\mathbf{Y}_i - \boldsymbol{\mu}_i) = 0, \tag{2}$$

where $\hat{R}^{-1}(\boldsymbol{\alpha})$ denotes the estimated working correlation matrix. The estimating equations can be solved using a modified Fisher scoring algorithm. Within the iterative Fisher scoring algorithm, the parameter α in $R(\boldsymbol{\alpha})$ can be estimated by residual-based moment method, see Hardin and Hilbe (2003). Liang and Zeger (1986) showed that the resulted estimator is consistent even if R is misspecified.

Penalized generalized estimating equations for longitudinal data with high-dimensional covariates

With high-dimensional covariates, it is often reasonable to assume that many of these covariates are not relevant for modeling the marginal mean of the response variable, in other words, the regression coefficients vector $\boldsymbol{\beta}$ can be assumed to be sparse in the sense that most of its components are exactly zero. Wang et al. (2012) introduced penalized generalized estimating equations (PGEE) for simultaneous estimation and variable selection in this setting. More specifically, they propose to estimate $\boldsymbol{\beta}$ by $\hat{\boldsymbol{\beta}}$, which solves the following set of penalized estimating equations

$$\mathbf{U}(\boldsymbol{\beta}) = \mathbf{S}(\boldsymbol{\beta}) - \mathbf{q}_\lambda(|\boldsymbol{\beta}|) \circ \text{sign}(\boldsymbol{\beta}) \tag{3}$$

where $\mathbf{q}_\lambda(|\boldsymbol{\beta}|) = (q_\lambda(|\beta_1|), \ldots, q_\lambda(|\beta_p|))^T$, $\text{sign}(\boldsymbol{\beta}) = (\text{sign}(\beta_1), \ldots, \text{sign}(\beta_p))^T$, and $\mathbf{q}_\lambda(|\boldsymbol{\beta}|) \circ \text{sign}(\boldsymbol{\beta})$ denotes the Hadamard product (element-wise product) of these two vectors. The penalty function $q_\lambda(|\beta_j|)$, the jth component of $\mathbf{q}_\lambda(|\boldsymbol{\beta}|)$, takes a zero value for a large value of $|\beta_j|$ and takes a large value for a small value of $|\beta_j|$. Consequently, the generalized estimating equation $S(\beta_j)$, the jth component of $\mathbf{S}(\boldsymbol{\beta})$, is not penalized if $|\beta_j|$ is large in magnitude, whereas $S(\beta_j)$ is penalized if $|\beta_j|$ is smaller than a cut-off value (greater than zero). Hence, the role of the penalty function $q_\lambda(|\beta_j|)$ is to shrink the estimates of small coefficients toward zero. The coefficients whose estimates are shrunken to zero are excluded from the final model. The cut-off value is chosen as 10^{-3} as in Cho and Qu (2013), Wang et al. (2012), and Wang et al. (2007). The penalty has a tuning parameter λ that controls the model complexity. Wang et al. (2012) studied the SCAD penalty function (Fan and Li, 2001) which is defined on $[0, +\infty]$ as

$$q_\lambda(t) = \lambda \Big\{ I(t < \lambda) + \frac{(a\lambda - t)_+}{(a-1)\lambda} I(t \geq \lambda) \Big\}, \tag{4}$$

where $\lambda \geq 0$, $a > 2$ and $b_+ = bI(b > 0)$ for a real number b. Following the recommendation of Fan and Li (2001), we use $a = 3.7$ and find it usually works well. The nonconvex SCAD penalty function alleviates the main drawback of L_1 penalty function in that it avoids over-penalizing large coefficients and hence leads to consistent variable selection under more relaxed conditions. Under regularity conditions, the estimated coefficients for redundant covariates are shrunken to exactly zero.

Motivated by Johnson et al. (2008), Wang et al. (2012) proposed to solve the penalized estimated equations in Equation (3) by combining the minorization-maximization (MM) algorithm with a Newton-Raphson (NR) algorithm. At the jth iteration,

$$\hat{\boldsymbol{\beta}}^j = \hat{\boldsymbol{\beta}}^{j-1} + \big[\mathbf{H}(\hat{\boldsymbol{\beta}}^{j-1}) + N\mathbf{E}(\hat{\boldsymbol{\beta}}^{j-1})\big]^{-1}\big[\mathbf{S}(\hat{\boldsymbol{\beta}}^{j-1}) - N\mathbf{E}(\hat{\boldsymbol{\beta}}^{j-1})\hat{\boldsymbol{\beta}}^{j-1}\big], \tag{5}$$

where

$$\begin{aligned} \mathbf{H}(\hat{\boldsymbol{\beta}}^{j-1}) &= \sum_{i=1}^{N} \mathbf{X}_i^T \mathbf{A}_i^{1/2}(\hat{\boldsymbol{\beta}}^{j-1})\hat{R}^{-1}(\boldsymbol{\alpha})\mathbf{A}_i^{1/2}(\hat{\boldsymbol{\beta}}^{j-1})\mathbf{X}_i \\ \mathbf{E}(\hat{\boldsymbol{\beta}}^{j-1}) &= \operatorname{diag}\Big\{\frac{q_\lambda(|\hat{\beta}_1|+)}{\epsilon + |\hat{\beta}_1|}, \ldots, \frac{q_\lambda(|\hat{\beta}_p|+)}{\epsilon + |\hat{\beta}_p|}\Big\}, \end{aligned} \tag{6}$$

where ϵ is a small value (e.g., 10^{-6}). Wang et al. (2012) derived the asymptotic theory of PGEE in a high-dimensional framework where the number of covariates p increases as N increases, and p can reach the same order as N. An important feature of PGEE is that even if the working correlation structure is misspecified, the consistency of model selection holds, that is, with probability approaching one, it correctly identifies the zero coefficients to be zero and the nonzero coefficients to be nonzero. They also suggested a sandwich formula to estimate the asymptotic covariance matrix of the penalized GEE estimator as follows:

$$\operatorname{Cov}(\hat{\boldsymbol{\beta}}) \approx \big[\mathbf{H}(\hat{\boldsymbol{\beta}}) + N\mathbf{E}(\hat{\boldsymbol{\beta}})\big]^{-1}\mathbf{M}(\hat{\boldsymbol{\beta}})\big[\mathbf{H}(\hat{\boldsymbol{\beta}}) + N\mathbf{E}(\hat{\boldsymbol{\beta}})\big]^{-1}, \tag{7}$$

where

$$\mathbf{M}(\hat{\boldsymbol{\beta}}) = \sum_{i=1}^{N} \mathbf{X}_i^T \mathbf{A}_i^{1/2}(\hat{\boldsymbol{\beta}})\hat{R}^{-1}\boldsymbol{\epsilon}_i\boldsymbol{\epsilon}_i^T\hat{R}^{-1}\mathbf{A}_i^{1/2}(\hat{\boldsymbol{\beta}})\mathbf{X}_i, \tag{8}$$

where $\boldsymbol{\epsilon}_i = \mathbf{A}_i^{-1/2}(\hat{\boldsymbol{\beta}})(\mathbf{Y}_i - \boldsymbol{\mu}_i)$.

The tuning parameter λ in Equation (4) can be selected using K-fold cross-validation, where K is a positive integer. The data is divided into non-overlapping K sub-samples of equal sizes. The kth sub-sample being left out as the testing data set, and the remaining data are used as the training data set, $k = 1, \ldots, K$. We use the set N_{-k} to denote the indice set of the subjects in the training data set and use $|N_{-k}|$ to denote the cardinality of N_{-k}. We fit the PGEE under working independence assumption using the training data and then evaluate the prediction error using the test data by $PE_{-k}(\lambda)$, which is defined as

$$PE_{-k}(\lambda) = \frac{1}{|N_{-k}|}\sum_{i\in N_{-k}}\frac{1}{n_i}\sum_{t=1}^{n_i}\big(Y_{it} - g(\mathbf{X}_{it}^T\hat{\boldsymbol{\beta}})\big)^2,$$

We repeat the above computation for each of the K subsamples, and the overall cross-validated prediction error is given by

$$CV(\lambda) = \frac{1}{K}\sum_{k=1}^{K} PE_{-k}(\lambda) \tag{9}$$

Given a set of λ values over a grid, we choose the value of λ that yields the smallest $CV(\lambda)$.

PGEE estimation algorithm

The algorithm for solving penalized generalized estimating equations is summarized as follows:

1. Determine a reasonable grid of values for λ.
2. Given a value of λ:
   - Assign an initial value for $\boldsymbol{\beta}$.
   - Compute $\mathbf{U}(\tilde{\boldsymbol{\beta}})$, $\mathbf{H}(\tilde{\boldsymbol{\beta}})$, and $\mathbf{E}(\tilde{\boldsymbol{\beta}})$ for current value of $\tilde{\boldsymbol{\beta}}$.

- Update the current estimate of $\boldsymbol{\beta}$ through formula in Equation (5).
- Stop the iteration sequence if a convergence criterion is satisfied such as when the L_1 norm of the difference of estimated $\boldsymbol{\beta}$ between iterations is below a threshold (e.g., 10^{-3}) and denote the estimated value at convergence as $\hat{\boldsymbol{\beta}}$.
- Compute the cross-validation value of λ via Equation (9).

3. Repeat step 2 for each value of λ over the grid and find the value of λ that gives the smallest cross-validation prediction error.
4. Find the estimated parameter $\hat{\boldsymbol{\beta}}$ corresponding to the selected λ, and compute the sandwich covariance matrix by Equation (7).

The PGEE package

The R package **PGEE** consists of three core functions: `CVfit`, `PGEE`, and `MGEE`, respectively. The main function `PGEE` fits PGEEs to the longitudinal data with high-dimensional covariates. Prior to model fitting with `PGEE`, the cross-validated tuning parameter should be computed via the function `CVfit`. The package also includes the function `MGEE` which fits the unpenalized GEEs to the data. The `PGEE` and `MGEE` functions are written by the authors. The R package **PGEE** depends on the R packages **MASS** (Ripley, 2015) and **mvtnorm** (Genz et al., 2015) only.

In this section, we introduce the input arguments of the functions `CVfit` and `PGEE`, whereas the function `MGEE` shares the same arguments with the function `PGEE` except the arguments `lambda`, `pindex`, and `eps`.

The usage and input arguments of `CVfit` function are summarized as follows:

```
CVfit(formula, id, data, family, scale.fix = TRUE, scale.value = 1, fold, lambda.vec,
pindex, eps, maxiter, tol)
```

The function `CVfit` applies the step 3 in the estimation algorithm in Section 2.2.4 via Equation (9). It uses the function `PGEE` inside such that both functions share common input arguments. The input argument `formula` is a formula expression in the form of `response predictors` as in `lm` and `glm` functions. The argument `id` is a vector for identifying subjects/clusters and the argument `data` is a data frame which stores the response and covariates in `formula` with `id` variable as in `gee` function in R package **gee**. Please note that the function `PGEE` requires the covariates to be numeric variables and does not work with factor covariates. The argument `family` is a list of functions and expressions for defining `link` and `variance` functions. While families supported in the R package **PGEE** are `binomial`, `gaussian`, `gamma`, and `poisson`, link functions supported are `identity`, `log`, `logit`, `inverse`, `probit`, and `cloglog`. The argument `scale.fix` is a logical variable. The default value is `TRUE`. On the other hand, if `scale.fix = TRUE`, `scale.value` assigns a numeric value to which the scale parameter should be fixed at. Otherwise, the default value is `1`. The arguments `fold`, `pindex`, and `eps` are the main buildings of the function `CVfit` for cross-validation. The argument `fold` is the number of folds used in cross-validation. The argument `lambda.vec` is a vector of tuning parameters used in cross-validation. The argument `pindex` is an index vector showing the parameters which are not subject to penalization. The default value is `NULL`. However, in case of a model with intercept, the intercept parameter should be never penalized. The argument `eps` is a numerical value for the ϵ used in Equation (6). The default value is 10^{-6}. The argument `maxiter` is the number of iterations that is used in the estimation algorithm. The default value is 30. The argument `tol` is the tolerance level that is used in the estimation algorithm to evaluate algorithm convergence. The default value is 10^{-3}. The function `CVfit` returns an object class of `CVfit`. Applying the function `print` to the object returned by function `CVfit` provides detailed information related to cross-validation and gives the value of λ that minimizes the cross-validated value of prediction error.

PGEEs (Wang et al., 2012) and, in turn, the function `CVfit` in R package **PGEE** accommodate the SCAD penalty. Fan and Li (2001) demonstrated that the SCAD penalty function is a popular nonconvex penalty function that satisfies three desirable properties of variable selection (e.g., sparsity, unbiasedness, and continuity) simultaneously. Recently, a number of R packages have been developed for estimation and variable selection problems in linear regression models, logistic regression models, quantile regression models, and Cox proportional hazards models for cross-sectional data with high-dimensional covariates; see the R package **ncvreg** (Breheny and Huang, 2011) for linear and logistic regression models with SCAD and MCP penalization functions, the R package **penalized** (Goeman, 2010) for generalized linear regression models and Cox proportional hazards models with L_1 and L_2 penalty functions, the R package **glmnet** (Friedman et al., 2010) for generalized linear regression models and Cox proportional hazards models with LASSO and elastic-net penalty functions, and the R package **rqPen** (Sherwood and Maidman, 2016) for quantile regression penalized quantile regression

with LASSO, SCAD, and MCP functions. However, these packages do not apply to the clustered data setting which we consider in this paper.

The usage and input arguments of PGEE function are summarized as follows:

```
PGEE(formula, id, data, na.action = NULL, family =  gaussian(link = "identity"),
corstr = "independence", Mv = NULL, beta_int = NULL, R = NULL, scale.fix = TRUE,
scale.value = 1, lambda, pindex = NULL, eps = 10^-6, maxiter = 30, tol = 10^-3,
silent = TRUE)
```

The input arguments `formula`, `data`, `id`, and `family` are the same as those in the `CVfit` function. Here, the default value for `family` is `gaussian`. The argument `na.action` is a function to remove missing values from the data, where only `na.omit` is allowed. The argument `corstr` is a character string, which specifies the type of correlation structure within the repeated measurements of a subject. The correlation structures supported in the R package **PGEE** are `"AR-1"`,`"exchangeable"`, `"fixed"`, `"independence"`, `"stat_M_dep"`, `"non_stat_M_dep"`, and `"unstructured"`. The default `corstr` type is `"independence"`. If either `"stat_M_dep"` or `"non_stat_M_dep"` is specified in `corstr`, then the argument `Mv` assigns a numeric value for `Mv`, which is one minus less than the largest number of repeated measurements of a subject has in the data. Otherwise, the default value is `NULL`. When the longitudinal data is unbalanced, the use of `"non_stat_M_dep"` and `"unstructured"` is not allowed in the argument `corstr`. If `corstr = "fixed"` is specified, then the argument `R` is a user specified square correlation matrix of dimension maximum of the number of repeated measurements of a subject has in the data. Otherwise, the default value is `NULL`. The argument `beta_int` is a user specified vector of initial values for regression parameters including the intercept. The default value is `NULL` which gets initial values through fitting a `glm` to the whole longitudinal data. The argument `lambda` is a numerical value returned by `CVfit` function. The input arguments `scale.fix`, `scale.value`, `pindex`, `eps`, `maxiter`, and `tol` are the same as those in the `CVfit` function. The argument `silent` is a logical variable; if false, the regression parameter estimates at each iteration are printed. The default value is `TRUE`.

The function `PGEE` returns an object class of `PGEE`. Applying the functions `print` and `summary` to an object returned by function `PGEE` provides detailed information related to the model fitting and summarizes the results as illustrated in the next section.

The function `MGEE` closely follows the syntax of the function `gee` in the R package **gee** except that the argument `subset` for data sub-setting and the argument `contrasts` for coding factor variables in terms of dummy variables are not used in the function `MGEE`. Furthermore, while any lag order can be assumed in the argument `corstr = "AR-M"` of the function `gee`, only first-order lag is allowed in the function `MGEE` (e.g., `corstr = "AR-1"`). On the other hand, there is much discrepancy between the arguments of the function `MGEE` and the function `geeglm` in the R package **geepack** since the latter inherits its arguments mostly from the function `glm`. As the result, arguments in **geepack** such as `weights`, `subset`, `etastart`, `mustart`, `offset`, and `waves` are not available in our function `MGEE`. Its `corstr` menu consists of `"AR-1"`,`"exchangeable"`, `"fixed"`, `"independence"`, and `"unstructured"` structures, which is less comprehensive compared to the `corstr` menu of the function `MGEE`. Lastly, in addition to the sandwich variance estimator, the function `geeglm` offers jackknife variance estimator for data sets with small number of clusters via the argument `std.err`.

Example

In this section, we demonstrate the use of the R package **PGEE** using a yeast cell-cycle gene expression data set collected in the CDC15 experiment of Spellman et al. (1998). The experiment measured genome-wide mRNA levels of 6178 yeast open reading frames (ORFs) (translates DNA sequences into its corresponding amino acid sequences which will appear in the final protein). This experiment covered two cell-cycle periods, where measurements were taken at 7-minute intervals over a 119-minutes period yielding a total of 18 time points.

As discussed in Wang et al. (2012) and Wang et al. (2007), the cell-cycle process is a regulated life process where the cell grows, replicates its DNA and prepares itself for cell-division. This process is generally divided into M/G1-G1-S-G2-M stages, where M refers to "mitosis", G1 refers "GAP 1", S refers to DNA "synthesis", and G2 refers to "GAP 2", respectively. Spellman et al. (1998) identified a total of 800 genes that showed periodic variation during the cell-cycle process. However, to better understand the phenomenon underlying cell-cycle process, it is important to identify transcription factors (TFs) that regulate the gene expression levels of cell cycle-regulated genes. Specifically, TFs are proteins that control gene regulation by determining the rate of transcription of genetic information from DNA to mRNA.

As Wang et al. (2012) and Wang et al. (2007), we used a subset of 297 cell-cycled-regularized genes and the binding probabilities of a total of 96 TFs obtained from a mixture model approach of Wang

et al. (2007) based on the ChIP data of Lee et al. (2002). We applied PGEE to identify the TFs that are significantly associated with gene expression level at M/G1, G1, S, G2, and M stages (each stage has different number of time points from the cycle). We fitted the following PGEE to each stage with three different working correlation matrices (independence, exchangeable, and Ar(1)):

$$Y_{it} = \gamma_0 + \gamma_1 time_{it} + \sum_{p=1}^{96} \beta_p x_{ip}, \tag{10}$$

where $time_{it}$ is the time variable and x_{ip}'s are standardized transcription factor values ($p = 1, 2, \ldots, 96$). When fitting PGEE, only β_p's were subject to penalization. Table 1 summarizes the number of TFs which are identified as statistically significant for each stage under three different working correlation structures.

Table 1: Number of TFs selected at each stage in the yeast cell-cycle process with PGEE

Correlation	M/G1	G1	S	G2	M
Independence	16	12	14	8	9
Exchangeable	15	13	12	8	7
Ar(1)	15	13	13	8	8

For illustration, we used the yeast data from "G1" stage. Specifically, like the R package **gee**, the package **PGEE** requires the response and covariate columns to be ordered by `id` column and within `id` column according to `time` column. In our example, the yeast data was saved under the name of `yeastG1` object. The first column was `id` column, which identifies the genes. Then, while the continuous responses were placed under the column `y`, the `time` variable and the standardized values of 96 TFs were placed subsequently. Due to space limitation, we illustrated a portion of the `yeastG1` data as follows:

```
> # load data
> data(yeastG1)
> data = yeastG1
> # get the column names
> colnames(data)[1:9]
[1] "id"    "y"     "time"  "ABF1"  "ACE2"  "ADR1"  "ARG80" "ARG81" "ARO80"
> # see some portion of yeast G1 data
> head(data,5)[1:9]
  id    y time        ABF1       ACE2       ADR1      ARG80       ARG81      ARO80
1  1 0.88    3 -0.09702788  8.3839614 -0.1435702 -0.1482691 -0.09690053 -0.1073776
2  1 0.32    4 -0.09702788  8.3839614 -0.1435702 -0.1482691 -0.09690053 -0.1073776
3  1 1.09   12 -0.09702788  8.3839614 -0.1435702 -0.1482691 -0.09690053 -0.1073776
4  1 0.73   13 -0.09702788  8.3839614 -0.1435702 -0.1482691 -0.09690053 -0.1073776
5  2 0.66    3 -0.34618104 -0.1418099 -0.1397801 -0.1476834 -0.08486203 -0.1073536
```

Prior to model fitting with the function `PGEE`, we needed to compute the cross-validated value of tuning paremeter over a grid via the function `CVfit`. This process requires a trial-error period, where one can start with a wide grid interval and then narrow it down. As described in Section 2.3, we determined the main input arguments of the function `CVfit` as follows:

```
> library(PGEE)
> # define the input arguments
> formula <- "y ~.-id"
> family <- gaussian(link = "identity")
> lambda.vec <- seq(0.01,0.2,0.01)
> # find the optimum lambda
> cv <- CVfit(formula = formula, id = id, data = data, family = family, scale.fix = TRUE,
+ scale.value = 1, fold = 4, lambda.vec = lambda.vec, pindex = c(1,2), eps = 10^-6,
+ maxiter = 30, tol = 10^-6)
> # print the results
> print(cv)
Call:
CVfit(formula = formula, id = id, data = data, family = family,
    scale.fix = TRUE, scale.value = 1, fold = 4, lambda.vec = lambda.vec,
    pindex = c(1, 2), eps = 10^-6, maxiter = 30, tol = 10^-6)
```

```
4-fold CV results:
      lambda       Cv
 [1,]   0.01 720.6857
 [2,]   0.02 482.1046
 [3,]   0.03 382.6932
 [4,]   0.04 358.1970
 [5,]   0.05 344.1034
 [6,]   0.06 338.4713
 [7,]   0.07 335.7137
 [8,]   0.08 333.5432
 [9,]   0.09 330.7405
[10,]   0.10 327.8395
[11,]   0.11 326.3243
[12,]   0.12 325.3068
[13,]   0.13 324.6835
[14,]   0.14 324.5388
[15,]   0.15 324.8667
[16,]   0.16 325.7849
[17,]   0.17 327.1245
[18,]   0.18 328.4365
[19,]   0.19 329.5468
[20,]   0.20 330.6265

Optimal tuning parameter:
Best lambda
       0.14
> # see the returned values by CVfit
> names(cv)
[1] "fold"     "lam.vect" "cv.vect"  "lam.opt"  "cv.min"   "call"
> # get the optimum lambda
> cv$lam.opt
[1] 0.14
```

After selecting the tuning parameter λ via the function `CVfit`, we apply the `PGEE` function with the working correlation matrix type `corstr = "independence"` as follows:

```
> # fit the PGEE model
> myfit1 <- PGEE(formula = formula, id = id, data = data, na.action = NULL,
+ family = family, corstr = "independence", Mv = NULL,
+ beta_int = c(rep(0,dim(data)[2]-1)), R = NULL, scale.fix = TRUE,
+ scale.value = 1, lambda = cv$lam.opt, pindex = c(1,2), eps = 10^-6,
+ maxiter = 30, tol = 10^-6, silent = TRUE)
```

For comparison, we also fit the same model with `corstr = "exchangeable"` and `corstr = "AR-1"` (see Table 1). Here, we use 0 initial values for the regression coefficients for γ_0, γ_1, and β_p's ($p = 1, 2, \ldots, 96$) by assigning 0's for `beta_int`. Alternatively, the initial estimates could be obtained via fitting a `glm` while setting `beta_int = NULL`. We specify the vector of index for unpenalized parameters as `pindex = c(1,2)` since the first two regression coefficients γ_0 and γ_1 in Equation (10) are not subject to penalization. Furthermore, the returned values of `myfit1` object (in a similar way `summary(myfit1)` object) can be found out by the `names` function:

```
> # get the values returned by myfit object
> names(myfit1)
 [1] "title"               "version"             "model"
 [4] "call"                "terms"               "nobs"
 [7] "iterations"          "coefficients"        "nas"
[10] "linear.predictors"   "fitted.values"       "residuals"
[13] "family"              "y"                   "id"
[16] "max.id"              "working.correlation" "scale"
[19] "epsilon"             "lambda.value"        "robust.variance"
[22] "naive.variance"      "xnames"              "error"
```

The returned objects by `PGEE` function are in similar format as those returned by gee function in R package. For example, while we could obtain whole model fitting results by `summary(myfit1)` object,

we could also obtain the summary of the model fitting results, i.e., estimate of regression coefficients, their corresponding naive and robust standard errors as well as z-values through `coefficients` method for `summary(myfit1)` object as follows:

```
> # see a portion of the results returned by coef(summary(myfit1))
> head(coef(summary(myfit1)),7)
                Estimate   Naive S.E.      Naive z  Robust S.E.   Robust z
(Intercept)  9.835775e-02 0.0603431824  1.629972904 4.334557e-02  2.2691533
time         9.774627e-03 0.0065644728  1.489019375 3.274155e-03  2.9853891
ABF1        -4.032513e-03 0.0095653833 -0.421573608 2.054339e-03 -1.9629245
ACE2         1.824265e-06 0.0002669801  0.006832963 1.634333e-06  1.1162135
ADR1         1.911308e-07 0.0001733864  0.001102340 7.611678e-07  0.2511020
ARG80        2.017436e-07 0.0001741572  0.001158399 8.684975e-07  0.2322903
ARG81        2.374483e-05 0.0007900111  0.030056320 2.296590e-05  1.0339165
```

Any regression estimate less than 10^{-3} in magnitude can be considered as equal to 0 (and thus not selected) in PGEE. In this sense, we obtained the variables whose regression estimates greater than 10^{-3} and their summary statistics as follows:

```
> # see the variables which have non-zero coefficients
> index1 <- which(abs(coef(summary(myfit1))[,"Estimate"]) > 10^-3)
> names(abs(coef(summary(myfit1))[index1,"Estimate"]))
 [1] "(Intercept)" "time"        "ABF1"        "FKH1"        "FKH2"        "GAT3"
 [7] "GCR2"        "MBP1"        "MSN4"        "NDD1"        "PHD1"        "RGM1"
[13] "RLM1"        "SMP1"        "SRD1"        "STB1"        "SWI4"        "SWI6"

> # see the PGEE summary statistics of these non-zero variables
> coef(summary(myfit1))[index1,]
               Estimate  Naive S.E.    Naive z Robust S.E.  Robust z
(Intercept)  0.098357752 0.060343182  1.6299729 0.043345573  2.269153
time         0.009774627 0.006564473  1.4890194 0.003274155  2.985389
ABF1        -0.004032513 0.009565383 -0.4215736 0.002054339 -1.962925
FKH1        -0.009152898 0.013746477 -0.6658359 0.004173178 -2.193268
FKH2        -0.091503036 0.029629441 -3.0882471 0.017178993 -5.326449
GAT3         0.009780852 0.014721289  0.6644019 0.002192983  4.460067
GCR2        -0.005837966 0.011396288 -0.5122690 0.003227041 -1.809077
MBP1         0.102623543 0.028474614  3.6040363 0.017389748  5.901382
MSN4         0.011652400 0.015301265  0.7615318 0.004533629  2.570215
NDD1        -0.068098866 0.027962789 -2.4353389 0.017078278 -3.987455
PHD1         0.018224333 0.017586392  1.0362747 0.006676215  2.729740
RGM1         0.031474714 0.022152842  1.4207980 0.006025010  5.224010
RLM1         0.004245315 0.009823147  0.4321746 0.003155203  1.345497
SMP1         0.018181353 0.017691495  1.0276889 0.007614400  2.387759
SRD1        -0.009422532 0.013882871 -0.6787164 0.005117179 -1.841353
STB1         0.038198667 0.022075228  1.7303860 0.017485954  2.184534
SWI4         0.007370389 0.012622711  0.5838990 0.004184668  1.761284
SWI6         0.033957904 0.022673644  1.4976818 0.013225660  2.567577
```

For comparison, we fitted the unpenalized GEEs via `MGEE` function under the same settings defined above as follows:

```
> # fit the GEE model
> myfit2 <- MGEE(formula = formula, id = id, data = data, na.action = NULL,
+ family = family, corstr = "independence", Mv = NULL,
+ beta_int = c(rep(0,dim(data)[2]-1)), R = NULL, scale.fix = TRUE,
+ scale.value = 1, maxiter = 30, tol = 10^-6, silent = TRUE)
```

Finally, we obtain the TFs which were significantly associated with the gene expression levels at G1 stage via PGEE and GEE analyses, respectively.

```
> # see the significantly associated TFs in PGEE analysis
> names(which(abs(coef(summary(myfit1))[index1,"Robust z"]) > 1.96))
 [1] "(Intercept)" "time"        "ABF1"        "FKH1"        "FKH2"        "GAT3"
 [7] "MBP1"        "MSN4"        "NDD1"        "PHD1"        "RGM1"        "SMP1"
[13] "STB1"        "SWI6"
> # see the significantly associated TFs in GEE analysis
```

```
> names(which(abs(coef(summary(myfit2))[,"Robust z"]) > 1.96))
 [1] "(Intercept)" "time"        "ABF1"        "ARG81"       "ASH1"        "CAD1"
 [7] "GAT3"        "GCN4"        "GCR1"        "GRF10.Pho2." "MBP1"        "MET31"
[13] "MET4"        "MTH1"        "NDD1"        "PDR1"        "ROX1"        "STB1"
[19] "STP1"        "YAP5"        "ZAP1"
```

When the results of PGEE and GEE analysis are compared, it is observed that while TFs such that *FKH1*, *FKH2*, *MSN4*, *PHD1*, *RGM1*, and *SMP1* are determined as significantly associated by PGEE analysis, these TFs are not detected by GEE analysis. The direct use of classical unpenalized GEE in high-dimensional longitudinal data analysis may lead to misleading results as the efficiency of PGEE over GEE has been showed in the simulation studies presented in Wang et al. (2012).

We repeat the steps above for each M/G1, G1, S, G2, and M stages under three different working correlation matrices (independence, exchangeable, and Ar(1)) and identify the number of TFs selected at each stage where the results are presented in Table 1. The results in Table 1 suggest that PGEE is robust to working correlation matrix specification and the selected TFs do not change significantly across working correlation matrix types within a stage. Furthermore, the analysis also reveals that different TFs may play different roles on gene expression levels in each cell-cycle process and there may be a small overlap in the selected TFs at different stages (e.g., only *FKH1*, *FKH2*, and *SMP1* are related to all stages of the yeast cell-cycle process).

Conclusion

In this paper, we present the R package **PGEE** which implements the PGEEs approach in Wang et al. (2012). The PGEE procedure performs simultaneous estimation and variable selection for longitudinal data analysis with high-dimensional covariates. We believe that this package is useful to practitioners in diverse fields where high-dimensional longitudinal data commonly arises such as genetics and large-scale health studies.

Acknowledgment

Lan Wang's research is supported by NSF grant NSF DMS-1512267. Gul Inan's visit to University of Minnesota is supported through a 2219-Fellowship by the Scientific and Technological Research Council of Turkey (TUBITAK). Authors would also like to thank the editor and the reviewer for their constructive comments which improved the quality of the paper.

Bibliography

P. Breheny and J. Huang. Coordinate descent algorithms for nonconvex penalized regression, with applications to biological feature selection. *Annals of Applied Statistics*, 5(1):232–253, 2011. URL https://doi.org/10.1214/10-AOAS388. [p396]

V. J. Carey. *gee: Generalized Estimation Equation Solver*, 2015. URL https://CRAN.R-project.org/package=gee. R package version 4.13-19. [p393]

H. Cho and A. Qu. Model selection for correlated data with diverging number of parameters. *Statistica Sinica*, 23:901–927, 2013. URL https://doi.org/10.5705/ss.2011.058. [p394]

J. Fan and R. Li. Variable selection via nonconcave penalized likelihood and its oracle properties. *Journal of the American Statistical Association*, 96(456):1348–1360, 2001. URL https://doi.org/10.1198/016214501753382273. [p394, 395, 396]

J. Friedman, T. Hastie, and R. Tibshirani. Regularization paths for generalized linear models via coordinate descent. *Journal of Statistical Software*, 33(1):1, 2010. URL https://doi.org/10.18637/jss.v033.i01. [p396]

A. Genz, F. Bretz, T. Miwa, X. Mi, and T. Hothorn. *mvtnorm: Multivariate Normal and t Distributions*, 2015. URL https://CRAN.R-project.org/package=mvtnorm. R package version 1.0-3. [p396]

J. Goeman. L1 penalized estimation in the Cox proportional hazards model. *Biometrical Journal*, 52(1):70–84, 2010. URL https://doi.org/10.1002/bimj.200900028. [p396]

U. Halekoh, S. Højsgaard, and J. Yan. The R package geepack for generalized estimating equations. *Journal of Statistical Software*, 15(2):1–11, 2006. URL https://doi.org/10.18637/jss.v015.i02. [p393]

J. W. Hardin and J. M. Hilbe. *Generalized Estimating Equations*. Wiley Online Library, 2003. [p394]

N. J. Horton and S. R. Lipsitz. Review of software to fit generalized estimating equation regression models. *The American Statistician*, 53(2):160–169, 1999. URL https://doi.org/10.1080/00031305.1999.10474451. [p394]

G. Inan, J. Zhou, and L. Wang. *PGEE: Penalized Generalized Estimating Equations in High-Dimension*, 2017. URL https://CRAN.R-project.org/package=PGEE. R package version 1.5. [p393]

B. A. Johnson, D. Lin, and D. Zeng. Penalized estimating functions and variable selection in semiparametric regression models. *Journal of the American Statistical Association*, 103(482):672–680, 2008. URL https://doi.org/10.1198/016214508000000184. [p395]

T. I. Lee, N. J. Rinaldi, F. Robert, D. T. Odom, Z. Bar-Joseph, G. K. Gerber, N. M. Hannett, C. T. Harbison, C. M. Thompson, I. Simon, et al. Transcriptional regulatory networks in saccharomyces cerevisiae. *Science*, 298(5594):799–804, 2002. URL https://doi.org/10.1126/science.1075090. [p398]

H. Lian, H. Liang, and L. Wang. Generalized additive partial linear models for clustered data with diverging number of covariates using GEE. *Statistica Sinica*, 24:173–196, 2014. [p393]

K.-Y. Liang and S. L. Zeger. Longitudinal data analysis using generalized linear models. *Biometrika*, 73 (1):13–22, 1986. URL https://doi.org/10.2307/2336267. [p393, 394]

B. Ripley. *MASS: Support Functions and Datasets for Venables and Ripley's MASS*, 2015. URL https://CRAN.R-project.org/package=MASS. R package version 7.3-45. [p396]

B. Sherwood and A. Maidman. *rqPen: Penalized Quantile Regression*, 2016. URL https://cran.r-project.org/web/packages/rqPen. R package version 1-4. [p396]

P. T. Spellman, G. Sherlock, M. Q. Zhang, V. R. Iyer, K. Anders, M. B. Eisen, P. O. Brown, D. Botstein, and B. Futcher. Comprehensive identification of cell cycle–regulated genes of the yeast saccharomyces cerevisiae by microarray hybridization. *Molecular Biology of Cell*, 9(12):3273–3297, 1998. URL https://doi.org/10.1091/mbc.9.12.3273. [p397]

L. Wang. GEE analysis of clustered binary data with diverging number of covariates. *The Annals of Statistics*, 39(1):389–417, 2011. URL https://doi.org/10.1214/10-AOS846. [p393]

L. Wang, G. Chen, and H. Li. Group SCAD regression analysis for microarray time course gene expression data. *Bioinformatics*, 23(12):1486–1494, 2007. URL https://doi.org/10.1093/bioinformatics/btm125. [p394, 397]

L. Wang, J. Zhou, and A. Qu. Penalized generalized estimating equations for high-dimensional longitudinal data analysis. *Biometrics*, 68(2):353–360, 2012. URL https://doi.org/10.1111/j.1541-0420.2011.01678.x. [p393, 394, 395, 396, 397, 401]

L. Wang, L. Xue, A. Qu, and H. Liang. Estimation and model selection in generalized additive partial linear models for correlated data with diverging number of covariates. *The Annals of Statistics*, 42(2): 592–624, 2014. URL https://doi.org/10.1214/13-AOS1194. [p393]

S. L. Zeger and K.-Y. Liang. Longitudinal data analysis for discrete and continuous outcomes. *Biometrics*, 4(1):121–130, 1986. [p393]

Gul Inan
Department of Statistics
Middle East Technical University
Ankara, 06800
Turkey
ginan@metu.edu.tr

Lan Wang
School of Statistics
University of Minnesota
Minneapolis, MN 55455
USA
wangx346@umn.edu