SIHR: Statistical Inference in High-Dimensional Linear and Logistic Regression Models

by Prabrisha Rakshit, Zhenyu Wang, Tony Cai, and Zijian Guo

Abstract We introduce the R package SIHR for statistical inference in high-dimensional generalized linear models with continuous and binary outcomes. The package provides functionalities for constructing confidence intervals and performing hypothesis tests for low-dimensional objectives in both one-sample and two-sample regression settings. We illustrate the usage of SIHR through simulated examples and present real data applications to demonstrate the package's performance and practicality.

1 Introduction

In many applications, it is common to encounter regression problems where the number of covariates p exceeds the sample size n. Much progress has been made in point estimation and support recovery in high-dimensional generalized linear models (GLMs), as evidenced by works such as Bühlmann and van de Geer (2011); Negahban et al. (2009); Huang and Zhang (2012); Tibshirani (1996); Fan and Li (2011); Zhang (2010); Sun and Zhang (2012); Belloni et al. (2011); Meinshausen and Yu (2009). In addition to estimation, van de Geer et al. (2014); Javanmard and Montanari (2014); Zhang and Zhang (2014) have proposed methods to correct the bias of penalized regression estimators and construct confidence intervals (CIs) for individual regression coefficients of the high-dimensional linear model. This debiased approach has sparked a rapidly growing research area focused on CI construction and hypothesis testing for low-dimensional objectives in high-dimensional GLMs.

The current paper presents the R package **SIHR**, which constructs confidence intervals and conducts hypothesis testing for various transformations of high-dimensional regression parameters for both continuous and binary outcomes. We consider the high-dimensional GLMs: for $1 \leq i \leq n$,

$$\mathbb{E}(y_i \mid X_{i\cdot}) = f(X_{i\cdot}^{\intercal}\beta), \quad \text{with } f(z) = \begin{cases} z & \text{for linear model;} \\ \exp(z)/\left[1+\exp(z)\right] & \text{for logistic model;} \end{cases} \tag{1}$$

where $y_i \in \mathbb{R}$ and $X_{i\cdot} \in \mathbb{R}^p$ denote respectively the outcome and the measured covariates of the i-th observation and $\beta \in \mathbb{R}^p$ denotes the high-dimensional regression vector. Throughout the paper, define $\Sigma = \mathbb{E}X_{i\cdot}X_{i\cdot}^{\intercal}$ and assume β to be a sparse vector with its sparsity level denoted as $\|\beta\|_0$. In addition to the one-sample setting, we examine the statistical inference methods for the following two-sample regression models,

$$\mathbb{E}(y_i^{(k)} \mid X_{i\cdot}^{(k)}) = f(X_{i\cdot}^{(k)\intercal}\beta^{(k)}) \quad \text{with } k = 1,2 \text{ and } 1 \leq i \leq n_k, \tag{2}$$

where $y_i^{(k)} \in \mathbb{R}$ and $X_{i\cdot}^{(k)} \in \mathbb{R}^p$ denote respectively the outcome and the measured covariates in the k-th sample, $f(\cdot)$ is the pre-specified link function defined as in (1), and $\beta^{(k)} \in \mathbb{R}^p$ denotes the high-dimensional regression vector in k-th sample.

The R package **SIHR** consists of five main functions `LF()`, `QF()`, `CATE()`, `InnProd()`, and `Dist()` implementing the statistical inferences for five different quantities correspondingly.

1. `LF()`, abbreviated for linear functional, implements the inference approach for $x_{\text{new}}^{\intercal}\beta$ in Cai et al. (2021a,b), with $x_{\text{new}} \in \mathbb{R}^p$ denoting a loading vector. With $x_{\text{new}} = e_j$ as a special case, `LF()` infers about the regression coefficient β_j (van de Geer et al., 2014; Javanmard and Montanari, 2014; Zhang and Zhang, 2014, e.g.). When x_{new} denotes a future observation's covariates, `LF()` makes inference for the conditional mean of the outcome for the individual. See the usage of `LF()` in the section Linear functional.

2. `QF()`, abbreviated for quadratic functional, makes inference for $\beta_{\text{G}}^{\intercal}A\beta_{\text{G}}$, following the proposal in Guo et al. (2019, 2021b); Cai and Guo (2020). β_{G} is the subvector of β with indices restricted to the pre-specified index set $\text{G} \in \{1,\ldots,p\}$ and $A \in \mathbb{R}^{|\text{G}|\times|\text{G}|}$, with $|\text{G}|$ denoting cardinality of G, is either a pre-specified submatrix or the unknown $\Sigma_{\text{G},\text{G}}$. $\beta_{\text{G}}^{\intercal}A\beta_{\text{G}}$ can be viewed as a total measure of effects of all the variables in the group G. See the section Quadratic functional for the usage.

3. `CATE()`, abbreviated for conditional average treatment effect, is to make inference for $f(x_{\text{new}}^{\intercal}\beta^{(2)}) - f(x_{\text{new}}^{\intercal}\beta^{(1)})$, see Cai et al. (2021a) for detailed discussion. This difference measures the discrepancy between conditional means, closely related to the conditional average treatment effect for the new observation with covariates x_{new}. We demonstrate its usage in the section Conditional average treatment effect.

4. `InnProd()`, abbreviated for inner product, implements the statistical inference for $\beta_{G}^{(1)\intercal} A \beta_{G}^{(2)}$ with $A \in R^{|G|\times|G|}$, which was proposed in Guo et al. (2019); Ma et al. (2022). The inner product measures the similarity between the high-dimensional vectors $\beta^{(1)}$ and $\beta^{(2)}$, which is useful in capturing the genetic relatedness in the GWAS applications (Guo et al., 2019; Ma et al., 2022). The usage is detailed in the section Inner product.

5. `Dist()`, short-handed for distance, makes inference for the weighted distance $\gamma_{G}^{\intercal} A \gamma_{G}$ with $\gamma = \beta^{(2)} - \beta^{(1)}$. The distance measure is useful in comparing different high-dimensional regression vectors and constructing a generalizable model in the multisource learning problem Guo et al. (2023). See the section Distance for its usage.

There are a few other R packages for high-dimensional inference. The packages **hdi** and **SSLasso** (available at `http://web.stanford.edu/~montanar/sslasso/code.html`) implement the coordinate debiased Lasso estimators proposed in van de Geer et al. (2014) and Javanmard and Montanari (2014), respectively. These functions provide debiased estimators of β along with their standard error estimators. These existing packages enable confidence interval construction and hypothesis testing for linear transformations of β, but not the quadratic form or inner products implemented in `QF()`, `InnProd()`, and `Dist()`. Even for the linear transformation, their implementation requires debiasing p regression parameters. In contrast, our R package **SIHR** is computationally more efficient as it directly performs a single debiasing for the pre-specified linear transformation.

The **DoubleML** package focuses on estimating low-dimensional parameters of interest, such as causal or treatment effect parameters, in the presence of high-dimensional nuisance parameters that can be estimated using machine learning methods, while our package aims to estimate arbitrary linear and weighted quadratic combinations of the coefficient vector in high-dimensional regression. The selective inference is implemented by the R package **selectiveInference**. They focus on parameters based on the selected model, while we focus on fixed parameters independent of the selected models.

In the remainder of this paper, we review the inference methods in Section Methodological background and introduce the main functions of the package in Section Usage of the package, accompanied by illustrative examples. Then, a comparative analysis is conducted in Section Comparative analysis. Finally, we demonstrate the application of our proposed methods to real data in Section Real data applications.

2 Methodological background

We briefly review the penalized maximum likelihood estimator of β in the high-dimensional GLM (1), defined as:

$$\widehat{\beta} = \arg\min_{\beta\in\mathbb{R}^p} \ell(\beta) + \lambda \sum_{j=2}^{p} \frac{\|X_{\cdot j}\|_2}{\sqrt{n}} |\beta_j|, \tag{3}$$

with $X_{\cdot j}$ denoting the j-th column of X, and

$$\ell(\beta) = \begin{cases} \frac{1}{n}\sum_{i=1}\left(y_i - X_{i\cdot}^{\intercal}\beta\right)^2 & \text{for linear model} \\ -\frac{1}{n}\sum_{i=1}^{n} y_i \log\left[\frac{f(X_{i\cdot}^{\intercal}\beta)}{1-f(X_{i\cdot}^{\intercal}\beta)}\right] - \frac{1}{n}\sum_{i=1}^{n}\log\left(1 - f(X_{i\cdot}^{\intercal}\beta)\right) & \text{for GLM with binary outcome.} \end{cases} \tag{4}$$

To facilitate the methodological discussion, we take the first column of X set as the constant 1 and hence does not include a penalty on β_1 in the above equation (3). In the penalized regression (3), we do not penalize the intercept coefficient β_1 and the tuning parameter $\lambda \asymp \sqrt{\log p/n}$ is chosen by cross-validation. The penalized estimators have been shown to achieve the optimal convergence rates and satisfy desirable variable selection properties (Meinshausen and Bühlmann, 2006; Bickel et al., 2009; Zhao and Yu, 2006; Wainwright, 2009). However, these estimators are not ready for statistical inference due to the non-negligible estimation bias induced by the penalty term (van de Geer et al., 2014; Javanmard and Montanari, 2014; Zhang and Zhang, 2014).

In section Linear functional for GLM, we propose a unified inference method for $x_{\text{new}}^{\intercal}\beta$ under linear and logistic outcome models. We also discuss inferences for quadratic functionals $\beta_{G}^{\intercal} A \beta_{G}$ and $\beta_{G}^{\intercal}\Sigma_{G,G}\beta_{G}$ in section Quadratic functional for GLM. In the case of the two-sample high-dimensional regression model (2), we develop the inference method for conditional treatment effect $\Delta(x_{\text{new}}) =$

$f(x_{\text{new}}^{\intercal}\beta^{(2)}) - f(x_{\text{new}}^{\intercal}\beta^{(1)})$ in section Conditional average treatment effects; we consider inference for $\beta_{\text{G}}^{(1)\intercal} A \beta_{\text{G}}^{(2)}$ and $\beta_{\text{G}}^{(1)\intercal}\Sigma_{\text{G,G}}\beta_{\text{G}}^{(2)}$ in section Inner product of regression vectors and $\gamma_{\text{G}}^{\intercal} A \gamma_{\text{G}}$ and $\gamma_{\text{G}}^{\intercal}\Sigma_{\text{G,G}}\gamma_{\text{G}}$ with $\gamma = \beta^{(2)} - \beta^{(1)}$ in section Distance of regression vectors.

2.1 Linear functional for linear model

To illustrate the main idea, we start with the linear functional for the linear model, which will be extended to a unified version in the section Linear functional for GLM. For the linear model in (1), we define $\epsilon_i = y_i - X_{i\cdot}^{\intercal}\beta$ and rewrite the model as $y_i = X_{i\cdot}^{\intercal}\beta + \epsilon_i$ for $1 \le i \le n$.

Given the vector $x_{\text{new}} \in \mathbb{R}^p$, a natural idea for the point estimator is to use the plug-in estimator $x_{\text{new}}^{\intercal}\widehat{\beta}$ with the initial estimator $\widehat{\beta}$ defined in (3). However, the bias $x_{\text{new}}^{\intercal}(\widehat{\beta} - \beta)$ is not negligible. The work Cai et al. (2021a) proposed the bias-corrected estimator as,

$$\widehat{x_{\text{new}}^{\intercal}\beta} = x_{\text{new}}^{\intercal}\widehat{\beta} + \widehat{u}^{\intercal}\frac{1}{n}\sum_{i=1}^{n} X_{i\cdot}\left(y_i - X_{i\cdot}^{\intercal}\widehat{\beta}\right), \tag{5}$$

where the second term on the right hand side in (5) is the estimate of negative bias $-x_{\text{new}}^{\intercal}(\widehat{\beta} - \beta)$, and the projection direction $\widehat{u}$ is defined as

$$\widehat{u} = \arg\min_{u \in \mathbb{R}^p} u^{\intercal}\widehat{\Sigma}u \quad \text{subject to: } \|\widehat{\Sigma}u - x_{\text{new}}\|_{\infty} \le \|x_{\text{new}}\|_2 \mu_0 \tag{6}$$

$$\left| x_{\text{new}}^{\intercal}\widehat{\Sigma}u - \|x_{\text{new}}\|_2^2 \right| \le \|x_{\text{new}}\|_2^2 \mu_0, \tag{7}$$

where $\widehat{\Sigma} = \frac{1}{n}\sum_{i=1}^{n} X_{i\cdot}X_{i\cdot}^{\intercal}$ and $\mu_0 \asymp \sqrt{\log p/n}$. The bias-corrected estimator $\widehat{x_{\text{new}}^{\intercal}\beta}$ satisfies the following error decomposition,

$$\widehat{x_{\text{new}}^{\intercal}\beta} - x_{\text{new}}^{\intercal}\beta = \underbrace{\widehat{u}^{\intercal}\frac{1}{n}\sum_{i=1}^{n} X_{i\cdot}^{\intercal}\epsilon_i}_{\text{asymp. normal}} + \underbrace{\left(\widehat{\Sigma}\widehat{u} - x_{\text{new}}\right)^{\intercal}(\beta - \widehat{\beta})}_{\text{remaining bias}}. \tag{8}$$

The constrained optimization problem in (6) and (7) is designed to minimize the error on the right-hand side of the above equation: the first constraint in (6) controls the "remaining bias" term in the above equation while the objective function in (6) is used to minimize the variance of the "asymp. normal" term. Importantly, the second constraint in (7) ensures the standard error of the "asymp. normal" term always dominates the "remaining bias" term. Based on the asymptotic normality, we construct the CI for $x_{\text{new}}^{\intercal}\beta$ as

$$\text{CI} = \left(\widehat{x_{\text{new}}^{\intercal}\beta} - z_{\alpha/2}\sqrt{\widehat{\text{V}}},\quad \widehat{x_{\text{new}}^{\intercal}\beta} + z_{\alpha/2}\sqrt{\widehat{\text{V}}}\right) \quad \text{with } \widehat{\text{V}} = \frac{\widehat{\sigma}^2}{n}\widehat{u}^{\intercal}\widehat{\Sigma}\widehat{u},$$

where $\widehat{\sigma}^2 = \frac{1}{n}\sum_{i=1}^{n}(y_i - X_{i\cdot}^{\intercal}\widehat{\beta})^2$, and $z_{\alpha/2}$ denotes the upper $\alpha/2$ quantile for the standard normal distribution.

Remark 1 *It has been shown in Cai et al. (2021a) that the remaining bias term in (8) becomes negligible in comparison to the variance of the asymptotic normal term when the sample size is relatively large. However, for applications with a given sample size, we may also enlarge the standard error by a certain factor (e.g., 1.1) to accommodate the bias component in (8).*

2.2 Linear functional for GLM

In this subsection, we generalize the inference method specifically for the linear model in Linear functional for linear model to GLM in (1). Given the initial estimator $\widehat{\beta}$ defined in (3), the key step is to estimate the bias $x_{\text{new}}^{\intercal}(\widehat{\beta} - \beta)$. We can propose a generalized version of the bias-corrected estimator for $x_{\text{new}}^{\intercal}\beta$ as

$$\widehat{x_{\text{new}}^{\intercal}\beta} = x_{\text{new}}^{\intercal}\widehat{\beta} + \widehat{u}^{\intercal}\frac{1}{n}\sum_{i=1}^{n}\omega(X_{i\cdot}^{\intercal}\widehat{\beta})\left(y_i - f(X_{i\cdot}^{\intercal}\widehat{\beta})\right)X_{i\cdot}, \tag{9}$$

where the projection direction $\widehat{u}$ is defined in the following (10) and $\omega : \mathbb{R} \to \mathbb{R}$ denotes a weight function specified in the following Table 1 associated with different link functions.

In Table 1, we consider different GLM models and present the link function $f(\cdot)$, its derivative $f'(\cdot)$, and the corresponding weight function $\omega(\cdot)$. Note that there are two ways of specifying the weights $w(z)$ for logistic regression, where the linearization weighting was proposed in Guo et al.

Model	Outcome Type	$f(z)$	$f'(z)$	$\omega(z)$	Weighting
linear	Continuous	z	1	1	
logistic	Binary	$\frac{e^z}{1+e^z}$	$\frac{e^z}{(1+e^z)^2}$	$\frac{(1+e^z)^2}{e^z}$	Linearization
logistic_alter	Binary	$\frac{e^z}{1+e^z}$	$\frac{e^z}{(1+e^z)^2}$	1	Link-specific

Table 1: Definitions of the functions ω and f for different GLMs.

(2021b) for logistic regression while the link-specific weighting function was proposed in Cai et al. (2021b) for general link function $f(\cdot)$. The projection direction $\widehat{u} \in \mathbb{R}^p$ in (9) is constructed as follows:

$$\widehat{u} = \arg\min_{u\in\mathbb{R}^p} u^\intercal \left[\frac{1}{n}\sum_{i=1}^{n}\omega(X_{i\cdot}^\intercal\widehat{\beta})f'(X_{i\cdot}^\intercal\widehat{\beta})X_{i\cdot}X_{i\cdot}^\intercal\right]u \quad \text{subject to:}$$
$$\left\|\frac{1}{n}\sum_{i=1}^{n}\omega(X_{i\cdot}^\intercal\widehat{\beta})f'(X_{i\cdot}^\intercal\widehat{\beta})X_{i\cdot}X_{i\cdot}^\intercal u - x_{\text{new}}\right\|_\infty \le \|x_{\text{new}}\|_2\mu_0 \tag{10}$$
$$\left|x_{\text{new}}^\intercal\frac{1}{n}\sum_{i=1}^{n}\omega(X_{i\cdot}^\intercal\widehat{\beta})f'(X_{i\cdot}^\intercal\widehat{\beta})X_{i\cdot}X_{i\cdot}^\intercal u - \|x_{\text{new}}\|_2^2\right| \le \|x_{\text{new}}\|_2^2\mu_0.$$

It has been established that $\widehat{x_{\text{new}}^\intercal\beta}$ in (9) is asymptotically unbiased and normal for the linear model (Cai et al., 2021a), the logistic model (Guo et al., 2021a; Cai et al., 2021b). The variance of $\widehat{x_{\text{new}}^\intercal\beta}$ can be estimated by $\widehat{\mathrm{V}}$, defined as

$$\widehat{\mathrm{V}} = \widehat{u}^\intercal\left[\frac{1}{n^2}\sum_{i=1}^{n}\left(\omega(X_{i\cdot}^\intercal\widehat{\beta})\right)^2\widehat{\sigma}_i^2X_{i\cdot}X_{i\cdot}^\intercal\right]\widehat{u} \quad \text{with :} \tag{11}$$

$$\widehat{\sigma}_i^2 = \begin{cases}\frac{1}{n}\sum_{j=1}^{n}\left(y_j - X_{j\cdot}^\intercal\widehat{\beta}\right)^2, & \text{for linear model}\\ f(X_{i\cdot}^\intercal\widehat{\beta})(1-f(X_{i\cdot}^\intercal\widehat{\beta})), & \text{for logistic regression with } f(z)=\exp(z)/[1+\exp(z)]\end{cases}. \tag{12}$$

Based on the asymptotic normality, the CI for $x_{\text{new}}^\intercal\beta$ is:

$$\mathrm{CI} = \left(\widehat{x_{\text{new}}^\intercal\beta} - z_{\alpha/2}\sqrt{\widehat{\mathrm{V}}},\quad \widehat{x_{\text{new}}^\intercal\beta} + z_{\alpha/2}\sqrt{\widehat{\mathrm{V}}}\right).$$

Subsequently, for the binary outcome case, we estimate the case probability $\mathbb{P}(y_i = 1 \mid X_{i\cdot} = x_{\text{new}})$ by $f(\widehat{x_{\text{new}}^\intercal\beta})$ and construct the CI for $f(x_{\text{new}}^\intercal\beta)$, with $f(z)=\exp(z)/[1+\exp(z)]$, as:

$$\mathrm{CI} = \left(f\left(\widehat{x_{\text{new}}^\intercal\beta} - z_{\alpha/2}\sqrt{\widehat{\mathrm{V}}}\right), f\left(\widehat{x_{\text{new}}^\intercal\beta} + z_{\alpha/2}\sqrt{\widehat{\mathrm{V}}}\right)\right).$$

2.3 Quadratic functional for GLM

We now move our focus to inference for the quadratic functional $\mathrm{Q}_A = \beta_G^\intercal A\beta_G$, where $G \subset \{1,\ldots,p\}$ and $A \in \mathbb{R}^{|G|\times|G|}$ denotes a pre-specified matrix of interest. Without loss of generality, we set $G = \{1,2,\cdots,|G|\}$. With the initial estimator $\widehat{\beta}$ defined in (3), the plug-in estimator $\widehat{\beta}_G^\intercal A\widehat{\beta}_G$ has the following estimation error,

$$\widehat{\beta}_G^\intercal A\widehat{\beta}_G - \beta_G^\intercal A\beta_G = 2\widehat{\beta}_G^\intercal A(\widehat{\beta}_G - \beta_G) - (\widehat{\beta}_G - \beta_G)^\intercal A(\widehat{\beta}_G - \beta_G).$$

The last term in the above decomposition $(\widehat{\beta}_G - \beta_G)^\intercal A(\widehat{\beta}_G - \beta_G)$ is the higher-order approximation error under regular conditions; thus the bias of $\widehat{\beta}_G^\intercal A\widehat{\beta}_G$ mainly comes from the term $2\widehat{\beta}_G^\intercal A(\widehat{\beta}_G - \beta_G)$, which can be expressed as $2\,x_{\text{new}}^\intercal(\widehat{\beta} - \beta)$ with $x_{\text{new}} = (\widehat{\beta}_G^\intercal A,\ \mathbf{0})^\intercal$. Hence the term can be estimated directly by applying the linear functional approach in section Linear functional for GLM. Utilizing this idea, Guo et al. (2021b, 2019) proposed the following estimator of Q_A,

$$\widehat{\mathrm{Q}}_A = \widehat{\beta}_G^\intercal A\widehat{\beta}_G + 2\,\widehat{u}_A^\intercal\left[\frac{1}{n}\sum_{i=1}^{n}\omega(X_{i\cdot}^\intercal\widehat{\beta})\left(y_i - f(X_{i\cdot}^\intercal\widehat{\beta})\right)X_{i\cdot}\right], \tag{13}$$

where $\widehat{u}_A$ is the projection direction defined in (10) with $x_{\text{new}} = (\widehat{\beta}_G^\intercal A,\ \mathbf{0}^\intercal)^\intercal$. Since Q_A is non-negative if A is positive semi-definite, we truncate $\widehat{\mathrm{Q}}_A$ at 0 and define $\widehat{\mathrm{Q}}_A = \max\left(\widehat{\mathrm{Q}}_A,\ 0\right)$. We further estimate the variance of the $\widehat{Q}_A$ by

$$\widehat{V}_A(\tau) = 4\widehat{u}_A^{\intercal}\left[\frac{1}{n^2}\sum_{i=1}^{n}\omega^2(X_{i\cdot}^{\intercal}\widehat{\beta})\widehat{\sigma}_i^2 X_{i\cdot}X_{i\cdot}^{\intercal}\right]\widehat{u}_A + \frac{\tau}{n}, \tag{14}$$

where $\widehat{\sigma}_i^2$ is defined in (12) and the term τ/n with $\tau > 0$ (default value $\tau = 1$) is introduced as an upper bound for the term $(\widehat{\beta}_G - \beta_G)^{\intercal}A(\widehat{\beta}_G - \beta_G)$. Then given a fixed value of τ, we construct the CI for Q_A as $\mathrm{CI}(\tau) = \left(\max\left(\widehat{Q}_A - z_{\alpha/2}\sqrt{\widehat{V}_A(\tau)},\ 0\right),\ \widehat{Q}_A + z_{\alpha/2}\sqrt{\widehat{V}_A(\tau)}\right)$.

Now we turn to the estimation of $Q_\Sigma = \beta_G^{\intercal}\Sigma_{G,G}\beta_G$ where the matrix $\Sigma_{G,G}$ is unknown and estimated by $\widehat{\Sigma}_{G,G} = \frac{1}{n}\sum_{i=1}^{n} X_{iG}X_{iG}^{\intercal}$. Decompose the error of the plug-in estimator $\widehat{\beta}_G^{\intercal}\widehat{\Sigma}_{G,G}\widehat{\beta}$:

$$\widehat{\beta}_G^{\intercal}\widehat{\Sigma}_{G,G}\widehat{\beta} - \beta_G\Sigma_{G,G}\beta_G = 2\,\widehat{\beta}_G^{\intercal}\widehat{\Sigma}_{G,G}(\widehat{\beta}_G - \beta_G) + \beta_G^{\intercal}(\widehat{\Sigma}_{G,G} - \Sigma_{G,G})\beta_G - (\widehat{\beta}_G - \beta_G)^{\intercal}\widehat{\Sigma}_{G,G}(\widehat{\beta}_G - \beta_G).$$

The first term $\widehat{\beta}_G^{\intercal}\widehat{\Sigma}_{G,G}(\widehat{\beta}_G - \beta_G)$ is estimated by applying linear functional approach in Linear functional for GLM with $x_{\text{new}} = (\widehat{\beta}_G^{\intercal}\widehat{\Sigma}_{G,G},\ \mathbf{0})^{\intercal}$; the second term $\beta_G^{\intercal}(\widehat{\Sigma}_{G,G} - \Sigma_{G,G})\beta_G$ can be controlled asymptotically by central limit theorem; and the last term $(\widehat{\beta}_G - \beta_G)^{\intercal}\widehat{\Sigma}_{G,G}(\widehat{\beta}_G - \beta_G)$ is negligible due to high-order bias. Guo et al. (2021b) proposed the following estimator of Q_Σ

$$\widehat{Q}_\Sigma = \widehat{\beta}_G^{\intercal}\widehat{\Sigma}_{G,G}\widehat{\beta}_G + 2\,\widehat{u}_\Sigma^{\intercal}\left[\frac{1}{n}\sum_{i=1}^{n}\omega(X_{i\cdot}^{\intercal}\widehat{\beta})\left(y_i - f(X_{i\cdot}^{\intercal}\widehat{\beta})\right)X_{i\cdot}\right],$$

where $\widehat{u}_\Sigma$ is the projection direction constructed in (10) with $x_{\text{new}} = (\widehat{\beta}_G^{\intercal}\widehat{\Sigma}_{G,G},\ \mathbf{0})^{\intercal}$. We introduce the estimator $\widehat{Q}_\Sigma = \max(\widehat{Q}_\Sigma,\ 0)$ and estimate its variance as

$$\widehat{V}_\Sigma(\tau) = 4\widehat{u}_\Sigma^{\intercal}\left[\frac{1}{n^2}\sum_{i=1}^{n}\omega^2(X_{i\cdot}^{\intercal}\widehat{\beta})\widehat{\sigma}_i^2 X_{i\cdot}X_{i\cdot}^{\intercal}\right]\widehat{u}_\Sigma + \frac{1}{n^2}\sum_{i=1}^{n}\left(\widehat{\beta}_G^{\intercal}X_{i,G}X_{i,G}^{\intercal}\widehat{\beta}_G - \widehat{\beta}_G^{\intercal}\widehat{\Sigma}_{G,G}\widehat{\beta}_G\right)^2 + \frac{\tau}{n}, \tag{15}$$

where $\widehat{\sigma}_i^2$ is defined in (12) and the term τ/n with $\tau > 0$ is introduced as an upper bound for the term $(\widehat{\beta}_G - \beta_G)^{\intercal}\widehat{\Sigma}_{G,G}(\widehat{\beta}_G - \beta_G)$. Then, for a fixed value of τ, we can construct the CI for Q_Σ as

$$\mathrm{CI}(\tau) = \left(\max\left(\widehat{Q}_\Sigma - z_{\alpha/2}\sqrt{\widehat{V}_\Sigma(\tau)},\ 0\right),\ \widehat{Q}_\Sigma + z_{\alpha/2}\sqrt{\widehat{V}_\Sigma(\tau)}\right). \tag{16}$$

2.4 Conditional average treatment effects

The inference methods developed for one sample can be generalized to make inferences for conditional average treatment effects (CATE). From a causality viewpoint, we consider the data set $\{(X_{i\cdot}, y_i, D_i)\}$ for $i = 1, \ldots, n$, where $D_i \in \{1, 2\}$ indicates the treatment assigned to the i-th observation. For a new observation with covariates $X_{i\cdot} = x_{\text{new}}$, we define CATE as $\Delta(x_{\text{new}}) = \mathbb{E}(y_i|X_{i\cdot}, D_i = 2) - \mathbb{E}(y_i|X_{i\cdot}, D_i = 1)$.

We group observations $\{i : D_i = k\}$ into the k-th data sample $\{(X_i^{(k)}, y_i^{(k)}\}$ for $k = 1, 2$, where $1 \leq i \leq n_k$ and $n_1 + n_2 = n$. Subsequently, we rewrite $\mathbb{E}(y_i|X_{i\cdot}, D_i = k)$ as $\mathbb{E}[y_i^{(k)}|X_i^{(k)} = x_{\text{new}}]$ for $k = 1, 2$. Using the GLM model outlined in (2), the CATE can be formulated as

$$\Delta(x_{\text{new}}) = \mathbb{E}[y_i^{(2)}|X_i^{(2)} = x_{\text{new}}] - \mathbb{E}[y_i^{(1)}|X_i^{(1)} = x_{\text{new}}] = f(x_{\text{new}}^{\intercal}\beta^{(2)}) - f(x_{\text{new}}^{\intercal}\beta^{(1)}).$$

Following (9), we construct the bias-corrected point estimators of $\widehat{x_{\text{new}}^{\intercal}\beta^{(1)}}$ and $\widehat{x_{\text{new}}^{\intercal}\beta^{(2)}}$, together with their corresponding variances $\widehat{V}^{(1)}$ and $\widehat{V}^{(2)}$ as (11). For the first sample $(X_i^{(1)}, y_i^{(1)})$, where $1 \leq i \leq n_1$, we use the methods described in equations (9) and (11) to compute the bias-corrected point estimator $\widehat{x_{\text{new}}^{\intercal}\beta^{(1)}}$ and the variance estimator $\widehat{V}^{(1)}$, respectively. Similarly, for the second sample $(X_i^{(2)}, y_i^{(2)})$, where $1 \leq i \leq n_2$, we apply the same procedures to derive the point estimator $\widehat{x_{\text{new}}^{\intercal}\beta^{(2)}}$ and the variance estimator $\widehat{V}^{(2)}$.

The paper Cai et al. (2021a) proposed to estimate $\Delta(x_{\text{new}})$ by $\widehat{\Delta}(x_{\text{new}})$ as follows,

$$\widehat{\Delta}(x_{\text{new}}) = f(\widehat{x_{\text{new}}^{\intercal}\beta^{(2)}}) - f(\widehat{x_{\text{new}}^{\intercal}\beta^{(1)}}).$$

Its variance can be estimated with delta method by:

$$\widehat{V}_{\Delta} = \left(f'(\widehat{x_{\text{new}}^{\intercal}\beta^{(1)}})\right)^2 \widehat{V}^{(1)} + \left(f'(\widehat{x_{\text{new}}^{\intercal}\beta^{(2)}})\right)^2 \widehat{V}^{(2)}.$$

Then we construct the CI for $\Delta(x_{\text{new}})$ as

$$\text{CI} = \left(\widehat{\Delta}(x_{\text{new}}) - z_{\alpha/2}\sqrt{\widehat{V}_{\Delta}}, \widehat{\Delta}(x_{\text{new}}) + z_{\alpha/2}\sqrt{\widehat{V}_{\Delta}}\right).$$

2.5 Inner product of regression vectors

The paper Guo et al. (2019); Ma et al. (2022) have investigated the CI construction for $\beta_{G}^{(1)\intercal} A \beta_{G}^{(2)}$, provided with a pre-specified submatrix $A \in \mathbb{R}^{|G| \times |G|}$ and the set of indices $G \subset \{1, \ldots, p\}$. With $\widehat{\beta}^{(1)}$ and $\widehat{\beta}^{(2)}$ denoting the initial estimators fitted on first and second data sample via (3), respectively, the plug-in estimator $\widehat{\beta}_{G}^{(1)\intercal} A \widehat{\beta}_{G}^{(2)}$ admits the following bias,

$$\begin{aligned}\widehat{\beta}_{G}^{(1)\intercal} A \widehat{\beta}_{G}^{(2)} - \beta_{G}^{(1)\intercal} A \beta_{G}^{(2)} = \widehat{\beta}_{G}^{(2)\intercal} A \left(\widehat{\beta}_{G}^{(1)} - \beta_{G}^{(1)}\right) + \widehat{\beta}_{G}^{(1)\intercal} A \left(\widehat{\beta}_{G}^{(2)} - \beta_{G}^{(2)}\right) \\ - \left(\widehat{\beta}_{G}^{(1)} - \beta_{G}^{(1)}\right)^{\intercal} A \left(\widehat{\beta}_{G}^{(2)} - \beta_{G}^{(2)}\right).\end{aligned}$$

The key step is to estimate the components $\widehat{\beta}_{G}^{(2)\intercal} A \left(\widehat{\beta}_{G}^{(1)} - \beta_{G}^{(1)}\right)$ and $\widehat{\beta}_{G}^{(1)\intercal} A \left(\widehat{\beta}_{G}^{(2)} - \beta_{G}^{(2)}\right)$, since the last term $(\widehat{\beta}_{G}^{(1)} - \beta_{G}^{(1)})^{\intercal} A (\widehat{\beta}_{G}^{(2)} - \beta_{G}^{(2)})$ is negligible due to high-order bias. We propose the following bias-corrected estimator for $\beta_{G}^{(1)\intercal} A \beta_{G}^{(2)}$

$$\begin{aligned}\widehat{\beta_{G}^{(1)\intercal} A \beta_{G}^{(2)}} = \widehat{\beta}_{G}^{(1)\intercal} A \widehat{\beta}_{G}^{(2)} + \widehat{u}_1^{\intercal} \frac{1}{n_1} \sum_{i=1}^{n_1} \omega(X_{i\cdot}^{(1)\intercal}\widehat{\beta}^{(1)}) \left(y_i^{(1)} - f(X_{i\cdot}^{(1)\intercal}\widehat{\beta}^{(1)})\right) X_{i\cdot}^{(1)} \\ + \widehat{u}_2^{\intercal} \frac{1}{n_2} \sum_{i=1}^{n_2} \omega(X_{i\cdot}^{(2)\intercal}\widehat{\beta}^{(2)}) \left(y_i^{(2)} - f(X_{i\cdot}^{(2)\intercal}\widehat{\beta}^{(2)})\right) X_{i\cdot}^{(2)}.\end{aligned} \tag{17}$$

Here $\widehat{u}_1$ represents the projection direction computed in (10), using the first sample data and $x_{\text{new}} = (\widehat{\beta}_{G}^{(2)\intercal} A,\ \mathbf{0})^{\intercal}$. Similarly, $\widehat{u}_2$ is the projection direction derived from the second sample data, using $x_{\text{new}} = (\widehat{\beta}_{G}^{(2)\intercal} A,\ \mathbf{0})^{\intercal}$. The corresponding variance of $\widehat{\beta_{G}^{(1)\intercal} A \beta_{G}^{(2)}}$, when A is a known positive definite matrix, is estimated as

$$\widehat{V}_A(\tau) = \widehat{V}^{(1)} + \widehat{V}^{(2)} + \frac{\tau}{\min(n_1, n_2)},$$

where $\widehat{V}^{(k)}$ is computed as in (11) for the k-th regression model ($k = 1, 2$) and the term $\tau/\min(n_1, n_2)$ with $\tau > 0$ is introduced as an upper bound for the term $(\widehat{\beta}_{G}^{(1)} - \beta_{G}^{(1)})^{\intercal} A (\widehat{\beta}_{G}^{(2)} - \beta_{G}^{(2)})$.

We also consider the case of unknown $A = \Sigma_{G,G}$. As a natural generalization, the quantity $\beta_{G}^{(1)\intercal} \Sigma_{G,G} \beta_{G}^{(2)}$ is well defined if the two regression models in (2) share the design covariance matrix $\Sigma = \mathbb{E} X_{i\cdot}^{(1)} X_{i\cdot}^{(1)\intercal} = \mathbb{E} X_{i\cdot}^{(2)} X_{i\cdot}^{(2)\intercal}$. We follow the above procedures by replacing A with $\widehat{\Sigma}_{G,G} = \frac{1}{n_1+n_2}\sum_{i=1}^{n_1+n_2} X_{i,G} X_{i,G}^{\intercal}$ where X is the row-combined matrix of $X^{(1)}$ and $X^{(2)}$. The variance of $\widehat{\beta_{G}^{(1)\intercal} \Sigma_{G,G} \beta_{G}^{(2)}}$ is now estimated as

$$\widehat{V}_{\Sigma}(\tau) = \widehat{V}^{(1)} + \widehat{V}^{(2)} + \frac{1}{(n_1+n_2)^2} \sum_{i=1}^{n_1+n_2} \left(\widehat{\beta}_{G}^{(1)\intercal} X_{i,G} X_{i,G}^{\intercal} \widehat{\beta}_{G}^{(2)} - \widehat{\beta}_{G}^{(1)\intercal} \widehat{\Sigma}_{G,G} \widehat{\beta}_{G}^{(2)}\right)^2 + \frac{\tau}{\min(n_1, n_2)}.$$

We then construct the CI for $\beta_{G}^{(1)\intercal} A \beta_{G}^{(2)}$ as

$$\text{CI}(\tau) = \begin{cases} \left(\widehat{\beta_{G}^{(1)\intercal} A \beta_{G}^{(2)}} - z_{\alpha/2}\widehat{V}_A(\tau),\ \widehat{\beta_{G}^{(1)\intercal} A \beta_{G}^{(2)}} + z_{\alpha/2}\widehat{V}_A(\tau)\right) & \text{if } A \text{ is specified} \\ \left(\widehat{\beta_{G}^{(1)\intercal} \Sigma_{G,G} \beta_{G}^{(2)}} - z_{\alpha/2}\widehat{V}_{\Sigma}(\tau),\ \widehat{\beta_{G}^{(1)\intercal} \Sigma_{G,G} \beta_{G}^{(2)}} + z_{\alpha/2}\widehat{V}_{\Sigma}(\tau)\right) & A = \Sigma_{G,G} \text{ is unknown.} \end{cases}$$

2.6 Distance of regression vectors

We denote $\gamma = \beta^{(2)} - \beta^{(1)}$ and its initial estimator $\widehat{\gamma} = \widehat{\beta}^{(2)} - \widehat{\beta}^{(1)}$. The quantity of interest is the distance between two regression vectors $\gamma_{\mathrm{G}}^{\intercal} A \gamma_{\mathrm{G}}$, given a pre-specified submatrix $A \in \mathbb{R}^{|\mathrm{G}|\times|\mathrm{G}|}$ and the set of indices $\mathrm{G} \in \{1,\ldots,p\}$. The bias of the plug-in estimator $\widehat{\gamma}_{\mathrm{G}}^{\intercal} A \widehat{\gamma}_{\mathrm{G}}$ is:

$$\widehat{\gamma}_{\mathrm{G}}^{\intercal} A \widehat{\gamma}_{\mathrm{G}} - \gamma_{\mathrm{G}}^{\intercal} A \gamma_{\mathrm{G}} = 2\, \widehat{\gamma}_{\mathrm{G}}^{\intercal} A \left(\widehat{\beta}_{\mathrm{G}}^{(2)} - \beta_{\mathrm{G}}^{(2)}\right) - 2\, \widehat{\gamma}_{\mathrm{G}}^{\intercal} A \left(\widehat{\beta}_{\mathrm{G}}^{(1)} - \beta_{\mathrm{G}}^{(1)}\right) - \left(\widehat{\gamma}_{\mathrm{G}} - \gamma_{\mathrm{G}}\right)^{\intercal} A \left(\widehat{\gamma}_{\mathrm{G}} - \gamma_{\mathrm{G}}\right).$$

The key step is to estimate the error components $\widehat{\gamma}_{\mathrm{G}}^{\intercal} A \left(\widehat{\beta}_{\mathrm{G}}^{(1)} - \beta_{\mathrm{G}}^{(1)}\right)$ and $\widehat{\gamma}_{\mathrm{G}}^{\intercal} A \left(\widehat{\beta}_{\mathrm{G}}^{(2)} - \beta_{\mathrm{G}}^{(2)}\right)$ in the above decomposition. We apply linear functional techniques twice here, and propose the bias-corrected estimator:

$$\begin{aligned}\widehat{\gamma_{\mathrm{G}}^{\intercal} A \gamma_{\mathrm{G}}} = \widehat{\gamma}_{\mathrm{G}}^{\intercal} A \widehat{\gamma}_{\mathrm{G}} &- 2\, \widehat{u}_1^{\intercal} \frac{1}{n_1} \sum_{i=1}^{n_1} \omega(X_{i\cdot}^{(1)\intercal} \widehat{\beta}^{(1)}) \left(y_i^{(1)} - f(X_{i\cdot}^{(1)\intercal} \widehat{\beta}^{(1)})\right) X_{i\cdot}^{(1)} \\ &+ 2\, \widehat{u}_2^{\intercal} \frac{1}{n_2} \sum_{i=1}^{n_2} \omega(X_{i\cdot}^{(2)\intercal} \widehat{\beta}^{(2)}) \left(y_i^{(2)} - f(X_{i\cdot}^{(2)\intercal} \widehat{\beta}^{(2)})\right) X_{i\cdot}^{(2)},\end{aligned} \tag{18}$$

where $\widehat{u}_1$ and $\widehat{u}_2$ are the projection directions defined in (10) with $x_{\text{new}} = \left(\widehat{\gamma}_{\mathrm{G}}^{\intercal} A, \mathbf{0}\right)^{\intercal}$ but on two different sample data respectively. The second term on right-hand-side of (18) is to estimate $-2\, x_{\text{new}}^{\intercal}(\widehat{\beta}_{\mathrm{G}}^{(1)} - \beta_{\mathrm{G}}^{(1)})$ and the third term on right-hand-side of (18) is to estimate $-2\, x_{\text{new}}^{\intercal}(\widehat{\beta}_{\mathrm{G}}^{(2)} - \beta_{\mathrm{G}}^{(2)})$.

To maintain non-negativity of distance, we define $\widehat{\gamma_{\mathrm{G}}^{\intercal} A \gamma_{\mathrm{G}}} = \max\left\{\widehat{\gamma_{\mathrm{G}}^{\intercal} A \gamma_{\mathrm{G}}},\ 0\right\}$ and estimate its corresponding asymptotic variance as

$$\widehat{\mathrm{V}}_A(\tau) = 4\, \widehat{\mathrm{V}}^{(1)} + 4\, \widehat{\mathrm{V}}^{(2)} + \frac{\tau}{\min(n_1, n_2)},$$

where $\widehat{\mathrm{V}}^{(k)}$ is computed as in (11) for the k-th regression model ($k = 1, 2$) and the term $\tau / \min(n_1, n_2)$ with $\tau > 0$ is introduced as an upper bound for the term $(\widehat{\gamma}_{\mathrm{G}} - \gamma_{\mathrm{G}})^{\intercal} A (\widehat{\gamma}_{\mathrm{G}} - \gamma_{\mathrm{G}})$. With asymptotic normality, we construct the CI for $\gamma_{\mathrm{G}}^{\intercal} A \gamma_{\mathrm{G}}$ as

$$\mathrm{CI}(\tau) = \left(\max\left(\widehat{\gamma_{\mathrm{G}}^{\intercal} A \gamma_{\mathrm{G}}} - z_{\alpha/2} \sqrt{\widehat{\mathrm{V}}_A(\tau)},\ 0\right),\ \widehat{\gamma_{\mathrm{G}}^{\intercal} A \gamma_{\mathrm{G}}} + z_{\alpha/2} \sqrt{\widehat{\mathrm{V}}_A(\tau)}\right).$$

We further consider the unknown matrix $A = \Sigma_{\mathrm{G,G}}$ and construct the point estimator $\widehat{\gamma_{\mathrm{G}}^{\intercal} \Sigma_{\mathrm{G,G}} \gamma_{\mathrm{G}}}$ in a similar way as outlined in (18). In this case, the submatrix A is substituted with $\widehat{\Sigma}_{\mathrm{G,G}}$, where $\widehat{\Sigma}_{\mathrm{G,G}} = \frac{1}{n_1+n_2} \sum_{i=1}^{n_1+n_2} X_{i,\mathrm{G}} X_{i,\mathrm{G}}^{\intercal}$ with X as the row-combined matrix of $X^{(1)}$ and $X^{(2)}$. Its corresponding asymptotic variance is

$$\widehat{\mathrm{V}}_{\Sigma}(\tau) = 4\, \widehat{\mathrm{V}}^{(1)} + 4\, \widehat{\mathrm{V}}^{(2)} + \frac{1}{(n_1+n_2)^2} \sum_{i=1}^{n_1+n_2} \left(\widehat{\gamma}_{\mathrm{G}}^{\intercal} X_{i,\mathrm{G}} X_{i,\mathrm{G}}^{\intercal} \widehat{\gamma}_{\mathrm{G}} - \widehat{\gamma}_{\mathrm{G}}^{\intercal} \widehat{\Sigma}_{\mathrm{G,G}} \widehat{\gamma}_{\mathrm{G}}\right)^2 + \frac{\tau}{\min(n_1, n_2)}.$$

Next we present the CI for $\gamma_{\mathrm{G}}^{\intercal} \Sigma_{\mathrm{G,G}} \gamma_{\mathrm{G}}$.

$$\mathrm{CI}(\tau) = \left(\max\left(\widehat{\gamma_{\mathrm{G}}^{\intercal} \Sigma_{\mathrm{G,G}} \gamma_{\mathrm{G}}} - z_{\alpha/2} \sqrt{\widehat{\mathrm{V}}_{\Sigma}(\tau)},\ 0\right),\ \widehat{\gamma_{\mathrm{G}}^{\intercal} \Sigma_{\mathrm{G,G}} \gamma_{\mathrm{G}}} + z_{\alpha/2} \sqrt{\widehat{\mathrm{V}}_{\Sigma}(\tau)}\right).$$

3 Usage of the package

The **SIHR** package contains a set of functions for conducting inference for various transformations of high-dimensional regression vectors, such as linear and quadratic functions. We summarize the functions and their corresponding objectives in the following Table 2.

3.1 Linear functional

The function `LF()`, shorthanded for Linear Functional, performs inference for $x_{\text{new}}^{\intercal} \beta$, under the high-dimensional model (1), where x_{new} is a given vector. A typical `LF()` code snippet looks like:

Function	Inference Objective	Description
LF()	$x_{\text{new}}^{\intercal}\beta$	Generates a LF object which includes the bias-corrected estimator of $x_{\text{new}}^{\intercal}\beta$ in high-dimensional GLM and the corresponding standard error, which are further used to construct CI and conduct hypothesis testing related to $x_{\text{new}}^{\intercal}\beta$.
QF()	$\beta_{\text{G}}^{\intercal}A\beta_{\text{G}}$	Generates a QF object which includes the bias-corrected estimator of $\beta_{\text{G}}^{\intercal}A\beta_{\text{G}}$ in high-dimensional GLM, for $A \in \mathbb{R}^{\lvert\text{G}\rvert\times\lvert\text{G}\rvert}$ and index set $\text{G} \in \{1,\ldots,p\}$, and computes the corresponding standard error, which are further used to construct CI and conduct hypothesis testing related to $\beta_{\text{G}}^{\intercal}A\beta_{\text{G}}$.
CATE()	$f(x_{\text{new}}^{\intercal}\beta^{(2)}) - f(x_{\text{new}}^{\intercal}\beta^{(1)})$	Generates a CATE object which includes the bias-corrected estimator of $f(x_{\text{new}}^{\intercal}\beta^{(2)}) - f(x_{\text{new}}^{\intercal}\beta^{(1)})$ in high-dimensional GLMs and the corresponding standard error, which are further used to construct CI and conduct hypothesis testing related to $f(x_{\text{new}}^{\intercal}\beta^{(2)}) - f(x_{\text{new}}^{\intercal}\beta^{(1)})$.
InnProd()	$\beta_{\text{G}}^{(1)\intercal}A\beta_{\text{G}}^{(2)}$	Generates an InnProd object which includes the bias-corrected estimator of $\beta_{\text{G}}^{(1)\intercal}A\beta_{\text{G}}^{(2)}$ in high-dimensional GLMs, for $A \in \mathbb{R}^{\lvert\text{G}\rvert\times\lvert\text{G}\rvert}$ and index set $\text{G} \in \{1,\ldots,p\}$, and computes the corresponding standard error, which are further used to construct CI and conduct hypothesis testing related to $\beta_{\text{G}}^{(1)\intercal}A\beta_{\text{G}}^{(2)}$.
Dist()	$\gamma_{\text{G}}^{\intercal}A\gamma_{\text{G}}$ with $\gamma = \beta^{(2)} - \beta^{(1)}$	Generates a Dist object which includes the bias-corrected estimator of $\gamma_{\text{G}}^{\intercal}A\gamma_{\text{G}}$ in high-dimensional GLMs, for $A \in \mathbb{R}^{\lvert\text{G}\rvert\times\lvert\text{G}\rvert}$ and index set $\text{G} \in \{1,\ldots,p\}$, and the corresponding standard error, which are further used to construct CI and conduct hypothesis testing related to $\gamma_{\text{G}}^{\intercal}A\gamma_{\text{G}}$.
ci()	—	Input object (`LF/ QF/ CATE/ InnProd/ Dist`), returns CI.
summary()	—	Input object (`LF/ QF/ CATE/ InnProd/ Dist`), computes and returns a list of summary statistics, including plug-in estimator, bias-corrected estimator together with associated standard error and p-value.

Table 2: Functions of SIHR, which perform statistical inference for low-dimensional objectives in high-dimensional GLM for continuous and binary outcomes.

```
LF(X, y, loading.mat, model = c("linear", "logistic", "logistic_alter"),
  intercept = TRUE, intercept.loading = FALSE, beta.init = NULL, lambda = NULL,
  mu = NULL, prob.filter = 0.05, rescale = 1.1, verbose = FALSE)
```

In the following we provide descriptions of the various arguments of the `LF` function :

- `X` is the design matrix of dimension $n \times p$ and `y` is the response vector of length n.
- `loading.mat` is the matrix of loading vectors where each column corresponds to a new future observation x_{new}. It is designed to allow for taking multiple x_{new}'s as input, thereby saving the computational time of constructing the initial estimator multiple times.
- `model` (default = `"linear"`) specifies which high-dimensional regression model to be fitted, the choices being `c("linear", "logistic", "logistic_alter")`, where "linear" corresponds to the linear model and "logistic" and "logistic_alter" correspond to the logistic regression; see Table 1.
- `intercept` (default = `TRUE`) is a logical argument that specifies whether an intercept term should be fitted while computing the initial estimator in (3).
- `intercept.loading` (default = `FALSE`) is a logical argument that specifies whether the intercept term should be included for defining the objective $x_{\text{new}}^{\intercal}\beta$. Specifically, setting `intercept.loading = TRUE` prepend a column of 1's to the matrix `loading.mat`.
- `beta.init` (default = `NULL`) allows the user to supply the initial estimator $\widehat{\beta}$ of the regression vector. If `beta.init` is left as `NULL`, the initial estimator $\widehat{\beta}$ in (3) is computed using function `cv.glmnet` in glmnet.
- `lambda` (default = `NULL`) denotes the scaled tuning parameter λ used for computing the initial estimator $\widehat{\beta}$ in (3) which can either be pre-specified or can be set to `NULL` whence `LF` uses the function `cv.glmnet` in glmnet to compute the tuning parameter.

- `mu` (default = `NULL`) denotes the tuning parameter μ_0 in (10). When `mu` is set as `NULL`, it is computed as the smallest μ_0 such that (10) has a finite solution.
- `prob.filter` (default = `0.05`) is specific to `model = "logistic"`. From Table 1, observe that `model = "logistic"` sets the weight for i-th individual as $\frac{1}{\mathbb{P}(y_i=1|X_{i\cdot})\cdot(1-\mathbb{P}(y_i=1|X_{i\cdot}))}$ which can blow up if the estimated probabilities $\mathbb{P}(y_i \mid X_{i\cdot})$ are very close to 0 or 1. We discard those samples for which the estimated probability lies outside [`prob.filter`, $1-$ `prob.filter`] before proceeding with the algorithm.
- `rescale` (default = `1.1`) denotes the factor used to enlarge the standard error to account for the finite sample bias, as pointed out in Remark 1.
- `verbose` (default = `FALSE`) is a logical argument that specifies whether intermediate message(s) should be printed, the projection direction be returned.

Remark 2 *The structure of the* `loading.mat` *is designed so that each column corresponds to a future observation x_{new}. This matrix structure optimizes computational efficiency by allowing the debiasing algorithm to process multiple linear functionals simultaneously; that is, when* `loading.mat` *contains multiple columns, the* `LF()` *function only requires computing the initial estimator $\widehat{\beta}$ in* (3) *once. Specifically, when* `loading.mat` *is set as the identity matrix of dimension p, where p represents the number of covariates, the* `LF()` *function conducts inference for all p individual regression coefficients concurrently.*

Next, we provide an example to illustrate the usage of `LF()` in the linear regression model.

Example 1. For $1 \le i \le n$ with $n = 100$, the covariates $X_{i\cdot}$ are independently generated from the multivariate normal distribution with mean $\mu = 0_p$ and covariance $\Sigma = \mathbf{I}_p$ with $p = 120$, where $\mathbf{I}_p$ is an identity matrix of dimension p. The regression vector $\beta \in \mathbb{R}^p$ is generated as $\beta_1 = 0.5, \beta_2 = 1$ and $\beta_j = 0$ if $3 \le j \le p$. The outcome is generated as $y_i = X_{i\cdot}^{\intercal}\beta + \epsilon_i$ with independently generated standard normal ϵ_i.

```
n <- 100; p <- 120
mu <- rep(0,p); Cov <- diag(p)
beta <- rep(0,p); beta[c(1,2)] <- c(0.5, 1)
X <- MASS::mvrnorm(n, mu, Cov)
y <- X %*% beta + rnorm(n)
```

We now generate two observations $x_{\text{new}}^{(1)}, x_{\text{new}}^{(2)}$ and apply the `LF()` function to construct the point estimators of $x_{\text{new}}^{(1)\intercal}\beta$ and $x_{\text{new}}^{(2)\intercal}\beta$, together with their standard error estimates.

```
loading1 <- c(1, 1, rep(0, p-2))
loading2 <- c(-0.5, -1, rep(0, p-2))
loading.mat <- cbind(loading1, loading2)
Est <- LF(X, y, loading.mat, model = 'linear')
```

Having fitted the model, we have two following functions `ci()` and `summary()`. We first report the 95% CIs for $x_{\text{new}}^{(1)\intercal}\beta$ and $x_{\text{new}}^{(2)\intercal}\beta$, where the true values $x_{\text{new}}^{(1)\intercal}\beta = 1.5$ and $x_{\text{new}}^{(2)\intercal}\beta = -1.25$ are contained in the corresponding CIs.

```
ci(Est)
#>  loading     lower      upper
#>1       1  1.167873  1.8753934
#>2       2 -1.544138 -0.7995375
```

Then, we apply the `summary()` function to return a list of the summary statistics, including the plugin estimator, bias-corrected estimator, the standard error for the bias-corrected estimator and the p-value corresponding to the hypothesis testing $H_0 : x_{\text{new}}^{\intercal}\beta = 0$ vs $H_1 : x_{\text{new}}^{\intercal}\beta \neq 0$. It is observed that the bias-corrected estimators are closer to the true values compared to the plug-in estimators.

```
summary(Est)
#>Call:
#>Inference for Linear Functional
#>
#>Estimators:
#> loading est.plugin est.debias Std. Error z value  Pr(>|z|)
#>       1      1.268      1.522     0.1805   8.430 0.000e+00 ***
#>       2     -1.033     -1.172     0.1900  -6.169 6.868e-10 ***
```

3.2 Quadratic functional

For a given index set $\mathrm{G} \subset \{1, \ldots, p\}$, the function `QF()`, abbreviated for Quadratic Functional, conducts inference for $\beta_{\mathrm{G}}^{\intercal} A \beta_{\mathrm{G}}$ if $A \in \mathbb{R}^{|\mathrm{G}| \times |\mathrm{G}|}$ is the submatrix pre-specified or $\beta_{\mathrm{G}}^{\intercal} \Sigma_{\mathrm{G},\mathrm{G}} \beta_{\mathrm{G}}$ under the high-dimensional regression model (1). The function `QF()` can be called with the following arguments.

```
QF(X, y, G, A = NULL, model = c("linear", "logistic", "logistic_alter"),
   intercept = TRUE, beta.init = NULL, split = TRUE, lambda = NULL, mu = NULL,
   prob.filter = 0.05, rescale = 1.1, tau = c(0.25, 0.5, 1), verbose = FALSE)
```

In the function `QF()`, the parameters `X`, `y`, `model`, `intercept`, `beta.init`, `lambda`, `mu`, `prob.filter`, `rescale` maintain the same definitions as in the `LF()` function. In the following, we primarily focus on elaborating the additional arguments introduced for the `QF()` function.

- $\mathrm{G} \subset \{1, \ldots, p\}$ is the set of indices of interest.
- `A` is the matrix in the quadratic form, of dimension $|\mathrm{G}| \times |\mathrm{G}|$. If `A` is specified, it will conduct inference for $\beta_{\mathrm{G}}^{\intercal} A \beta_{\mathrm{G}}$; otherwise, if left `NULL`, it will turn to $\beta_{\mathrm{G}}^{\intercal} \Sigma_{\mathrm{G},\mathrm{G}} \beta_{\mathrm{G}}$.
- `split` (default = `TRUE`) indicates whether we conduct sample splitting. When `split=FALSE`, the initial estimator of regression coefficients in (3) is computed using one half of the sample while the remaining half is used for bias correction in (13). When `split=TRUE`, the full data is used for computing both the initial estimator and conducting the bias correction.
- `tau.vec` (default = `c(0.25,0.5,1)`) allows the user to supply a vector of possible values for τ used in (14) and (15).

In the following, we illustrate the usage of `QF()` in the linear regression model.

Example 2. For $1 \leq i \leq n$, with $n = 200$, the covariates $X_{i\cdot}$ are generated from multivariate normal distribution with mean $\mu = 0_p$ and covariance $\Sigma \in \mathbb{R}^{p \times p}$, with p = 150, where $\Sigma_{j,k} = 0.5^{|j-k|}$ for $1 \leq j,k \leq p$. The regression coefficients β is constructed as $\beta_j = 0.2$ for $25 \leq j \leq 50$ and $\beta_j = 0$ otherwise. We generate the outcome following the model $y_i = X_{i\cdot}^{\intercal} \beta + \epsilon_i$ with ϵ_i generated as the standard normal.

```
n <- 200; p <- 150
mu <- rep(0,p)
Cov <- matrix(0, p, p)
for(j in 1:p) for(k in 1:p) Cov[j,k] <- 0.5^(abs(j-k))
beta <- rep(0, p); beta[25:50] <- 0.2
X <- MASS::mvrnorm(n, mu, Cov)
y <- X%*%beta + rnorm(n)
```

We apply the `QF()` function to obtain the point estimator of $\beta_{\mathrm{G}}^{\intercal} \Sigma_{\mathrm{G},\mathrm{G}} \beta_{\mathrm{G}}$ with $G = \{40, \ldots, 60\}$ along with the standard error estimator.

```
test.set <- c(40:60)
Est <- QF(X, y, G = test.set, A = NULL, model = "linear", split = FALSE)
```

We run the function `ci()` that outputs the CIs for Q_{Σ} corresponding to different values of τ. With the default $\tau = c(0.25, 0.5, 1)$, we obtain three different CIs for $\beta_{\mathrm{G}}^{\intercal} \Sigma_{\mathrm{G},\mathrm{G}} \beta_{\mathrm{G}}$; see (16). Note that the true value $\beta_{\mathrm{G}}^{\intercal} \Sigma_{\mathrm{G},\mathrm{G}} \beta_{\mathrm{G}} = 1.16$ belongs to all of these constructed CIs.

```
ci(Est)
#>  tau     lower    upper
#>1 0.25 0.8118792 1.466422
#>2 0.50 0.8046235 1.473677
#>3 1.00 0.7905648 1.487736
```

Subsequently, we employ the `summary()` function to yield the bias-corrected and plug-in estimators, alongside the standard errors for the debiased estimator across different values of τ. Additionally, it provides the p-values for the hypothesis testing $H_0 : \mathrm{Q}_{\Sigma} = 0$ versus $H_1 : \mathrm{Q}_{\Sigma} > 0$.

```
summary(Est)
#> Call:
#> Inference for Quadratic Functional
#>
#>  tau est.plugin est.debias Std. Error z value  Pr(>|z|)
#> 0.25      0.904      1.139     0.1670   6.822 8.969e-12 ***
#> 0.50      0.904      1.139     0.1707   6.674 2.486e-11 ***
#> 1.00      0.904      1.139     0.1779   6.405 1.504e-10 ***
```

Similarly to the `LF()` case, our proposed bias-corrected estimator effectively corrects the plugin estimator's bias, where the true value is 1.16.

3.3 Conditional average treatment effect

The function `CATE()`, shorthanded for Conditional Average Treatment Effect, conducts inference for $\Delta(x_{\text{new}}) = f(x_{\text{new}}^{\intercal}\beta^{(2)}) - f(x_{\text{new}}^{\intercal}\beta^{(1)})$ under the high-dimensional regression model (2). This function can be implemented as follows:

```
CATE(X1, y1, X2, y2, loading.mat, model = c("linear", "logistic", "logistic_alter"),
   intercept = TRUE, intercept.loading = FALSE, beta.init1 = NULL, beta.init2 = NULL,
   lambda = NULL, mu = NULL, prob.filter = 0.05, rescale = 1.1, verbose = FALSE)
```

The majority of the arguments remain consistent with those of the `LF()` function. We will highlight the new parameters specific to the `CATE()` function.

- `X1` and `y1` respectively denote the design matrix and the response vector for the first sample of data, while `X2` and `y2` denote those for the second sample of data.
- `beta.init1` (default = `NULL`) is the initial estimator in (3) for the first sample, while `beta.init2` (default = `NULL`) is for the second sample. If left as `NULL`, they are computed using `cv.glmnet` in **glmnet**.
- `lambda` (default = `NULL`) represents the common tuning parameter λ for computing the initial estimators `beta.init1` and `beta.init2`. If left as `NULL`, `cv.glmnet` in **glmnet** is employed for its computation, done separately for each sample.
- `mu` (default = `NULL`) represents the common tuning parameter μ_0 in (10) for computing the projection directions for the two samples. When unspecified and left as `NULL`, it is computed as the smallest μ_0 such that (10) has a finite solution, done separately for each sample.

We consider the logistic regression case to illustrate `CATE()` with the argument `model = "logistic_alter"`.

Example 3. In the first group of data, the covariates $X_{i\cdot}^{(1)}$, for $1 \leq i \leq n_1$ with $n_1 = 100$, follow multivariate normal distribution with $\mu = 0_p$ and covariance $\Sigma^{(2)} = \mathbf{I}_p$; in the second group of data, the covariates $X_{i\cdot}^{(2)}$, for $1 \leq i \leq n_2$ with $n_2 = 180$, follow multivariate normal distribution with $\mu = 0_p$ and covariance $\Sigma^{(2)} \in \mathbb{R}^{p\times p}$ with $p = 120$ and $\Sigma_{j,k}^{(2)} = 0.5^{|j-k|}$ for $1 \leq j,k \leq p$. We generate the binary outcomes following the model $y_i^{(k)} \sim \text{Bernoulli}(f(X_{i\cdot}^{(k)\intercal}\beta^{(k)}))$ with $f(z) = \exp(z)/[1+\exp(z)]$ for $k = 1,2$. See the following code for details of $\beta^{(1)}$ and $\beta^{(2)}$.

```
n1 <- 100; n2 <- 180; p <- 120
mu1 <- mu2 <- rep(0,p)
Cov1 <- diag(p)
Cov2 <- matrix(0, p, p)
for(j in 1:p) for(k in 1:p) Cov2[j,k] <- 0.5^(abs(j-k))
beta1 <- rep(0, p); beta1[c(1,2)] <- c(0.5, 0.5)
beta2 <- rep(0, p); beta2[c(1,2)] <- c(1.8, 1.8)
X1 <- MASS::mvrnorm(n1, mu1, Cov1); val1 <- X1%*%beta1
X2 <- MASS::mvrnorm(n2, mu2, Cov2); val2 <- X2%*%beta2
y1 <- rbinom(n1, 1, exp(val1)/(1+exp(val1)))
y2 <- rbinom(n2, 1, exp(val2)/(1+exp(val2)))
```

We then employ the function `CATE()` to obtain point estimator of $\Delta(x_{\text{new}})$ and the associated standard error estimator. By setting `model = "logistic_alter"`, we set the weight $w(.) = 1$ in (9). See Table 1.

```
loading.mat <- c(1, 1, rep(0, p-2))
Est <- CATE(X1, y1, X2, y2,loading.mat, model = "logistic_alter")
```

Having fitted the model, it allows for method `ci()` and `summary()` as `LF()` does. We mainly demonstrate the `ci()` function and first construct confidence interval for $x_{\text{new}}^{\intercal}(\beta^{(2)} - \beta^{(1)})$ and observe that 95% CI covers the true value 2.6.

```
ci(Est)
#>  loading    lower    upper
#>1       1 1.614269 4.514703
```

If we further specify the argument `probability` as `TRUE` for the logistic regression, `ci()` yields the CI for $f(x_{\text{new}}^{\intercal}\beta^{(2)}) - f(x_{\text{new}}^{\intercal}\beta^{(1)})$ whose true value is 0.2423.

```
ci(Est, probability = TRUE)
#>  loading     lower     upper
#>1       1 0.1531872 0.5086421
```

3.4 Inner product

The function `InnProd()`, shorthanded for Inner Product, conducts inference for $\beta_{\mathrm{G}}^{(1)\intercal} A \beta_{\mathrm{G}}^{(2)}$ with $A \in \mathbb{R}^{|\mathrm{G}| \times |\mathrm{G}|}$ where G denotes the prespecified index set. When the matrix A is not specified, the default inference target becomes $\beta_{\mathrm{G}}^{(1)\intercal} \Sigma_{\mathrm{G,G}} \beta_{\mathrm{G}}^{(2)}$.

```
InnProd(X1, y1, X2, y2, G, A = NULL, model = c("linear", "logistic", "logistic_alter"),
   intercept = TRUE, beta.init1 = NULL, beta.init2 = NULL, split = TRUE, lambda = NULL,
   mu = NULL, prob.filter = 0.05, rescale = 1.1, tau = c(0.25,0.5,1), verbose = FALSE)
```

The arguments of `InnProd()` are similarly defined as for the function `CATE()`, and we mainly highlight the new arguments in the following.

- `G` is the pre-specified index set, a subset of $\{1, \cdots, p\}$.
- `A` is the matrix in the inner product form. If the matrix `A` is specified, it will conduct inference for $\beta_{\mathrm{G}}^{(1)\intercal} A \beta_{\mathrm{G}}^{(2)}$; otherwise, it will turn to $\beta_{\mathrm{G}}^{(1)\intercal} \Sigma_{\mathrm{G,G}} \beta_{\mathrm{G}}^{(2)}$ where Σ is the common design covariance matrix corresponding to the two samples.

In the following code, we demonstrate the use of `InnProd()` in the linear regression.

Example 4. In the first group of data, the covariates $X_{i\cdot}^{(1)}$, for $1 \leq i \leq n_1$ with $n_1 = 200$, follow multivariate normal distribution with $\mu = 0_p$ and covariance $\Sigma^{(1)} = \mathbf{I}_p$; in the second group of data, the covariates $X_{i\cdot}^{(2)}$, for $1 \leq i \leq n_2$ with $n_2 = 260$, follow multivariate normal distribution with $\mu = 0_p$ and covariance $\Sigma^{(2)} \in \mathbb{R}^{p \times p}$ with $p = 120$ and $\Sigma_{j,k}^{(2)} = 0.5^{|j-k|}$ for $1 \leq j,k \leq p$. We generate following the model $y_i^{(k)} = X_{i\cdot}^{(k)\intercal} \beta^{(k)} + \epsilon_i^{(k)}$ with standard normal error $\epsilon_i^{(k)}$ for $k = 1, 2$.

```
n1 <- 200; n2 <- 260; p <- 120
mu1 <- mu2 <- rep(0,p)
Cov1 <- diag(p)
Cov2 <- matrix(0, p, p)
for(j in 1:p) for(k in 1:p) Cov2[j,k] <- 0.5^(abs(j-k))
beta1 <- rep(0, p); beta1[1:10] <- 0.5
beta2 <- rep(0, p); beta2[3:12] <- 0.4
X1 <- MASS::mvrnorm(n1, mu1, Cov1)
X2 <- MASS::mvrnorm(n2, mu2, Cov2)
y1 <- X1%*%beta1 + rnorm(n1)
y2 <- X2%*%beta2 + rnorm(n2)
```

After preparing the data, we utilize the `InnProd` function to build a debiased estimator and its associated standard error for $\beta_{\mathrm{G}}^{(1)\intercal} \beta_{\mathrm{G}}^{(2)}$ with $A = \mathbf{I}_{|\mathrm{G}|}$ and $G = \{1, 2, \ldots, 20\}$.

```
test.set <- c(1:20)
A <- diag(length(test.set))
Est <- InnProd(X1, y1, X2, y2, G = test.set, A, model = "linear")
```

Having fitted the model, it allows for method `ci()` and `summary()` as `QF()` does. Note that the true value $\beta_{\mathrm{G}}^{(1)\intercal} \beta_{\mathrm{G}}^{(2)} = 1.6$ is included in the following CIs with default values $\tau \in \{0.25, 0.5, 1\}$.

```
ci(Est)
#>    tau     lower    upper
#> 1 0.25 0.7432061 2.490451
#> 2 0.50 0.7128181 2.520839
#> 3 1.00 0.6520422 2.581615
```

3.5 Distance

The function `Dist()`, shorthanded for Distance, is designed to perform inference for the quadratic form $\gamma_{\mathrm{G}}^{\intercal} A \gamma_{\mathrm{G}}$, where $\gamma = \beta^{(2)} - \beta^{(1)}$ and the index set $\mathrm{G} \subset \{1, \ldots, p\}$. The matrix A can either be a pre-specified submatrix in $\mathbb{R}^{|\mathrm{G}| \times |\mathrm{G}|}$ or the covariance matrix $\Sigma_{\mathrm{G,G}}$ within the context of high-dimensional regression models.

```
Dist(X1, y1, X2, y2, G, A = NULL, model = c("linear", "logistic", "logistic_alter"),
   intercept = TRUE, beta.init1 = NULL, beta.init2 = NULL, split = TRUE, lambda = NULL,
   mu = NULL, prob.filter = 0.05, rescale = 1.1, tau = c(0.25,0.50,1), verbose = FALSE)
```

The arguments of `Dist()` are similarly defined as for the function `CATE()`, and we mainly highlight the new arguments in the following.

- `G` is the pre-specified index set, a subset of $\{1, \cdots, p\}$.
- `A` is the matrix in the inner product form. If the matrix `A` is specified, it will conduct inference for $\gamma_G^\intercal A \gamma_G$; otherwise, it will turn to $\gamma_G^\intercal \Sigma_{G,G} \gamma_G$ where Σ is the common design covariance matrix corresponding to the two samples.

In the following example, we demonstrate the application of the `Dist()` function within a linear regression context.

Example 5. For the first group of data, the covariates $X_{i\cdot}^{(1)}$, for each $1 \leq i \leq n_1$ where $n_1 = 220$, are drawn from a multivariate normal distribution with mean $\mu = 0_p$ and covariance $\Sigma^{(1)} = \mathbf{I}_p$. In the second group of data, the covariates $X_{i\cdot}^{(2)}$, for each $1 \leq i \leq n_2$ with $n_2 = 180$, also follow multivariate normal distribution with mean $\mu = 0_p$ and covariance $\Sigma^{(2)} \in \mathbb{R}^{p \times p}$ with $p = 100$ and $\Sigma_{j,k}^{(2)} = 0.5^{|j-k|}$ for $1 \leq j, k \leq p$. The regression coefficients $\beta^{(1)}$ and $\beta^{(2)}$ are generated in the following code. Outcomes for both groups are then generated according to the model $y_i^{(k)} = X_{i\cdot}^{(k)\intercal} \beta^{(k)} + \epsilon_i^{(k)}$ where $\epsilon_i^{(k)}$ is a standard normal error for $k = 1, 2$.

```
n1 <- 220; n2 <- 180; p <- 100
mu <- rep(0,p); Cov <- diag(p)
beta1 <- rep(0, p); beta1[1:2] <- c(0.5, 1)
beta2 <- rep(0, p); beta2[1:10] <- c(0.3, 1.5, rep(0.08, 8))
X1 <- MASS::mvrnorm(n1, mu, Cov)
X2 <- MASS::mvrnorm(n2, mu, Cov)
y1 <- X1%*%beta1 + rnorm(n1)
y2 <- X2%*%beta2 + rnorm(n2)
```

Next we employ the `Dist()` function to construct a debiased estimator for $\gamma_G^\intercal \Sigma_{G,G} \gamma_G$ with $G = \{1, \ldots, 10\}$, alongside the corresponding estimated standard error.

```
test.set <- c(1:10)
Est <- Dist(X1, y1, X2, y2, G = test.set, A = NULL, model = "linear", split = FALSE)
```

Having fitted the model, it allows for methods `ci()` and `summary()` as `QF()` does. Here, the true value is $\gamma_G^\intercal \Sigma_{G,G} \gamma_G = 0.3412$. Similar to the previous instances, we note that the bias-corrected estimator effectively corrects the bias of the plugin estimator. Depending on the τ values, we obtain various CIs, all of which encompass the true value. It is important to mention that in case of negative lower boundaries, they will be truncated at 0 for $\tau = 0.5$ and $\tau = 1$.

```
ci(Est)
#>   tau    lower     upper
#>1 0.25 0.028202 0.6831165
#>2 0.50 0.000000 0.7196383
#>3 1.00 0.000000 0.7926819

summary(Est)
#> Call:
#> Inference for Distance
#>
#>  tau est.plugin est.debias Std. Error z value Pr(>|z|)
#> 0.25     0.4265     0.3557     0.1671   2.129  0.03327 *
#> 0.50     0.4265     0.3557     0.1857   1.915  0.05547 .
#> 1.00     0.4265     0.3557     0.2230   1.595  0.11070
```

4 Comparative analysis

In this section, we perform a comparative analysis of SIHR compared to existing methods in numerical simulations. Initially, we compare our approach to traditional plug-in Lasso estimators, implemented by **glmnet**, to demonstrate the effectiveness of bias correction. Subsequently, we will compare our method against other inference techniques implemented by **hdi** and **SSLasso** (available at `http://web.stanford.edu/~montanar/sslasso/code.html`). We aim to demonstrate that our proposed method ensures a unified guarantee of coverage across a wider range of settings while also significantly enhancing computational efficiency.

Throughout this section, we evaluate the performance of our `LF()` estimator using synthetic data, generated as follows. For $1 \le i \le n$,

$$X_{i\cdot} \overset{iid}{\sim} \mathcal{N}(0_p, \mathbf{I}_p), \;\; y_i \sim \begin{cases} \mathcal{N}(X_{i\cdot}^{\intercal}\beta, 1), & \text{for linear model;} \\ \text{Bernoulli}(X_{i\cdot}^{\intercal}\beta), & \text{for logistic model} \end{cases}, \;\; \text{where } \beta = (0.5, 0.75, 0.25, 0_{p-3}).$$

The sample size n varies across $\{200, 400\}$ with the number of covariates fixed as $p = 300$. The loading is set as $x_{\text{new}} = (1, 0.75, 0.5, 0_{p-3})^{\intercal}$, making our inference target $x_{\text{new}}^{\intercal}\beta = 1.1875$. All results summarized here are based on 500 simulation rounds.

4.1 Effectiveness of bias correction

We compare the bias-corrected estimator $\widehat{x_{\text{new}}^{\intercal}\beta}$, as defined in (9), against the plug-in Lasso estimator $x_{\text{new}}^{\intercal}\widehat{\beta}$, where $\widehat{\beta}$ represents the Lasso estimator in (3) implemented by **glmnet** with a tuning parameter selected through the package's built-in cross-validation.

Figure 1 reports the performance for logistic regression with $n = 400$ and displays histograms of 500 point estimates for the plug-in Lasso and the debiased estimates outputted by **SIHR**. The target $x_{\text{new}}^{\intercal}\beta = 1.1875$ is highlighted with red vertical lines. It is evident that the plug-in Lasso estimators exhibit significant bias and are not suitable for inference, whereas our bias-corrected estimators effectively correct this bias.

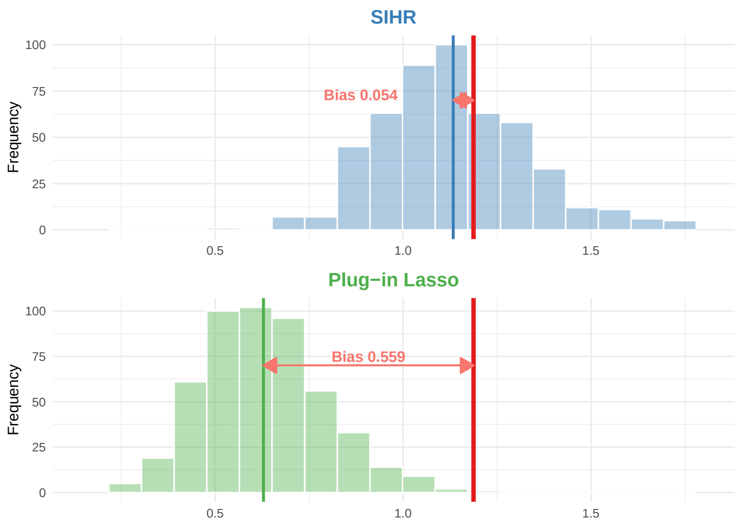

Figure 1: Comparison of the debiased estimates output by **SIHR** and plug-in Lasso estimates for $x_{\text{new}}^{\intercal}\beta$ in the logistic model with $n = 400$. The upper panel shows the bias-corrected point estimates derived using our package **SIHR**, while the lower panel features the plug-in point estimates from the **glmnet** package. Red vertical lines indicate the target value $x_{\text{new}}^{\intercal}\beta = 1.1875$. Biases between this target and the empirical means of the estimates are highlighted for each method.

4.2 Comparison with other inference methods

We compare the performance of the **SIHR** package with existing softwares, including R packages **hdi** and **SSLasso**. The performance metrics include the empirical coverage, averaged length of confidence intervals, and averaged computational time (in seconds). All metrics are reported as the average of 500 simulation rounds. The confidence intervals based on **hdi** and **SSLasso** are defined as follows:

$$\text{CI}_{\alpha}(x_{\text{new}}) = \left(x_{\text{new}}^{\intercal}\widetilde{\beta} - z_{\alpha/2}(x_{\text{new}}^{\intercal}\widetilde{V}x_{\text{new}})^{1/2}, \quad x_{\text{new}}^{\intercal}\widetilde{\beta} + z_{\alpha/2}(x_{\text{new}}^{\intercal}\widetilde{V}x_{\text{new}})^{1/2} \right),$$

where $\widetilde{\beta}$ denotes the debiased estimator and $\widetilde{V}$ denotes the estimated covariate matrix of $\widetilde{\beta}$, outputted by **hdi** and **SSLasso**. Note that **SSLasso** only provides inference for linear regression models.

Coverage and Length As shown in Table 3, the CIs based on **SIHR** achieve desired coverage across various scenarios, and the lengths decrease with larger sample sizes. In contrast, the coverage of CIs from **hdi** and **SSLasso** may be slightly undercovered, especially when $n = 200$.

Computation Efficiency We examine the computational efficiency of these methods and report the average computation time in the "Time" column, measured in seconds. The **SIHR** package demonstrates notable computational efficiency, with an average processing time of under 10 seconds. In comparison, using the algorithm in **hdi** with parameters $n = 400$ and $p = 300$ requires approximately 8 minutes. This significant difference stems from the fact that the **hdi** algorithm is not tailored for inferring linear functionals and separately implementing p bias correction steps, one for each regression coefficient, while **SIHR** only implements a single bias correction step.

		Linear			Logistic		
n	Method	Cov	Len	Time	Cov	Len	Time
200	**SIHR**	0.94	0.47	3	0.94	0.94	3
	hdi	0.90	0.38	211	0.91	0.80	213
	SSLasso	0.83	0.34	17	-	-	-
400	**SIHR**	0.96	0.34	12	0.96	0.74	14
	hdi	0.94	0.27	475	0.89	0.56	489
	SSLasso	0.94	0.29	38	-	-	-

Table 3: Comparison of methods implemented by packages **SIHR**, **hdi**, **SSLasso** in both linear and logistic models with $n \in \{200, 400\}$. The columns labeled "Cov" and "Len" denote the empirical coverage and the length of the confidence intervals over 500 simulation runs, respectively; "Time" indicates the average computation time in seconds. The columns titled "Linear" and "Logistic" refer to the regression model applied. According to the experimental design, a valid inference method should achieve a coverage rate of approximately 0.95.

5 Real data applications

5.1 Motif regression

We showcase the application of the `LF()` function in motif regression analysis, which investigates the impact of motif matching scores on gene expression levels, as discussed in the literature (Beer and Tavazoie, 2004; Conlon et al., 2003; Das et al., 2004; Yuan et al., 2007). Motifs are specific DNA sequences bound to transcription factors, playing crucial roles in controlling transcription activities, such as gene expressions (Yuan et al., 2007). The matching score of a motif measures its prevalence, reflecting how prominently a motif appears in the upstream regions of genes. These matching scores are recognized for their effectiveness in predicting gene expression levels. Our goal is to quantitatively assess the association between these matching scores and gene expression, elucidating the underlying biological mechanisms. In this analysis, we work with a dataset that includes the expression levels of $n = 2587$ genes, where matching scores of $p = 666$ motifs are observed on each gene. The structure of the data is organized as follows: for $1 \leq i \leq 2587$,

- y_i : the expression level of gene i;
- $X_{i,j}$: the matching score of the j-th motif on gene i, for $1 \leq j \leq 666$.

Below, we display several observations of the response variable along with the first four covariates out of a total of 666.

```
colnames(X) <- paste0("X",1:ncol(X))
head(cbind(y, X[,1:4]))
#>                y        X1       X2       X3       X4
#>   YAL002W  0.51 1.1595129 1.573024 1.239862 1.144537
#>   YAL003W -3.06 1.9581497 1.928997 1.228753 1.118513
#>   YAL007C -1.86 1.3047351 1.617691 1.299527 1.126370
#>   YAL025C -1.54 0.8057353 1.487356 1.395147 1.003005
#>   YAL034C  1.00 0.8886961 1.860788 1.569881 1.316531
#>   YAL035W -2.05 1.3377646 1.152577 1.532653 1.012072
```

We seek to investigate the relationships between the matching scores of individual motifs ($X_{\cdot,j}$ for $1 \leq j \leq 666$) and gene expression levels (y). Our objective is to access the significance of these associations. For this purpose, the LF() function from the package **SIHR** is utilized to compute 95% confidence intervals for the 666 regression coefficients.

```
p <- ncol(X)
loading.mat <- diag(p)
Est <- LF(X, y, loading.mat, model='linear')
ci(Est)
```

We then summarize and visualize the resulting 666 confidence intervals in Figure 2. The results reveal that 25 of these intervals, highlighted in blue, lie entirely above zero, indicating a positive association between the matching scores of these specific motifs and gene expression levels. Conversely, 23 intervals, marked in green, fall completely below zero, suggesting a negative influence of these motifs on gene expression levels. Overall, these results demonstrate that 48 motifs out of the total have a statistically significant influence on gene expression, offering valuable insights into the regulatory mechanisms involved.

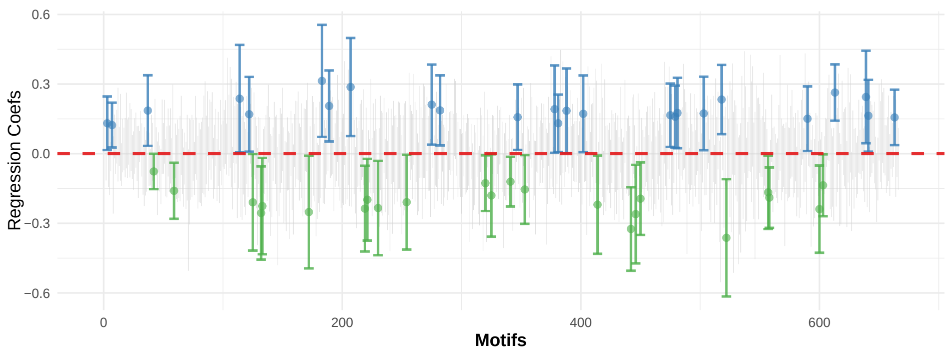

Figure 2: Motif: Constructed CIs for the 666 regression coefficients. Motifs represented by blue CIs indicate a significant positive association with gene expression levels, whereas those with green CIs demonstrate significant negative associations.

5.2 Fasting glucose level data

We have illustrated the application of the LF() function in a linear regression context. We now demonstrate its use on another real application in a logistic regression setting. In this study, we examine the impact of polymorphic genetic markers on glucose levels in a stock mice population. The data, accessible at https://wp.cs.ucl.ac.uk/outbredmice/heterogeneous-stock-mice/, uses fasting glucose levels, dichotomized at 11.1 mmol/L, as the response variable —an important indicator for type-2 diabetes. Specifically, glucose levels below 11.1 mmol/L are considered normal and labeled as $y_i = 0$, while levels above 11.1 mmol/L are classified as high, indicating pre-diabetic or diabetic conditions, and labeled as $y_i = 1$.

The dataset initially comprises $10,346$ polymorphic genetic markers for a sample size $n = 1,269$. Given the large number of markers and the significant correlation among some of them, we implement a selection criterion to ensure the maximum absolute correlation among the markers does not exceed 0.75. After filtering, we narrow down to a subset of $2,341$ genetic markers. Additionally, we include "gender" and "age" as baseline covariates. To prepare for the analysis, both the genetic markers and baseline covariates are standardized. To sum up, the data structure is organized as follows: for $i = 1, \ldots, 1269$:

- y_i : binary indicator of whether the fasting glucose level is above 11.1 mmol/L
- $X_{i,j}$: genetic marker j for mouse i with $j = 1, \ldots, 2341$
- $X_{i,2342}$: gender of mouse i
- $X_{i,2343}$: age of mouse i

Below, we display several observations of the response variable along with the first four covariates out of a total of 2343.

```
head(cbind(y, X[,1:4]))
#>              y rs3674785_G rs13475705_A rs13475706_G rs3684358_C
#>   A048005080 1    0.184158   -0.5697056    0.6063887  -0.3444252
#>   A048006555 0   -1.258413   -0.5697056   -0.9113771   1.1272098
#>   A048007096 0    0.184158    1.5137423   -0.9113771  -0.3444252
#>   A048010273 1   -1.258413   -0.5697056   -0.9113771   1.1272098
#>   A048010371 0    0.184158   -0.5697056    0.6063887  -0.3444252
#>   A048011287 0    0.184158    1.5137423   -0.9113771  -0.3444252
```

Given the real dataset, we aim to investigate the association of each polymorphic marker ($X_{\cdot,j}$ for $1 \leq j \leq 2341$) with fasting glucose levels (y) and determine the statistical significance of each association. We employ the function `LF()` configured with `model = "logistic"` to compute confidence intervals for the initial 2341 regression coefficients, which correspond to all polymorphic markers, as demonstrated in the following code:

```
p <- ncol(X)
loading.mat <- diag(p)[,-c(2342,2343)]
Est <- LF(X, y, loading.mat, model='logistic')
ci(Est)
```

We then visualize the obtained 2341 confidence intervals in Figure 3. It reveals that 13 genetic markers display CIs exclusively above 0 (marked in blue), signifying a significant positive correlation with fasting glucose levels. Conversely, 16 markers exhibit CIs entirely below 0 (marked in green), denoting a significant negative correlation with fasting glucose levels. These results showcase that 29 genetic markers out of the total have a statistically significant impact on glucose levels.

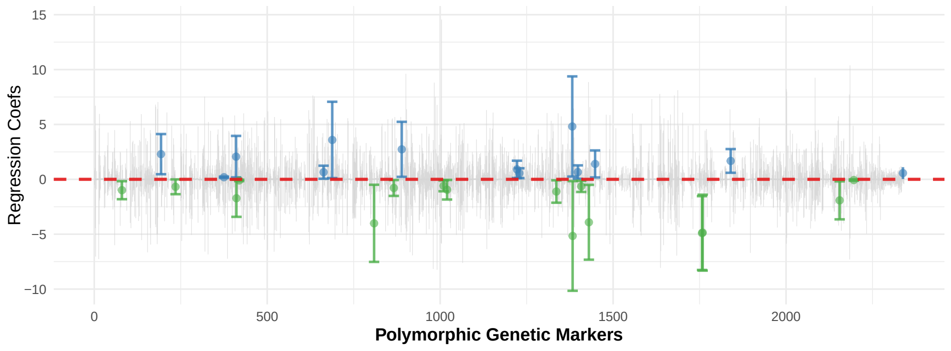

Figure 3: Glucose: Constructed CIs for the first 2341 regression coefficients. Genetic markers represented by blue CIs indicate a significant positive association with the fasting glucose level, whereas those with green CIs demonstrate significant negative associations.

6 Conclusion

There has been significant recent progress in debiasing inference methods for high-dimensional GLMs. This paper highlights the application of advanced debiasing techniques in high-dimensional GLMs using the R package **SIHR**. The package provides tools for estimating bias-corrected point estimators and constructing CIs for various low-dimensional objectives in both one- and two-sample regression settings. Through extensive simulations and real-data analyses, we demonstrate the practicality and versatility of the package across diverse fields of study, making it an essential addition to the literature.

7 Acknowledgement

Prabrisha Rakshit and Zhenyu Wang contributed equally to this work and are considered co-first authors. Dr. Tony Cai's research was supported in part by NSF grant DMS-2015259 and NIH grants R01-GM129781 and R01-GM123056. Dr. Zijian Guo's research was supported in part by NSF grants DMS-1811857 and DMS-2015373 and NIH grants R01-GM140463 and R01-LM013614. Dr. Zijian Guo is grateful to Dr. Lukas Meier for sharing the motif regression data used in this paper.

References

M. A. Beer and S. Tavazoie. Predicting gene expression from sequence. *Cell*, 117(2):185–198, 2004. [p41]

A. Belloni, V. Chernozhukov, and L. Wang. Square-root lasso : pivotal recovery of sparse signals via conic programming. *Biometrika*, 98(4):791–806, 2011. [p27]

P. J. Bickel, Y. Ritov, and A. B. Tsybakov. Simultaneous analysis of lasso and dantzig selector. *The Annals of statistics*, 37(4):1705–1732, 2009. [p28]

P. Bühlmann and S. van de Geer. *Statistics for high-dimensional data: methods, theory and applications*. Springer Science & Business Media, 2011. [p27]

T. Cai, T. Tony Cai, and Z. Guo. Optimal statistical inference for individualized treatment effects in high-dimensional models. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 83 (4):669–719, 2021a. [p27, 28, 29, 30, 31]

T. T. Cai and Z. Guo. Semisupervised inference for explained variance in high dimensional linear regression and its applications. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 82(2):391–419, 2020. [p27]

T. T. Cai, Z. Guo, and R. Ma. Statistical inference for high-dimensional generalized linear models with binary outcomes. *Journal of the American Statistical Association*, pages 1–14, 2021b. [p27, 30]

E. M. Conlon, X. S. Liu, J. D. Lieb, and J. S. Liu. Integrating regulatory motif discovery and genome-wide expression analysis. *Proceedings of the National Academy of Sciences*, 100(6):3339–3344, 2003. [p41]

D. Das, N. Banerjee, and M. Q. Zhang. Interacting models of cooperative gene regulation. *Proceedings of the National Academy of Sciences*, 101(46):16234–16239, 2004. [p41]

J. Fan and R. Li. Variable selection via nonconcave penalized likelihood and its oracle properties. *Journal of the American Statistical Association*, 96:1348–1360, 2011. [p27]

Z. Guo, W. Wang, T. T. Cai, and H. Li. Optimal estimation of genetic relatedness in high-dimensional linear models. *Journal of the American Statistical Association*, 114:358–369, 2019. [p27, 28, 30, 32]

Z. Guo, P. Rakshit, D. S. Herman, and J. Chen. Inference for the case probability in high-dimensional logistic regression. *The Journal of Machine Learning Research*, 22(1):11480–11533, 2021a. [p30]

Z. Guo, C. Renaux, P. Bühlmann, and T. Cai. Group inference in high dimensions with applications to hierarchical testing. *Electronic Journal of Statistics*, 15(2):6633–6676, 2021b. [p27, 29, 30, 31]

Z. Guo, X. Li, L. Han, and T. Cai. Robust inference for federated meta-learning. *arXiv preprint arXiv:2301.00718*, 2023. [p28]

J. Huang and C.-H. Zhang. Estimation and selection via absolute penalized convex minimization and its multistage adaptive applications. *Journal of Machine Learning Research*, 13(Jun):1839–1864, 2012. [p27]

A. Javanmard and A. Montanari. Confidence intervals and hypothesis testing for high-dimensional regression. *The Journal of Machine Learning Research*, 15(1):2869–2909, 2014. [p27, 28]

R. Ma, Z. Guo, T. T. Cai, and H. Li. Statistical inference for genetic relatedness based on high-dimensional logistic regression. *arXiv preprint arXiv:2202.10007*, 2022. [p28, 32]

N. Meinshausen and P. Bühlmann. High-dimensional graphs and variable selection with the lasso. *The annals of statistics*, 34(3):1436–1462, 2006. [p28]

N. Meinshausen and B. Yu. Lasso-type recovery of sparse representations for high-dimensional data. *Annals of Statistics*, 37(1):246–270, 2009. [p27]

S. Negahban, B. Yu, M. J. Wainwright, and P. K. Ravikumar. A unified framework for high-dimensional analysis of *m*-estimators with decomposable regularizers. In *Advances in Neural Information Processing Systems*, pages 1348–1356, 2009. [p27]

T. Sun and C.-H. Zhang. Scaled sparse linear regression. *Biometrika*, 99(4):879–898, 2012. [p27]

R. Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 58(1):267–288, 1996. [p27]

S. van de Geer, P. Bühlmann, Y. Ritov, and R. Dezeure. On asymptotically optimal confidence regions and tests for high-dimensional models. *The Annals of Statistics*, 42:1166–1202, 2014. [p27, 28]

M. J. Wainwright. Sharp thresholds for high-dimensional and noisy sparsity recovery using ℓ_1-constrained quadratic programming (lasso). *IEEE transactions on information theory*, 55(5):2183–2202, 2009. [p28]

Y. Yuan, L. Guo, L. Shen, and J. S. Liu. Predicting gene expression from sequence: a reexamination. *PLoS computational biology*, 3(11):e243, 2007. [p41]

C.-H. Zhang. Nearly unbiased variable selection under minimax concave penalty. *Annals of Statistics*, 38(2):894–942, 2010. [p27]

C.-H. Zhang and S. S. Zhang. Confidence intervals for low dimensional parameters in high dimensional linear models. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 76(1):217–242, 2014. [p27, 28]

P. Zhao and B. Yu. On model selection consistency of lasso. *The Journal of Machine Learning Research*, 7: 2541–2563, 2006. [p28]

Prabrisha Rakshit
Rutgers, The State University of New Jersey
USA
prabrisha.rakshit@rutgers.edu

Zhenyu Wang
Rutgers, The State University of New Jersey
USA
zw425@stat.rutgers.edu

Tony Cai
University of Pennsylvania
USA
tcai@wharton.upenn.edu

Zijian Guo
Rutgers, The State University of New Jersey
USA
zijguo@stat.rutgers.edu