Journal of Data Science 24 (4), 802–821
July 2026

DOI: 10.6339/25-JDS1207
Computing in Data Science

The R Package geeVerse for Ultra-High-Dimensional Heterogeneous Data Analysis with Generalized Estimating Equations

Tianhai Zu[1,*], Brittany Green[2], and Yan Yu[3]

[1]*Department of Operations and Analytics, University of Texas at San Antonio, United States of America*
[2]*Department of Information Systems, Analytics, and Operations, University of Louisville, United States of America*
[3]*Department of Operations, Business Analytics and Information Systems, University of Cincinnati, United States of America*

Abstract

High or ultra-high-dimensional data are becoming increasingly common in various fields. They often display diverse characteristics, including heterogeneity, longitudinal responses, and imbalanced measurements. These complexities make it challenging to integrate different modeling options and their combinations in order to fully leverage this rich data source. This paper provides an easy-to-use, and stand-alone, R package, *geeVerse*, that can implement any combination of 1) simultaneous variable selection and estimation, 2) quantile regression or mean regression for heterogeneous data, 3) longitudinal or cross-sectional data analysis, 4) balanced or imbalanced data, and 5) moderate, high, or even ultra-high-dimensional data. To accomplish this, we propose computationally efficient implementations of penalized generalized estimating equations (GEE) for quantile and mean regression. We present multiple applications with ultra-high-dimensional data including analysis of a resampled genetic dataset, quantile and mean regressions, analysis of cross-sectional and longitudinal data, differing correlation structures, and differing number of repeated measurements per subject. We also demonstrate our approach on two real data applications.

Keywords *GEE; longitudinal data; quantile; variable selection*

1 Introduction

High and even ultra-high-dimensional longitudinal datasets are becoming increasingly prevalent in various fields. Such rich data enable in-depth analysis to identify important features and provide valuable insights. For example, in Zu et al. (2023), the phenotype response variable of interest is blood pressure levels measured at four waves of a study spanning 17 years. The risk factors involve over $500,000$ single nucleotide polymorphisms (SNPs) along with time-varying phenotype variables such as age, total cholesterol, smoking status, and body mass index of about $1,500$ participants. The primary goal is to identify a sparse set of important genotype and phenotype risk factors associated with blood pressure levels, particularly focusing on the most perilous levels at high quantiles. Zu et al. (2023) advocate simultaneous variable selection

*Corresponding author Email: tianhai.zu@utsa.edu.

© 2026 The Author(s). Published by the School of Statistics and the Center for Applied Statistics, Renmin University of China. Open access article under the CC BY license.
Received September 11, 2025; Accepted November 2, 2025

and estimation through penalized quantile regression for longitudinal data as an alternative approach to traditional genome-wide association studies (GWAS) that focus on one SNP at a time at the mean level (e.g., Evangelou et al. 2018). Penalized quantile regression is particularly useful in such applications, as different risk factors may be associated with dangerously high blood pressure levels compared to those for mean blood pressure levels. Furthermore, quantile regression is advantageous as high-dimensional data tend to be heterogeneous.

Generalized estimating equations (GEE) (Liang and Zeger, 1986) are widely used for modeling the conditional mean of longitudinal data, incorporating within-subject correlations. GEE's popularity also stems from its robustness, as it does not require specifying the full joint distribution or the correct working correlation matrix to achieve asymptotic consistency. R packages *gee* (Carey et al., 2023), *geeM* (McDaniel et al., 2013), and *geepack* (Højsgaard et al., 2006) are popular packages that implement GEE focusing primarily on modeling the conditional mean of longitudinal data. For variable selection and estimation of the conditional mean, penalized generalized estimating equations is an appealing approach (Wang et al., 2012), and an associated R package *PGEE* incorporates simultaneous variable selection and estimation for mean regression (Inan and Wang, 2017). Another package, *LassoGEE* (Li et al., 2023), focuses on the least absolute selection and shrinkage operator (LASSO) as the penalty function for variable selection, leveraging a computationally efficient iterative composite gradient descent algorithm. Additionally, the package *pgee.mixed* (Deshpande and Deshpande, 2023) provides an option for variable selection with bivariate mixed outcomes utilizing penalized generalized estimating equations.

There is a notable gap in software tools for modeling conditional quantiles with longitudinal responses and variable selection capabilities, which are crucial for many applications. Existing GEE packages are designed for longitudinal responses but focus primarily on modeling the conditional mean, whereas quantile regression R packages, such as *quantreg* (Koenker, 2025) for standard quantile regression (Koenker and Bassett Jr, 1978) and *rqPen* (Sherwood et al., 2023) for penalized quantile regression, are designed for cross-sectional data and do not address the specific challenges posed by longitudinal data structures. To address this gap, we introduce *geeVerse*, a R package that provides simultaneous variable selection and estimation for conditional quantile models for high- and ultra-high-dimensional longitudinal data using penalized generalized estimating equations.

The key contributions of our *geeVerse* package include: 1) It is the first package to implement quantile regression with simultaneous variable selection and estimation for ultra-high-dimensional longitudinal data, based on the recent work of Zu et al. (2023). This package accommodates various characteristics of high-dimensional datasets not currently contained in one package. 2) This package also provides an updated, more computationally efficient version of the *PGEE* package for conditional mean regression. This package also does not depend on other variable selection packages such as *glmnet*. It only depends on the *quantreg* (Koenker, 2025) package for initial values and the *Rcpp* package (Eddelbuettel and François, 2011). By offering these features, *geeVerse* serves as a companion to the methodological advancements presented in Zu et al. (2023), providing a versatile tool for analyzing complex, high-dimensional longitudinal data across various quantiles of interest.

2 Example Datasets

To illustrate the types of problems our package can be applied to, we first present an imbalanced longitudinal dataset, suitable for studying the progression of HIV in patients. We then present

a realistic high-dimensional simulated genetic dataset, as the participants' genetic data from Zu et al. (2023) cannot be publicly shared. Users can alternatively obtain genetic data by applying to the National Institutes of Health (NIH) with Institutional Review Board (IRB) approval (National Center for Biotechnology Information, 2025).

2.1 CD4 in HIV Patients

To demonstrate a typical usage of the *geeVerse* package, particularly its ability to handle imbalanced longitudinal data, we analyze the AIDS dataset from the R package *JM* (Rizopoulos, 2010). This dataset contains CD4 cell count measurements from 467 HIV-infected patients who participated in a randomized clinical trial, as shown in the following code chunks. The CD4 count, a key indicator of immune system health, along with other variables, was measured at irregular intervals over time, resulting in an imbalanced longitudinal structure. The number of observations per patient ranges from 1 to 5, with a median of 3 measurements, exemplifying the type of imbalanced data often encountered in clinical studies. We focus on four key covariates: drug treatment (a binary variable indicating whether the patient received the drug ddI or ddC), gender (male or female), previous opportunistic infection (prevOI) (a binary variable indicating whether the patient had a prior opportunistic infection), and AZT use (a binary variable indicating prior use of the drug zidovudine).

Using this data, we aim to identify which factors significantly influence CD4 cell counts, examining effects not only at the average but also across different quantiles with *geeVerse*. This approach allows us to investigate factors affecting not only the average CD4 count but also lower quantiles that could indicate an increased risk for opportunistic infections. Below is the code to present the first few rows of the data, with an analysis of this dataset provided in Section 5 and more information on the code in Section 4.

```
## Load the "aids" dataset from the JM package and present first several rows
if (!require("JM")) install.packages("JM"); library("JM")
head(aids[, c("patient","CD4","obstime","drug","gender","prevOI","AZT")])
```

```
##   patient       CD4 obstime drug gender prevOI         AZT
## 1       1 10.677078       0  ddC   male   AIDS intolerance
## 2       1  8.426150       6  ddC   male   AIDS intolerance
## 3       1  9.433981      12  ddC   male   AIDS intolerance
## 4       2  6.324555       0  ddI   male noAIDS intolerance
## 5       2  8.124038       6  ddI   male noAIDS intolerance
## 6       2  4.582576      12  ddI   male noAIDS intolerance
```

2.2 A Simulated Genetic Dataset Using HapGen2

This second example demonstrates a key application of the *geeVerse* package: identifying important genetic markers in a high-dimensional setting while simultaneously controlling for longitudinal clinical variables. The genetic markers considered are single nucleotide polymorphisms (SNPs), which are differences at individual nucleotide positions in the DNA sequence and occur in the millions genome-wide. This scenario is common in modern genetic studies and highlights two distinct advantages of our package. First, it showcases the ability to perform variable selection among a large number of SNPs to pinpoint true risk factors. Second, the simulation uses

a realistic genetic structure, making it a valuable tool for researchers. The first several rows of the simulated data are shown as follows:

```
## Load the geeVerse library
if (!require("geeVerse")) install.packages("geeVerse"); library("geeVerse")
```

```
## Loading required package: geeVerse
```

```
set.seed(2024)
sim_data <- generate_data(nsub = 1000, nobs = rep(5, 1000), p = 50,
                          beta0 = c(rep(1, 9), rep(0, 41)), rho = 0.6,
                          corstr = "exchangeable", ka = 0.5,
                          SNPs = simuGene[, 1:25])
head(round(sim_data[, 1:9], 2), 7)
```

```
##      y id control1 control2 control3 control4 control5 rs5762171 rs5996304
## 1 6.41  1     0.69     0.00    -0.77     0.13     0.79         1         1
## 2 6.76  1     0.04    -0.36     0.83     0.47     0.91         1         1
## 3 4.37  1    -1.27    -1.93     0.79     0.08     1.90         1         1
## 4 2.25  1     0.36    -1.18    -0.81     0.02    -0.65         1         1
## 5 6.25  1     0.34     0.39     0.83    -1.20     0.38         1         1
## 6 5.76  2     0.47     0.23     1.00     0.86    -1.63         0         1
## 7 4.14  2     0.46    -0.82     0.85     0.00    -0.56         0         1
```

The genetic markers in the `simuGene` dataset, provided in our *geeVerse* package, are generated using HapGen2 (Su et al., 2011), a widely used tool for resampling existing genotype data from the publicly available 1000 Genomes Project (1000 Genomes Project Consortium, 2015), accessed via the Impute2 webpage. HapGen2 employs a resampling-based approach to construct genotypes for simulated samples, preserving allele frequency and linkage disequilibrium (LD) patterns to ensure a realistic simulation environment. The resulting dataset includes 1000 simulated subjects, with 500 assigned higher-risk alleles and 500 assigned lower-risk alleles, containing only SNPs data due to CRAN package size limitations.

We use the `generate_data()` function to simulate the time-varying control variables and the longitudinal response variable using the `simuGene` data as input. To generate the response variable, we simulate using the setup described in Equation (3) from Example 1, incorporating the previously defined SNPs and control covariates. We specify 4 true signals among the SNPs and 5 true signals among the control variables, with heteroscedasticity in the error term. In the above R code, we used the first 25 resampled simulated SNPs and generated the other 25 simulated control variables, resulting in a total of $p = 50$ covariates.

We demonstrate the code to perform simultaneous variable selection and coefficient estimation for quantile regression at $\tau = 0.5$ (i.e., median regression) as below. A more detailed explanation of the code can be found in Section 4.

```
PQGEE_fit_median <- qpgee(y ~ . - id, data = sim_data, id = id,
                          corstr = "exchangeable", tau = 0.5,
                          method = "HBIC", ncore = 10)
summary(PQGEE_fit_median)
```

3 Models

Our package is flexible enough to handle various data characteristics, including cross-sectional or longitudinal data, quantile or mean regression, and high or ultra-high-dimensional covariates, where the number of covariates grows at an exponential order of the sample size (Fan and Lv, 2008). Importantly, it also supports any combination of these characteristics. To accommodate this variety, we consider data where each of $i = 1, \ldots, n$ participants has m_i repeated measurements of the response variable. That is, the response for participant i is $\mathbf{Y}_i = (Y_{i1}, Y_{i2}, \ldots, Y_{im_i})^\top$, where each participant may have a different number of measurements taken over time. If $m_i = 1$ for all participants, then the data is cross-sectional. If any participant has a value of m_i greater than 1, then the data is longitudinal. Repeated measurements from the same participant may be dependent, but observations from different participants are independent.

To incorporate the dependence structure of the response when the data is longitudinal, we implement the commonly used generalized estimating equations (Liang and Zeger, 1986). A key feature of GEE is that the estimates of the mean model parameters remain asymptotically consistent even if the specified correlation structure is incorrect. However, a working correlation structure must be chosen to construct the estimating equations. This choice influences the statistical efficiency of the parameter estimates and is used to build the robust sandwich variance estimator, which correctly accounts for the intra-subject dependence to provide valid standard errors. Common choices for the working correlation matrix include first-order autoregressive (AR1), exchangeable, and independence.

Accompanying each observation of the response at time point $j \in \{1, \ldots, m_i\}$, there are observed p-dimensional covariates defined as $\mathbf{X}_{ij} = (X_{ij1}, X_{ij2}, \ldots, X_{ijp})^\top$. Here the covariate dimension can be moderate, high, or even ultra-high. In high and ultra-high-dimensional datasets, sparsity is also common, where only a small amount of covariate coefficients are non-zero. Some covariates can be time-varying, which means covariates can have different values at each time point $j \in (1, \ldots, m_i)$. The covariates for participant i across all time points are defined as, $\mathbf{X}_i = (\mathbf{X}_{i1}, \mathbf{X}_{i2}, \ldots, \mathbf{X}_{im_i})^\top$.

To analyze this data, *geeVerse* allows users to utilize either mean or quantile regression, depending on their needs. Both approaches aim to identify a sparse set of important variables from a large set of potentially high-dimensional covariates for the corresponding conditional mean or quantile. This creates challenges in variable selection and estimation while also incorporating the dependence structure among longitudinal data. To address these challenges, we utilize a penalization approach that jointly models the data through simultaneous variable selection and estimation along with GEE. We consider the mean and quantile regression approaches separately.

When conditional quantiles are of interest, we incorporate quantile regression (Koenker and Bassett Jr, 1978) to identify important covariates for different conditional quantile response levels. In quantile regression, we are interested in the τth conditional quantile

$$\theta_\tau(\mathbf{Y}_i|\mathbf{X}_i) = \mathbf{X}_i\boldsymbol{\beta}_\tau, \tag{1}$$

with $\tau \in (0, 1)$. $\boldsymbol{\beta}_\tau = (\beta_{\tau 1}, \beta_{\tau 2}, \ldots, \beta_{\tau p})^\top$ is the coefficient vector where p can diverge and even be in ultra-high-dimensions, and $\boldsymbol{\beta}_\tau$ can have different values at distinct quantile levels τ. Here important covariates at one conditional quantile may differ from those at another conditional quantile or from those relevant to the mean.

To simultaneously estimate the coefficients $\boldsymbol{\beta}_\tau$ and select variables under model (1), we implement quantile penalized generalized estimating equations (Zu et al., 2023), incorporating within-subject correlations of the longitudinal responses. Let $\tau - I\{\mathbf{Y}_i \leqslant \mathbf{X}_i\boldsymbol{\beta}_\tau\}$ be the subgradient

of the check loss function. Let $\mathbf{W}_i$ be a weight matrix, where identity is used for simplicity of computation as suggested in Zu et al. (2023). Let $\mathbf{R}_i$ be a working correlation matrix. And let $\mathbf{q}_\lambda(|\boldsymbol{\beta}|)\mathbf{sgn}(\boldsymbol{\beta})$ be the penalty term as defined in Zu et al. (2023). The quantile penalized generalized estimating equations are

$$\mathbf{S}^P_{quant}(\boldsymbol{\beta}_\tau) = \sum_i \mathbf{X}_i^\top \mathbf{W}_i \mathbf{R}_i^{-1}(\tau - I\{\mathbf{Y}_i \leqslant \mathbf{X}_i\boldsymbol{\beta}_\tau\}) - n\mathbf{q}_\lambda(|\boldsymbol{\beta}_\tau|)\mathbf{sgn}(\boldsymbol{\beta}_\tau). \tag{2}$$

This equation consists of two main components: 1) a GEE component along with the subgradient of the quantile check loss and 2) the penalty term. Users can utilize one of the popular working correlation matrices: AR1, exchangeable, independence, or unstructured. Alternatively, users can provide their own user-defined working correlation matrix derived from the data or other approach. When the data is cross-sectional, an independence working correlation matrix is implemented as the default. For the choice of the penalty function, our *geeVerse* package implements the popular smoothly clipped absolute deviation (SCAD) penalty (Fan and Li, 2001) and the least absolute selection and shrinkage operator (LASSO) (Tibshirani, 1996). Here λ is the penalty tuning parameter controlling the level of regularization. For more detailed explanation and derivation, see Zu et al. (2023).

We apply iterative computational algorithms for effective analysis, given the challenges in estimating coefficients and selecting variables from sparse, ultra-high-dimensional data. The complexities involve the non-convex SCAD penalty function, the discontinuous quantile check function for the conditional quantile, and sparse, ultra-high-dimensional covariates. To address these, we apply an iterative algorithm for the conditional quantile model. We provide detailed information on the computational algorithms in the Supplementary Material.

4 Implementation Details

The core of the *geeVerse* package consists of two primary estimation functions: `qpgee()` for quantile models and `pgee()` for conditional mean models. These functions are designed to analyze longitudinal data with potentially ultra-high-dimensional predictors by implementing quantile penalized generalized estimating equations, which perform simultaneous variable selection and estimation.

4.1 Computational Algorithm

The computational engine of `qpgee()` is a robust iterative algorithm designed to solve the penalized estimating equation (2). The overall procedure involves two nested loops: an outer loop for selecting the optimal penalty tuning parameter λ and an inner loop for estimating the coefficients $\boldsymbol{\beta}_\tau$ for a given λ.

For the outer loop, when the optimal penalty is unknown, the algorithm automates the selection of λ by evaluating a grid of candidate values. This selection is guided by one of two criteria: the High-Dimensional Bayesian Information Criterion (HBIC) or k-fold Cross-Validation (CV). The algorithm selects the λ value that minimizes the chosen criterion.

The inner loop solves for $\boldsymbol{\beta}_\tau$ for a fixed λ. Since the indicator function in Equation (2) is discontinuous, direct application of gradient-based methods like Newton-Raphson is not possible. To overcome this, we first approximate the discontinuous function with a smooth one, making the estimating equation differentiable. A detailed derivation of this smoothed approximation is

Algorithm 1 QPGEE fitting procedure in *geeVerse*.

Input: Data $\{(\mathbf{Y}_i, \mathbf{X}_i)\}_{i=1}^n$, quantile τ, penalty grid Λ, convergence tolerance ϵ_{conv}, and shrinkage cutoff ϵ_{shrink}.
Initialize: Obtain initial coefficient vector $\hat{\boldsymbol{\beta}}^{(0)}$ from a non-penalized fit (e.g., `quantreg::rq`).
Outer Loop: Tuning Parameter Selection
for each penalty tuning parameter $\lambda \in \Lambda$ **do**
  Set $\hat{\boldsymbol{\beta}}_\lambda^{(0)} \leftarrow \hat{\boldsymbol{\beta}}^{(0)}$ and initialize the active set $\mathcal{A} \leftarrow \{1, \ldots, p\}$.
  $t \leftarrow 0$.
  repeat
    (a) Compute the working correlation matrix $\mathbf{R}$ based on the user-specified structure (`corstr`) and current residuals from $\hat{\boldsymbol{\beta}}_\lambda^{(t)}$.
    (b) Update coefficients for covariates in the active set $\mathcal{A}$ by applying one Newton-Raphson step to the smoothed penalized estimating equation:

$$\hat{\boldsymbol{\beta}}_{\mathcal{A}}^{(t+1)} \leftarrow \hat{\boldsymbol{\beta}}_{\mathcal{A}}^{(t)} - [\mathbf{H}(\hat{\boldsymbol{\beta}}_{\mathcal{A}}^{(t)})]^{-1} \tilde{\mathbf{S}}_{quant}^{P}(\hat{\boldsymbol{\beta}}_{\mathcal{A}}^{(t)}),$$

    where $\tilde{\mathbf{S}}_{quant}^{P}$ is the smoothed version of Eq. (2) and $\mathbf{H}$ is its derivative (see Supplementary Material for details). Coefficients for covariates not in $\mathcal{A}$ remain zero.
    (c) If $t \geqslant 2$, update the active set: For each index $j \in \mathcal{A}$, if $|\hat{\beta}_{\lambda,j}^{(t+1)}| < \epsilon_{shrink}$, set $\hat{\beta}_{\lambda,j}^{(t+1)} \leftarrow 0$ and remove j from $\mathcal{A}$.
    $t \leftarrow t + 1$.
  until $||\hat{\boldsymbol{\beta}}_\lambda^{(t)} - \hat{\boldsymbol{\beta}}_\lambda^{(t-1)}|| < \epsilon_{conv}$ or maximum iterations reached.
  Store the converged estimate $\hat{\boldsymbol{\beta}}_\lambda$ and calculate its selection criterion (HBIC or CV error).
end for
Final Model Selection:
Find $\lambda^* = \arg\min_{\lambda \in \Lambda} \text{Criterion}(\hat{\boldsymbol{\beta}}_\lambda)$.
Output: Return final coefficients $\hat{\boldsymbol{\beta}}_{\lambda^*}$.

provided in Zu et al. (2023). With this smooth approximation, the algorithm iteratively estimates the coefficients using a Newton-Raphson procedure.

To enhance computational efficiency, this estimation process employs an active set strategy. The algorithm begins with an "active set" containing all covariates. To mitigate the influence of potentially unstable initial estimates, this active set is updated starting from the third iteration. The algorithm and the dynamic truncation to 0 may depend on the initial estimate $\hat{\beta}_0$, which is set to the unpenalized GEE estimate for robustness. Empirical results show convergence is stable across various initial values. For general guidelines on obtaining starting values, see Bates and Watts (1988) and Nocedal and Wright (2006). In each subsequent iteration of the inner loop, any coefficient whose absolute value falls below a very small, pre-defined tolerance, `shrinkCutoff` (e.g., 10^{-4}, controllable via `qpgeeControl()`), is set to zero and removed from the active set. This dynamic reduction in the problem's dimensionality significantly accelerates convergence by focusing computation only on potentially non-zero predictors. When `shrinkCutoff` is set sufficiently small, it does not change the selected model.

The complete computational procedure, integrating both tuning and estimation, is summarized in Algorithm 1. Please note that if predictors are standardized, coefficients are back-transformed to their original scale before being returned.

4.2 Function `qpgee()`

The `qpgee()` function is the primary implementation of the fitting procedure detailed in Algorithm 1. It provides an interface for fitting penalized quantile GEE models with the following key features:

- Model Specification: The function requires a `formula`, a `data.frame`, and an `id` vector for the model structure. The use of a standard `formula` allows for familiar model specification and automatically handles the conversion of categorical variables into the appropriate dummy codes, similar to the behavior of core functions like `lm()`. The `id` argument identifies the subject or cluster to account for within-subject correlations. The desired conditional quantile is specified via the `tau` argument, which can be any value in $(0, 1)$.
- Penalization and Tuning: It performs simultaneous variable selection and estimation using either the SCAD (default) or LASSO penalty. The optimal tuning parameter `lambda` is selected automatically by specifying the `method` argument as `"HBIC"` (the default) or `"CV"` for cross-validation. Users can also supply a specific `lambda` value or a custom penalty grid of candidate values Λ, or simply 0 for no penalty. If no `lambda` is provided, a predefined penalty grid, Λ, of 30 (default) penalty values that decrease from 10 to 0.01 at the $\log_{10}$ scale for the high-dimensional scenario or from 10 to 0.001 at the $\log_{10}$ scale for the low-dimensional scenario, will be used.
- Dependence Structure: The within-subject dependence is modeled through the `corstr` argument, which accepts common structures such as `"AR1"`, `"exchangeable"`, `"independence"`, and `"unstructured"`.
- User Control: Finer control over the iterative fitting process is managed via the `control` argument, which takes a list of parameters generated by the `qpgeeControl()` function. Key options include `maxit` to set the maximum number of iterations, `epsilon` for the convergence tolerance, and `shrinkCutoff` for the threshold at which coefficients are shrunk to zero. The control function also houses the logical flag `standardize` for predictor scaling.
- Parallel Processing and Options: In addition, users can supply initial values with `betaint`, while the default being non-penalized, cross-sectional quantile regression estimates using *(*quantreg). The user can enable parallel processing to reduce computation time with `ncore`. If `ncore` is not specified, `qpgee()` will only use a single CPU core to ensure maximum compatibility.

4.3 Function `pgee()`

For analyses focused on the conditional mean, the package includes the `pgee()` function. Its main characteristics are:

- Purpose: It implements penalized GEE for mean regression, serving as a computationally accelerated alternative to the implementation in the *PGEE* package (Inan and Wang, 2017).
- Performance: The significant speed improvement is achieved through optimizations such as the re-implementation of core computational routines in `C++`, the use of efficient data structures, and the reduction of redundant matrix calculations.
- Interface and Compatibility: To ensure a seamless user experience, `pgee()` intentionally retains an identical interface and argument structure to the function in the original *PGEE* package. Thus, users can refer to the official documentation of the *PGEE* package for detailed descriptions. This design also ensures compatibility and allows for the use of its established auxiliary functions like `CVfit()` for tuning parameter selection.

Table 1: List of primary functions in *geeVerse* R package.

	Functions	Usage
Core	`qpgee()`	This function provides versatile modeling for both longitudinal and cross-sectional data, accommodating various quantile levels and employing both penalized and non-penalized approaches. This function automatically selects the optimal penalty level based on criteria such as the high-dimensional BIC and cross-validation. When no penalty parameter is provided, the function returns a `qpgee` model configured with this auto-selected penalty.
	`pgee()`	This function offers a longitudinal analysis of conditional mean regression with penalized GEE, which is an accelerated re-implementation of the existing PGEE package, enhanced with C++ and efficient procedures.
Generic	`predict()`	This S3 function generates predictions from `qpgee` models for both fitted and new data.
	`summary()`	This S3 function delivers a concise summary of `qpgee` model estimates, working correlation matrices and diagnostics.
	`generate_data()`	This function generates simulated longitudinal data, providing a convenient way to demonstrate the capabilities of the `qpgee()` function and test other methods. The procedure is detailed in Algorithm 2.

4.4 Generic Functions

In addition to the core functions of the *geeVerse* package, the generic functions enhance the package's versatility and user-friendliness. These functions assist users in predicting new data, interpreting results, and generating longitudinal data for numerical studies. More details about these functions can be found in Table 1. Please note this table does not include internal functions such as those handling different working correlation structures and penalty types.

5 Illustrations

Our versatile *geeVerse* package incorporates simultaneous variable selection and estimation for both conditional quantile and mean regression, longitudinal data that could be imbalanced, and potentially ultra-high-dimensional predictors. To demonstrate the usage of our *geeVerse* package, we analyze two real data applications.

5.1 CD4 in HIV Patients Example: A Closer Look

To showcase the versatility of the analysis possible with the *geeVerse* package, we conducted an analysis of the AIDS CD4 dataset using different quantile levels (τ) and correlation structures.

Algorithm 2 Data generation procedure in `generate_data()`.

Input: n subjects, $\{m_i\}$ obs/subject, p predictors, coefficients $\boldsymbol{\beta}_0$, correlation ρ, structure `corstr`, error distribution `dis`, heteroskedasticity κ, optional SNPs $\mathbf{G}$.
For each subject $i = 1, \ldots, n$:
Construct correlation matrix $\boldsymbol{\Sigma}_i$ $(m_i \times m_i)$ from ρ and `corstr`.
Draw error vector $\boldsymbol{\epsilon}_i$ from a multivariate distribution with covariance $\boldsymbol{\Sigma}_i$.
Concatenate subject errors $\{\boldsymbol{\epsilon}_i\}_{i=1}^{n}$ into a single vector $\boldsymbol{\epsilon}$ of length $N = \sum m_i$.
Generate predictor matrix $\mathbf{X}$ $(N \times p)$:
if $\mathbf{G}$ is provided **then**
    Combine longitudinal SNP data from $\mathbf{G}$ with control variables drawn from $N(0, 1)$.
else
    Draw all elements of $\mathbf{X}$ from $N(0, 1)$.
end if
Compute response $\mathbf{Y} = \mathbf{X}\boldsymbol{\beta}_0 + (\mathbf{1} + \kappa|\mathbf{X}_{:,1}|) \odot \boldsymbol{\epsilon}$.
Output: A data frame containing the response $\mathbf{Y}$, subject identifiers **id**, and predictors $\mathbf{X}$.

This analysis demonstrates how the package can provide insights into various aspects of the data, particularly in scenarios where the relationships between variables may differ across different quantiles of the response distribution.

We analyze the data using a low quantile level $\tau = 0.3$ with an AR1 correlation structure. The choice of an AR1 correlation structure for this analysis assumes that correlations between observations decrease exponentially with time. This is appropriate for longitudinal data where measurements closer in time are more strongly correlated, which aligns with the expected behavior of CD4 counts in HIV patients.

```
## Fit the qpgee model at tau = 0.3
qpgee_fit_aids_low <- qpgee(CD4 ~ gender + drug + prevOI + AZT + 0,
                            data = aids, id = patient,
                            corstr = "AR1", tau = 0.3, method = "HBIC")
summary(qpgee_fit_aids_low)
```

```
## Quantile Penalized Generalized Estimating Equations (QPGEE)
## Formula:  CD4 ~ gender + drug + prevOI + AZT + 0
## Number of Clusters:  467
## Number of observations: 1405
##
## Coefficients:
## genderfemale   gendermale      drugddI   prevOIAIDS   AZTfailure
##       5.4835       5.6487       0.4954      -3.0296       0.0000
##
## Converged:  TRUE
## Best lambda:  0.06723
## HBIC:  6.461
##
## Correlation structure: AR1
## Estimated correlation matrix (or parameters):
```

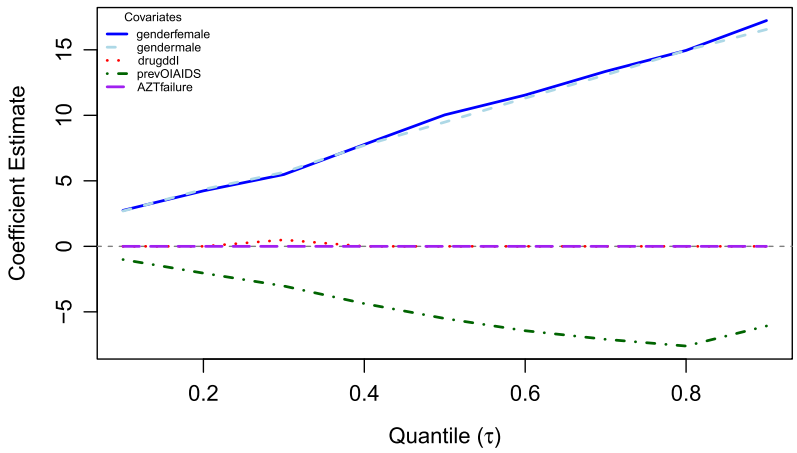


Figure 1: Quantile regression coefficient estimates for the AIDS dataset across different quantiles of CD4 counts. Each line represents the estimated effect of a covariate as the conditional quantile τ varies from 0.1 to 0.9.

```
##         [,1]   [,2]   [,3]   [,4]   [,5]
## [1,] 1.0000 0.5200 0.2704 0.1406 0.0731
## [2,] 0.5200 1.0000 0.5200 0.2704 0.1406
## [3,] 0.2704 0.5200 1.0000 0.5200 0.2704
## [4,] 0.1406 0.2704 0.5200 1.0000 0.5200
## [5,] 0.0731 0.1406 0.2704 0.5200 1.0000
```

We show the fitted `qpgee` model via the `summary()` function. The output includes the formula used to fit the model, the number of clusters, the list of coefficients estimated, whether the algorithm converged, the lambda value, and the minimum HBIC if the lambda is selected via HBIC.

To provide a more comprehensive view of how covariate effects vary across different quantiles of CD4 counts, Figure 1 shows the coefficient estimates for each covariate against the quantile level τ, from 0.1 to 0.9. The plot reveals considerable heterogeneity in the covariate effects across different quantiles. For example, previous opportunistic infection of AIDS (prevOIAIDS) shows a negative coefficient that intensifies with quantiles level until $\tau = 0.8$.

5.2 Gene Expression of Yeast Cell Example

We show the usage of our *geeVerse* package on the publicly available yeast cell cycle gene expression dataset, `yeastG1`, housed within the *geeVerse* package and the *PGEE* package (Inan and Wang, 2017). This dataset contains a subset of the data from the study in Spellman et al. (1998), which aimed to identify transcription factor proteins (TFs) that are influential and help regulate the cell cycle process. The cell cycle is a complex process consisting of stages related to cell growth, replication, and division. Transcription factors can determine the extent of genetic material transcription from DNA to mRNA during this process.

Our objective is to identify important covariates associated with gene expression during the G1 stage of the cell cycle, which is the response in this dataset. The covariates include "time" corresponding to the time point of the repeated measurement and 96 time-invariant transcription factor covariates calculated using the mixture modeling approach (Wang et al., 2007). We fit two

models, the conditional mean and the conditional 90th quantile, to this data and then compare the variable selection results.

We first load the data and set up the dataset to be analyzed for both models. The data has 283 subjects. For each subject, the 97 covariates, and the response, are measured over 4 time points. Note that the predictors have been standardized, no further standardization is required.

To fit the conditional 90th quantile model, we can either use default initial values or provide user-defined initial values. In this analysis, we present an example of using user-defined initial values from the cross-sectional non-penalized quantile regression function `rq()` in the quantreg package.

```
if (!require("quantreg")) install.packages("quantreg")
betaint = coefficients(quantreg::rq( y ~ . - id, data = yeastG1, tau = 0.5))
```

Then we use the `qpgee()` function in *geeVerse* with cross-validation to automatically choose the penalty parameter. We supply both the conditional quantile of interest, $\tau = 0.5$, and the exchangeable working correlation structure.

```
data("yeastG1")
qpgee_fit_G1 <- qpgee(y ~ . - id, data = yeastG1, id = id,
                      corstr = "exchangeable", tau = 0.5, method = "CV",
                      betaint = betaint, ncore = 10, control =
                      qpgeeControl(shrinkCutoff = 10^-6))
```

The important covariates with non-zero estimated coefficients selected for the 90th quantile are listed below. Here 10^{-3} is used as the threshold value to determine if a coefficient should be considered as 0 as in previous studies (Wang et al., 2012).

```
qpgee_fit_G1$coefficients[abs(qpgee_fit_G1$coefficients) > 10^-3]
```

```
##  (Intercept)         time         ABF1         FKH2         GAT3         MBP1
##  0.102013384  0.008392591 -0.003196664 -0.188063463  0.436695655  0.194505385
##         YAP5
## -0.334397314
```

We fit the conditional mean using the more efficient version of penalized GEE from our package. The main functions for this are `CVfit()` and `pgee()`.

```
CVfit_result <- CVfit(y ~ . - id , id = yeastG1$id, data = yeastG1, fold = 5)
PGEE_fit <- PGEE(y ~ . - id , id = yeastG1$id, data = yeastG1,
                 corstr = "exchangeable", lambda = CVfit_result$lam.opt)
```

The resulting covariates chosen as important are listed below for the mean.

```
PGEE_fit$coefficient[abs(PGEE_fit$coefficient) > 10^-3]
```

```
##  (Intercept)         time         ABF1        ARG81         FKH1         FKH2
##  0.006019800  0.017226351 -0.011287671  0.001326778 -0.010764345 -0.092378179
```

```
##         GAT3         GCR2         HIR1         IXR1         MBP1         MET4
##  0.038950040 -0.011733726 -0.002392962 -0.002435541  0.104979177 -0.007297364
##         MSN4         NDD1         PHD1         REB1         RLM1         SMP1
##  0.014100643 -0.068104270  0.017798989 -0.002378458  0.004115703  0.021574657
##         SRD1         STB1         STP1         SWI4         SWI6
## -0.005065792  0.039869082  0.005331856  0.008568817  0.035159494
```

Comparing the variable selection results from both models suggests that while the transcription factors identified as important do share a common set, there are also differences between the mean and the median. Analyzing both the conditional quantiles and the conditional mean can provide a more comprehensive understanding of the relationship between transcription factors and yeast cell cycle regulation.

5.3 Simulation Examples

To demonstrate the versatility of *geeVerse*, we also analyze three simulated datasets: longitudinal data with high-dimensionality and heteroskedasticity, cross-sectional data, and longitudinal data with imbalanced observations. In the fourth example, we examine longitudinal data characterized by high dimensionality and heteroskedasticity, where we apply a screening process to enhance computational efficiency for routine usage. The goal of these analyses is to identify important covariates and estimate the corresponding coefficients for conditional quantiles or the mean. The longitudinal nature of the response will also be incorporated if a dataset contains repeated measurements.

5.3.1 Example 1: Longitudinal Data with Heteroskedasticity

In this first example, we generate 100 participants, each with 10 observations, where the 200 covariates, $\mathbf{X}_i$, are each drawn from a normal distribution, $N(0, 1)$ using the `generate_data()` function provided by *geeVerse*. The longitudinal response for participant i is simulated from

$$\mathbf{Y}_i = \mathbf{X}_i\boldsymbol{\beta} + (1 + |X_1|)\epsilon_i. \tag{3}$$

Here the error term ϵ_i is simulated from a multivariate normal distribution $MVN(\mathbf{0}, \boldsymbol{\Sigma})$, where $\boldsymbol{\Sigma}$ has an exchangeable correlation structure with $\rho = 0.6$ for each participant's repeated observations. Observations of the same participant are assumed to be correlated, but observations between different participants are not correlated. The true coefficients are $\boldsymbol{\beta} = (1, \ldots, 1, 0, \ldots, 0)$, where, without loss of generality, the first 7 coefficients are 1 and the remaining coefficients are 0. This simulated dataset exhibits heteroskedasticity and the important variables may vary at different quantiles, a phenomenon frequently observed in real-world datasets.

```
set.seed(2024)
sim_data = generate_data(nsub = 100, nobs = rep(10, 100), p = 200,
                         beta0 = c(rep(1, 7), rep(0, 193)),
                         rho = 0.6, corstr = "exchangeable")
```

The argument `nsub` and `nobs` represent the number of subjects and the number of repeated measurements for each subject respectively. This argument accommodates both balanced data, where all elements are equal, and imbalanced data, where elements may differ. `p` refers to the total number of covariates created, `beta0` is the true coefficient vector for the important variables,

`rho` is the correlation parameter, and `corstr` is the true correlation structure. This function returns an R `data.frame` that contains response variable `y`, covariates `x` and subject identifier `id`.

We use the `qpgee()` function to perform our main analysis, applying quantile penalized GEE to simultaneously select important variables and estimate their corresponding parameters. Within `qpgee()`, three main steps occur. First, we calculate initial values using non-penalized quantile regression for cross-sectional data by implementing the `rq()` function from the *quantreg* package, where the goal is to estimate initial coefficients for a given conditional quantile level τ. Users can also provide their own initial values through the optional `betaint` argument. Second, `qpgee()` finds the optimal penalty parameter for the penalty function by utilizing HBIC from Zu et al. (2023) as discussed in the Supplementary Material or via cross-validation. Users can also provide a grid of penalty parameters to try through the optional `lambda` argument. Third, a final quantile penalized GEE model is fitted using this optimal penalty parameter.

Here, we demonstrate the usage of `qpgee()` with $\tau = 0.9$ and an exchangeable working correlation structure, `'exchangeable'`, but other common working correlation structure options are available in the package including AR1 and independence. The results from all three common working correlation structures are shown in Table 2. Alternatively, the user can provide a working correlation matrix of their own. In this example, we focus on the 90th conditional quantile, $\tau = 0.9$. We also provide the results in Table 2 for the 10th and 50th conditional quantiles.

```
qpgee_sim_fit = qpgee(y ~ 0 + . -id, data = sim_data, id = sim_data$id,
                      corstr = "exchangeable", tau = 0.9, ncore = 10,
                      control = qpgeeControl(shrinkCutoff = 10^-1))
summary(qpgee_sim_fit)
```

Given the page limit, we omit the output of `summary()` here. Instead, we use the `compile_result` function, which compares the results from the `qpgee()` function to the true important variables and true coefficient values from the data generating process described earlier. The `compile_result` function provides performance metrics including the F_1 score, calculated as $F_1 = 2\text{TP}/(2\text{TP} + \text{FP} + \text{FN})$, percentage of times the correct number of important variables are selected (correct %), number of true positives (TP), and the number of false positives (FP). Additionally, MSE is the mean squared error of the parameter estimate, $||\widehat{\boldsymbol{\beta}} - \boldsymbol{\beta}_0||_2^2$, and MAD is the mean absolute deviation $||\widehat{\boldsymbol{\beta}} - \boldsymbol{\beta}_0||_1$.

```
compile_result(qpgee_sim_fit, beta0 = c(rep(1, 7), rep(0, 193)))
```

```
##          F1    correct%          TP          FP         MSE         MAD
## 1.000000000 1.000000000 7.000000000 0.000000000 0.009150872 0.208743723
```

Table 2 provides the results for this example for multiple conditional quantile levels and working correlation structures. This table shows that variable selection performs well for each quantile level and different working correlation structures, since the F_1 score is 1 for all variations. Further, the small values of MSE and MAD indicate that the estimated coefficients are close to the true coefficient values. The estimation performance is slightly better when the working correlation structure matches the true correlation structure, since the MSE and MAD results of the exchangeable correlation are the smallest.

Please note that all data and results presented in this section were generated from a single run, with a random seed of 2024 for reproducibility.

Table 2: Variable selection and estimation results for longitudinal data with heteroskedasticity for Example 1. $F_1 = 2\text{TP}/(2\text{TP}+\text{FP}+\text{FN})$, where TP is the number of true positives, FP is the number of false positives. MSE is the mean squared error of the parameter estimate, $||\widehat{\boldsymbol{\beta}} - \boldsymbol{\beta}_0||_2^2$, and MAD is the mean absolute deviation $||\widehat{\boldsymbol{\beta}} - \boldsymbol{\beta}_0||_1$.

	Independence			Exchangeable			AR1		
	F_1 score	MSE	MAD	F_1 score	MSE	MAD	F_1 score	MSE	MAD
$\tau = 0.1$	1.00	0.01	0.26	1.00	0.01	0.16	1.00	0.01	0.22
$\tau = 0.5$	1.00	0.01	0.23	1.00	0.01	0.16	1.00	0.00	0.21
$\tau = 0.9$	1.00	0.02	0.29	1.00	0.01	0.18	1.00	0.01	0.23

Table 3: Variable selection and estimation results for high-dimensional cross-sectional data for Example 2.

	F_1 score	MSE	MAD
$\tau = 0.1$	1.00	0.0197	0.3168
$\tau = 0.5$	1.00	0.0162	0.3094
$\tau = 0.9$	1.00	0.0273	0.3678

5.3.2 Example 2: Cross-Sectional Data

Our package can also seamlessly handle cross-sectional data. To illustrate, we generate the cross-sectional data similar to Example 1 but with 1000 participants, each participant with one observation. That is, the number of observations for all participants, `nobs`, should be a vector of ones of length 1000. Also, the user should set the working correlation matrix to the independence matrix, `"independence"`, for cross-sectional data.

The code to implement cross-sectional data analysis is

```
set.seed(2024)
sim_data = generate_data(nsub = 1000, nobs = rep(1, 1000),  p = 200,
                         beta0 = c(rep(1, 7), rep(0, 193)),
                         rho = 0.6, corstr = "independence")
```

```
cs_qpgee <- qpgee(y ~ 0 + . -id, data = sim_data, id = sim_data$id,
                  corstr = "independence", tau = 0.9, ncore = 10)
```

Table 3 shows the results using our package to analyze cross-sectional high-dimensional data. This table suggests that the model continues to perform variable selection and estimation well, since the F_1 score is 1 at all conditional quantile levels and the MSE and MAD are small.

5.3.3 Example 3: Imbalanced Data

Frequently, longitudinal data is imbalanced, with participants possibly having varying numbers of observations. Imbalanced observations may occur from participant dropout over time or from difficulties in capturing data at every time point. Our package can easily handle imbalanced data via the argument `id`.

Table 4: Variable selection and estimation results for longitudinal data with heteroskedasticity when the number of observations is imbalanced from Example 3.

	Independence			Exchangeable			AR1		
	F_1 score	MSE	MAD	F_1 score	MSE	MAD	F_1 score	MSE	MAD
$\tau = 0.1$	1.00	0.01	0.18	1.00	0.01	0.19	1.00	0.01	0.19
$\tau = 0.5$	1.00	0.01	0.19	1.00	0.01	0.18	1.00	0.01	0.15
$\tau = 0.9$	1.00	0.02	0.32	1.00	0.01	0.19	1.00	0.00	0.14

In this example, we demonstrate the performance of our approach when the longitudinal data is imbalanced. We let the first half of the participants have 10 observations and the last half of the participants have only 5 observations. Therefore, we create the vector, `nobs`, where 10 is repeated for half the number of participants and 5 is repeated for the other half.

```
set.seed(2024)
sim_data = generate_data(nsub = 100, nobs = c(rep(10, 100/2), rep(5, 100/2)),
                         p = 200, beta0 = c(rep(1, 7), rep(0, 193)),
                         rho = 0.6, corstr = "exchangeable")
```

We use the same code as in Example 1 as `id` does not has to be the of the same length for each subject.

```
qpgee(y ~ . - id, data = sim_data, id = sim_data$id, corstr = "exchangeable",
      tau = 0.9, control = qpgeeControl(shrinkCutoff = 10^-1))
```

Table 4 shows the results when analyzing imbalanced observations among participants for three conditional quantile levels, $\tau = 0.1, 0.5, 0.9$. These results suggest variable selection performance remains high, since the F_1 score is near 1 for all models. Similarly, the estimated parameters are near the true parameters for this example, because the MSE and MAD remain small.

5.3.4 Example 4: Ultra-High-Dimensional Data with Screening Option

Often when data is ultra-high-dimensional, computational times can be prohibitive for routine usage. In this case, we employ a two-step approach, where we first reduce the dimension of the ultra-high-dimensional data then fit the penalized quantile GEE model. One popular screening approach is sure independence screening using the R package *SIS* (Saldana and Feng, 2018) based on the seminal work of Fan and Lv (2008). We simulate the data with $p = 1000$ predictors as below.

```
set.seed(2024)
sim_data = generate_data(nsub = 100, nobs = rep(10, 100), p = 1000,
                         beta0 = c(rep(1, 7), rep(0, 993)),
                         rho = 0.6, corstr = "exchangeable")
```

First, we screen down the data to a reduced dimension using `SIS()` from *SIS*, as shown in the code below, where `nsis` specify the number of variables to retain.

```
if (!require("SIS")) install.packages("SIS");
x_sis_ind = SIS::SIS(x = as.matrix(sim_data[, -c(1, 2)]),
                     y = sim_data$y, nsis = 200)$sis.ix0
```

Second, using the screened variable set `x_sis_ind` obtained from `SIS()`, we apply the `qpgee()` function to perform penalized variable selection and estimation for quantile regression. We use the HBIC approach to find the best-performing penalty parameter. Here we are interested in the conditional median and use the exchangeable working correlation structure.

```
qpgee_fit_hd = qpgee(y ~ 0 + . - id, data = sim_data[, c(1, 2, 2+x_sis_ind)],
                     id = sim_data$id, corstr = "exchangeable", tau = 0.5,
                     ncore = 10)
```

When evaluating the performance with screened variables, we map the fitted coefficients back to the original length of 1000 and then utilize `compile_result` function:

```
qpgee_fit_hd$coefficients <- replace(vector("numeric", 1000), x_sis_ind,
                                     qpgee_fit_hd$coefficients)
compile_result(qpgee_fit_hd, beta0 = c(rep(1, 7), rep(0, 993)))
```

```
##          F1   correct%         TP         FP        MSE        MAD
## 1.00000000 1.00000000 7.00000000 0.00000000 0.06933747 0.66269510
```

5.4 Computational Time Comparisons

geeVerse is the only package available to implement the recent work in Zu et al. (2023) for conditional quantile regression with simultaneous variable selection and estimation for ultra-high-dimensional longitudinal data using the main function `qpgee()`. We also make computational efficiency improvements to previous implementations of the mean penalized generalized estimating equations package, *PGEE*. Reducing computational time is important for practical usage, since variable selection often involves high-dimensional data, whereby computational burden can quickly escalate. See Section 4.3 for more details on these specific computational improvements.

To compare the computational time, we provide a typical run time for each method with the same fixed penalty level on Example 1 simulation data which includes 200 subjects, each with 10 observations, with covariate dimensions $p = 200$ and $p = 1000$. The typical run time was generated using an Apple MacBook Pro (2023) equipped with a 10-core M1 Max chip running 3.2GHz and 32GB unified memory with macOS Sonoma.

As shown in Table 5, we make marked improvements to execution times for the mean `pgee()`. For the quantile `qpgee()` approach, our package is the only option and we demonstrate with $\tau = 0.5$. The implementation of `qpgee()` is surprisingly very efficient, and the run times are even faster than the mean `pgee()` implementations. This efficiency is partly due to `qpgee()` leveraging sparse matrix computation whenever any coefficients shrink to zero, significantly reducing the time needed. In contrast, mean `pgee()` was implemented in the same procedure in the package *PGEE* where sparse computation is not easily achieved. See the Supplementary Material for other details.

Table 5: Computation time comparison of the `pgee()` function in *geeVerse* and *PGEE* packages. Execution time is measured in seconds. Computation time of `qpgee()` is also reported. To the best of our knowledge, there is no alternative package for `qpgee()`. The typical run time was generated using an apple MacBook pro (2023) equipped with a 10-core M1 max chip running 3.2GHz and 32GB unified memory with macOS sonoma.

Scenario	Package	Function	Typical Run Time
$p = 200$	*geeVerse*	`qpgee()`	4.40
	geeVerse	`pgee()`	4.55
	PGEE	`pgee()`	20.32
$p = 1000$	*geeVerse*	`qpgee()`	48.85
	geeVerse	`pgee()`	127.85
	PGEE	`pgee()`	468.39

6 Discussion

In this paper, we provide the R package, *geeVerse*, that allows simultaneous variable selection and estimation for conditional quantile and mean regression, and potentially imbalanced longitudinal data. We adopt penalized generalized estimating equations for both conditional quantile and mean models at high dimensions. It flexibly selects important variables from complex, high-dimensional data without requiring other variable selection packages. This is currently the only available R package for simultaneous variable selection and estimation methodology for quantile generalized estimating equations, and it provides a computationally accelerated implementation of the mean penalized generalized estimating equations.

While *geeVerse* offers robust functionality for handling imbalanced longitudinal data where entire time points may be missing, it does not handle missing values within covariates and does not provide automatic imputation capabilities. Though we have implemented C++ acceleration for key computational steps such as matrix inversions, full C++ implementation of the model fitting process remains a potential area for future development, albeit with potentially modest incremental gains in computational efficiency.

Post-selection inference, the process of computing valid confidence intervals or p-values after data-driven variable selection, is a well-known challenge in high-dimensional statistics. Similar to widely used packages like *glmnet* and *PGEE*, *geeVerse* does not currently implement these formal methods. While practical approaches like the bootstrap (Efron and Tibshirani, 1994) on subject-level, used in Zu et al. (2023), can be easily implemented, recent statistical research has produced more direct techniques. Some examples include selective inference, which provides exact inference by conditioning on the selected model (Tibshirani et al., 2016), and the de-biased LASSO, which adjusts estimators to form valid confidence intervals (van de Geer et al., 2014; Javanmard and Montanari, 2014). Extending these formal frameworks to penalized GEE for ultra-high-dimensional longitudinal data is a non-trivial task and remains an important direction for future research and package development.

For future extensions, we hope to address these limitations and include additional features such as missing data imputation methods and post-selection inference procedures. We also aim to incorporate semiparametric modeling approaches such as partially linear single-index models for longitudinal data. These enhancements would further extend the package's utility in handling real-world data complexities while maintaining its computational efficiency.

Supplementary Material

We provide supplementary material with a detailed model and an explanation of the computational algorithms used in *geeVerse*. We also provide the replication script for this paper at a public repository: Github.

References

1000 Genomes Project Consortium (2015). A global reference for human genetic variation. *Nature*, 526(7571): 68–74. https://doi.org/10.1038/nature15393

Bates DM, Watts DG (1988). *Nonlinear Regression Analysis and Its Applications. Wiley Series in Probability and Mathematical Statistics*. John Wiley & Sons, New York.

Carey VJ, Lumley TS, Moler C, Ripley B (2023). gee: Generalized estimation equation solver. R Package version 4.13–26.

Deshpande V, Deshpande MV (2023). pgee.mixed: Penalized generalized estimating equations for bivariate mixed outcomes. R Package version 0.1.0.

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

Efron B, Tibshirani RJ (1994). *An Introduction to the Bootstrap*. Chapman and Hall/CRC, Boca Raton.

Evangelou E, Warren HR, Mosen-Ansorena D, Mifsud B, Pazoki R, Gao H, et al. (2018). Genetic analysis of over 1 million people identifies 535 new loci associated with blood pressure traits. *Nature Genetics*, 50(10): 1412–1425. https://doi.org/10.1038/s41588-018-0205-x

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

Fan J, Lv J (2008). Sure independence screening for ultrahigh dimensional feature space. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 70(5): 849–911. https://doi.org/10.1111/j.1467-9868.2008.00674.x

Højsgaard S, Halekoh U, Yan J (2006). The R package geepack for generalized estimating equations. *Journal of Statistical Software*, 15: 1–11.

Inan G, Wang L (2017). PGEE: An R package for analysis of longitudinal data with high-dimensional covariates. *R Journal*, 9(1): 393–402.

Javanmard A, Montanari A (2014). Confidence intervals and hypothesis testing for high-dimensional statistical models. *Journal of Machine Learning Research*, 15: 2869–2909.

Koenker R (2025). quantreg: Quantile Regression. R package version 6.1.

Koenker R, Bassett Jr G (1978). Regression quantiles. *Econometrica: Journal of the Econometric Society*, 46: 33–50. https://doi.org/10.2307/1913643

Li Y, Gao X, Xu W (2023). LassoGEE: High-dimensional lasso generalized estimating equations. R Package version 1.0.

Liang KY, Zeger SL (1986). Longitudinal data analysis using generalized linear models. *Biometrika*, 73(1): 13–22. https://doi.org/10.1093/biomet/73.1.13

McDaniel LS, Henderson NC, Rathouz PJ (2013). Fast pure R implementation of GEE: Application of the matrix package. *The R journal*, 5(1): 181–187. https://doi.org/10.32614/RJ-2013-017

National Center for Biotechnology Information (2025). Framingham Heart Study. https://www.ncbi.nlm.nih.gov/projects/gap/cgi-bin/study.cgi?study_id=phs000007.v35.p16. dbGaP Study Accession: phs000007.v35.p16. [Accessed: 2025-08-18].

Nocedal J, Wright SJ (2006). *Numerical Optimization. Springer Series in Operations Research and Financial Engineering*. Springer, New York, 2nd edition.

Rizopoulos D (2010). JM: An R package for the joint modelling of longitudinal and time-to-event data. *Journal of Statistical Software*, 35(9): 1–33. https://doi.org/10.18637/jss.v035.i09

Saldana DF, Feng Y (2018). SIS: An R package for sure independence screening in ultrahigh-dimensional statistical models. *Journal of Statistical Software*, 83(2): 1–25. https://doi.org/10.18637/jss.v083.i02

Sherwood B, Maidman A, Li S (2023). rqPen: Penalized quantile regression. Version 4.1.

Spellman PT, Sherlock G, Zhang MQ, Iyer VR, Anders K, Eisen MB, et al. (1998). Comprehensive identification of cell cycle–regulated genes of the yeast saccharomyces cerevisiae by microarray hybridization. *Molecular Biology of the Cell*, 9(12): 3273–3297. https://doi.org/10.1091/mbc.9.12.3273

Su Z, Marchini J, Donnelly P (2011). HAPGEN2: Simulation of multiple disease SNPs. *Bioinformatics* (Oxford, England), 27(16): 2304–2305.

Tibshirani R (1996). Regression shrinkage and selection via the Lasso. *Journal of the Royal Statistical Society Series B-Methodological*, 58(1): 267–288. https://doi.org/10.1111/j.2517-6161.1996.tb02080.x

Tibshirani RJ, Taylor J, Lockhart R, Tibshirani RJ (2016). Exact post-selection inference for sequential regression procedures. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 78(1): 127–158. https://doi.org/10.1111/rssb.12107

van de Geer S, Bühlmann P, Ritov Y, Dezeure R (2014). On asymptotically optimal confidence regions and tests for high-dimensional models. *Annals of Statistics*, 42(3): 1166–1202.

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

Wang L, Zhou J, Qu A (2012). Penalized generalized estimating equations for high-dimensional longitudinal data analysis. *Biometrics*, 68(2): 353–360. https://doi.org/10.1111/j.1541-0420.2011.01678.x

Zu T, Lian H, Green B, Yu Y (2023). Ultra-high dimensional quantile regression for longitudinal data: An application to blood pressure analysis. *Journal of the American Statistical Association*, 118(541): 97–108. https://doi.org/10.1080/01621459.2022.2128806