# Learning Conditional Density with high-dimensional covariates

yiming li

December 2023

## 1 Introduction

Genome-wide association studies (GWAS) have identified more than 100,000 single nucleotide polymorphism (SNP) loci associated with a variety of complex human diseases or traits. However, most of the genes identified by GWAS are located in non-coding regions of the human genome, and their functions are unknown, which hinders the investigation of complex disease mechanisms and the utilization of human genetics for the improvement of clinical care. In that way, transcriptome-Wide Association Studies (TWAS) have recently been proposed to combine GWAS and RNA sequence data sets to identify genes associated with complex traits. It is an effective way to enhance meaningful GWAS discoveries, as it aggregates information along the causal pathways from gene to gene expression to phenotype.**ref12**

Typical TWAS pre-focus a relatively small reference set of individuals for whom the gene expression and SNPs information are available and then imputes the cis-genetic component of expression among a much larger set of individuals with known phenotype data from their SNP genotype data. The imputed expression can be viewed as a weighted linear model of genotypes based on the correlation between SNPs and gene expression in the small reference data while considering the linkage disequilibrium (LD) among SNPs. **ref3**

Our proposed method tries to learn potential non-linear and nonparametric TWAS associations between the imputed gene expression data and phenotype data instead of traditional parametric weighted linear association.

## 2 Preliminaries

### 2.1 Problem setting

Suppose in the reference site, we observe n i.i.d. samples $(X_i,\, T_i,\, \boldsymbol{G}_i)$, where the genotypes of the i-th subject on a segment of DNA of interest $\boldsymbol{G}_i \in \mathbb{R}^p$, the gene expression level of the i-th subject in the RNA sequence data $X_i \in \mathbb{R}^1$, and the phenotypes of the i-th subject in a GWAS data $T_i \in \mathbb{R}^1$.

# 3 Estimation Procedure

Naive approaches include imputing gene expression $X_i$ in GWAS using estimated $\mathbb{E}(X \mid \boldsymbol{G}_i)$ in RNA sequence data and assessing the correlation between $T_i$ and imputed $\widetilde{X}_i$. In other words, for naive method, $\boldsymbol{T} = \boldsymbol{\beta}^\top \boldsymbol{X}_{\boldsymbol{G}} + \boldsymbol{\epsilon}_{\boldsymbol{T}}$. In our method, we hope the association can be nonlinear and thus proposed the following basic model, $\boldsymbol{T} = g\left(\boldsymbol{X}_{\boldsymbol{G}}\right) + \boldsymbol{\epsilon}_{\boldsymbol{T}}$, where $\boldsymbol{X}_{\boldsymbol{G}}$ is the component of gene expression attributed to the genetic variants $\boldsymbol{G}, \boldsymbol{\epsilon}_{\boldsymbol{T}}$ is the random error with mean zero, the function $g\left(\boldsymbol{X}_{\boldsymbol{G}}\right)$ is the conditional mean of $\boldsymbol{T}$ given $\boldsymbol{X}_{\boldsymbol{G}}$ without any a priori parametric assumptions. This model is a generalization of the linear TWAS models in the previous literature.

Since $g\left(\boldsymbol{X}_{\boldsymbol{G}}\right)$ is a nonparametric function conditioning on the imputed gene expression level based on the specific segment of genotypes data, we hope to use the numerical approximation method, Monte-Carlo integration, to reflect the property and nature of interest function $g(\cdot)$.

## 3.1 Method

Considering the total probability rule, we have the following equation:

$$f(t \mid \boldsymbol{X}_{\boldsymbol{G}} = x) = \int_G f\left(t \mid \boldsymbol{X}_{\boldsymbol{G}} = x, \boldsymbol{G}\right) f(\boldsymbol{G} \mid \boldsymbol{X}_{\boldsymbol{G}} = x) d\boldsymbol{G}$$

where $f(\cdot)$ represents the probability density function (PDF) of some random variables of interest. $t$ represents some specific value of $\boldsymbol{T}$. $x$ also represents some specific value of $\boldsymbol{X}$. If we consider the Bayesian theorem, we can rewrite the conditional mean $\mathbb{E}[t|\boldsymbol{X}_{\boldsymbol{G}} = x] = \int t f\left(t \mid \boldsymbol{X}_{\boldsymbol{G}} = x\right) dt$ by

$$\mathbb{E}[t|x] = \frac{\iint t f\left(t \mid x, \boldsymbol{G}\right) f\left(x \mid \boldsymbol{G}\right) f(\boldsymbol{G}) d\boldsymbol{G} dt}{\int_{\boldsymbol{G}} f\left(x \mid \boldsymbol{G}\right) f(\boldsymbol{G}) d\boldsymbol{G}} \tag{1}$$

Assuming conditional independence of $\boldsymbol{T}$ and $\boldsymbol{X}_{\boldsymbol{G}}$ given $\boldsymbol{G}$, the equation (1) can be further simplified to

$$\mathbb{E}[t|x] = \frac{\iint t f\left(x \mid \boldsymbol{G}\right) f(t, \boldsymbol{G}) d\boldsymbol{G} dt}{\int_{\boldsymbol{G}} f\left(x \mid \boldsymbol{G}\right) f(\boldsymbol{G}) d\boldsymbol{G}} \tag{2}$$

Assuming the GWAS data $(T_i, \boldsymbol{G}_i)$ is a representative sample, $\mathbb{E}[t|x]$ can be well approximated by Monte-Carlo integration:

$$\mathbb{E}[t|x] \approx \frac{\sum_i T_i f\left(\boldsymbol{X}_{\boldsymbol{G}} = x \mid G_i\right)}{\sum_i f\left(\boldsymbol{X}_{\boldsymbol{G}} = x \mid G_i\right)}$$

where $f\left(\boldsymbol{X}_{\boldsymbol{G}} = x \mid G_i\right)$ is the likelihood of $\boldsymbol{X}_{\boldsymbol{G}} = x$ given genetic variants $\boldsymbol{G}_i$. This way, $g(x)$ is a weighted average of $T_i$, whose weight is its likelihood of having expression level $x$ given $\boldsymbol{G}_i$. The function $g(x)$ can be estimated pointwise over a sequence of $x$'s.

Thus, the key step in our proposed approach is to estimate the conditional density of $\boldsymbol{X_G}|\boldsymbol{G}$. One thing that needs to take care of is that the genotype data $\boldsymbol{G}$ is high-dimensional, and subsegments are highly correlated with each other. The general regression methods might not work well even with standard Lasso penalization. **ref 45** Although the group bridge Lasso and the hierarchical Lasso might better deal with the inner group structure of SNP data, they still have relatively strong requirements on the range of the tuning parameter. Thus we try to apply the decomposition methods to select a subset of SNPs.

Considering the requirements for dimension reduction on $\boldsymbol{G}$, we propose the following joint optimization question of the defined cost function:

$$\text{argmin}_{\boldsymbol{A},\boldsymbol{B},f}\text{Cost} = \|\boldsymbol{G} - \boldsymbol{A}\boldsymbol{B}^\top\| + L(\boldsymbol{X}, f(\boldsymbol{A})) + \Omega(\boldsymbol{A}, \boldsymbol{B})$$

where $\boldsymbol{B}$ defines the feature of SNPs and $\boldsymbol{A}$ is individual mutation loading on that factor or the projection of genotype data on these factors. The first part is the decomposition of the aimed matrix $\boldsymbol{G}$. For the second part related to PDF, a natural way is to consider the Gaussian mixture for the conditional distribution of $\boldsymbol{X} \mid \boldsymbol{G}$. In theory, any continuous distribution can be approximated by a Gaussian mixture with a sufficient number of components. If the genotypes can be approximated by D features, i.e., $\boldsymbol{A}_{n\times D}$, it makes sense to assume the gene expression is also a mixture of $D$ Gaussian distributions, whose means and variances depend on $\boldsymbol{A}_d, d = 1, 2, ..., D$. So we could assume

$$\boldsymbol{X} \mid \boldsymbol{G} \sim \sum_{d=1}^{D} \pi_d \Psi\left(\boldsymbol{\mu}_d\left(\boldsymbol{A}_d\right), \boldsymbol{\sigma}_d\left(\boldsymbol{A}_d\right)\right)$$

Where $\Psi\left(\boldsymbol{\mu}_d\left(A_d\right), \boldsymbol{\sigma}_d\left(A_d\right)\right)$ represents the normal distribution for total n subjects with mean $\boldsymbol{\mu}_d\left(A_d\right)$ and variance $\boldsymbol{\sigma}_d^2\left(A_d\right)$. $\boldsymbol{\mu}_d\left(A_d\right), \boldsymbol{\sigma}_d^2\left(A_d\right) \in \mathbb{R}^n$. In theory, any continuous function can be approximated by the combination of basis functions with a sufficient number of components. We hope $\boldsymbol{\mu}_d\left(A_d\right)$ and $\boldsymbol{\sigma}_d^2\left(A_d\right)$ can be well approximated by a combination of $K$ basis functions:

$$\boldsymbol{A}_d = (A_{1d}, A_{2d}, ..., A_{nd}), \boldsymbol{\mu}_d = (\mu_{1d}, \mu_{2d}, ..., \mu_{nd}), \boldsymbol{\sigma}_d = (\sigma_{1d}, \sigma_{2d}, ..., \sigma_{nd})$$
$$\mu_{id} = \sum_{q=1}^{K} \alpha_q B_q(A_{id}), \sigma_{id} = \sum_{q=1}^{K} \beta_q B_q(A_{id})$$

We hope to measure the similarity between the proposed Gaussian mixture function and the real data. Thus $L(\cdot)$ in the objective function could be the KL divergence of the Gaussian mixture. Besides, the dimensionally reduced genotype data might also be highly correlated. The sparsity of $\boldsymbol{B}$ should be considered. Thus $\Omega(\cdot)$ represents some constraints and penalization on the decomposed matrix $\boldsymbol{A}, \boldsymbol{B}$.

## 3.2 Algorithm

### 3.2.1 One Dimension Case

Consider the simplified case when $D = 1$ (intuitively $\pi = 1$) and the Frobenius norm on the decomposition of $\boldsymbol{G}$ and negative log-likelihood for KL divergence, it is equivalent to minimizing the following cost function:

$$\begin{aligned}
&\mathrm{C} = \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{p}(g_{ij} - a_i b_j)^2 + \sum_{i=1}^{n}\log\sigma_i + \frac{(x_i - \mu_i)^2}{2\sigma_i^2} \\
&\text{subject to} \quad \|B_1\|_2 = \sqrt{\sum_{j}^{p} b_j^2} = 1 \\
&\text{where} \quad \sigma_i = \sum_{q=1}^{K}\beta_q B_q(a_i), \quad \mu_i = \sum_{q=1}^{K}\alpha_q B_q(a_i)
\end{aligned}$$

For the above optimal question, we have four parameter groups $(a_i, b_j, \alpha_q, \beta_q)$ of interest. Since we can not write the closed form of the solution for four parameter groups simultaneously, we apply the component-wise algorithm to update four parameter groups, i.e., update a specific parameter group when keeping the other three groups constant and move forward until all four groups have been updated in this loop. Then start the new loop. For four parameter groups, we initially consider the gradients:

$$\begin{aligned}
\frac{\partial \mathrm{C}}{\partial a_i} &= \sum_{j=1}^{p}(a_i b_j - g_{ij})b_j + \frac{\sigma_i'}{\sigma_i} + \frac{(\mu_i - x_i)(\sigma_i\mu_i' - \sigma_i'\mu_i + \sigma_i' x_i)}{\sigma_i^3} \\
\frac{\partial \mathrm{C}}{\partial \beta_q} &= \sum_{i=1}^{n}\frac{B_q(u_i)}{\sigma_i} - \frac{(x_i - \mu_i)^2 B_q(u_i)}{\sigma_i^3} \\
\frac{\partial \mathrm{C}}{\partial \alpha_q} &= \sum_{i=1}^{n}\frac{(\mu_i - x_i)B_q(\mu_i)}{\sigma_i^2} \\
\frac{\partial \mathrm{C}}{\partial b_j} &= \sum_{i=1}^{n}(b_j a_i - g_{ij})a_i
\end{aligned}$$

For $b_j$ and $\alpha_q$, we can write down the closed form of solutions. $\alpha_q$ is the coefficient of a weighted linear regression.

$$\begin{aligned}
&b_j = \frac{\sum_{i=1}^{n} a_i g_{ij}}{\sum_{i=1}^{n} a_i^2} \quad \alpha_q = \frac{\sum_{i=1}^{n}\frac{(x_i - \mu_i^{-q})B_q(u_i)}{\sigma_i^2}}{\sum_{i=1}^{n}\frac{B_q(u_i)^2}{\sigma_i^2}} \\
&\text{where} \quad \mu_i^{-q} = \sum_{m\neq q}^{K}\alpha_m B_m(a_i).
\end{aligned}$$

However, as we mentioned above feature $\boldsymbol{B}$ should be sparse, we consider the standard Lasso penalization on $b_j$ and get the following solutions:

$$b_j^{lasso} = \text{sgn}(b_j^*)(|b_j^*| - \frac{\lambda}{\sum_{i=1}^n a_i^2})^+, \quad \text{where } b_j^* = \frac{\sum_{i=1}^n a_i g_{ij}}{\sum_{i=1}^n a_i^2}$$

For $a_i$ and $\beta_q$, it is hard to write out the closed form of solutions since each of these two equations contains a summation of fractions whose denominator ($\sigma_i$) is the combination of $B_q(a_i)$ or $\beta_q$. Thus, we consider the gradient descent or golden searching methods to approximate the optimal value of $\beta_q$. **ref6** For $a_i$, we refer to XXX's method and propose the following explicit solution:

$$a_i = \frac{\sum_{j=1}^p g_{ij} b_j - \frac{\sigma_i'}{\sigma_i} + \frac{(\mu_i - x_i)(\sigma_i \mu_i' - \sigma_i' \mu_i + \sigma_i' x_i)}{\sigma_i^3}}{\sum_{j=1}^p b_j^2}$$

Where we fix $\mu_i, \sigma_i, \mu_i', \sigma_i'$ even if they contain $a_i$, and we can get the same optimal value.

### 3.2.2 Additive Residual Model For The Multiple Layers Case

One common idea in regression is that the simpler the model, the better. Thus we always start with the case when the dimension is one and increase the number of layers step by step. Since the genotype data is high-dimensional, the computation time would be hugely prolonged if we always refresh the previous updated value and rebuild the model. One possible solution might be residual regression. Suppose we have obtained the information for the first layer using the algorithm mentioned in the one dimension case section, and we now want to add the second layer. The residual regression method concludes that we can regress the residual (the difference between the observed value and predicted value) on the added predictors. In our case, We fix the previously updated coefficients, $\boldsymbol{A}_{\text{pre}}, \boldsymbol{B}_{\text{pre}}, \boldsymbol{\alpha}_{\text{pre}}, \boldsymbol{\beta}_{\text{pre}}$ where pre represents previous updated value. Then we only update the residual of $\boldsymbol{G}$: $\boldsymbol{G}_{\text{res}} = \boldsymbol{G}_{\text{pre}} - \boldsymbol{A}_{\text{pre}} \boldsymbol{B}_{\text{pre}}^\top$ where res represents residual. And we also need to refresh the $\Gamma$ matrix, which represents the proportion of a specific subject belonging to each normal distribution in mixture Gaussian mode, just like what we do in the EM algorithm for the mixture Gaussian model.

To include as much information on genotype data $\boldsymbol{G}$ as possible, we hope $\boldsymbol{B}_{\text{new}}$ should be orthogonal with all the previous $\boldsymbol{B}_{\text{pre}}$. We realize this by multiplying $\boldsymbol{B}_{\text{new}}$ by a matrix $\boldsymbol{I}_p - \boldsymbol{B}_{\text{pre}} \left( \boldsymbol{B}_{\text{pre}}^\top \boldsymbol{B}_{\text{pre}} \right)^{-1} \boldsymbol{B}_{\text{pre}}^\top$ to project $\boldsymbol{B}_{\text{new}}$ on the orthogonal space of $\boldsymbol{B}_{\text{pre}}$.

In each added layer, all the updates are based on $G_{\text{res}}$, and repeat the algorithm for dimension one case. The only thing we need to change is the $\Gamma$ matrix. We also set the stop criterion for the algorithm: stop if the complete loss(aim loss) does not have a 10% decrease.

### 3.2.3 Algorithm Process

**Step 1:** Start with D = 1

- s1 update $a_i$ for i = 1:n
- s2 update $b_j$ for j = 1:p with tuning parameter $\lambda$
- s3 update $\alpha_q$ for q = 1:K
- s4 update $\beta_q$ for q = 1:K using gold searching
- s5 calculate $\Gamma$ matrix for Gaussian mixture model
- s6 compare where the 2-norms of difference for $\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{\alpha}, \boldsymbol{\beta}$ are smaller than the prespecified value. If yes, calculate the cost and move forward to **Step 2** D = D+1. If not, go back to s1

**Step 2:** $\boldsymbol{G} = \boldsymbol{G} - \boldsymbol{A}\boldsymbol{B}^\top$

- ss1 update $a_i$ for i = 1:n
- ss2 update $b_j$ for j = 1:p with tuning parameter $\lambda$, multiply $\boldsymbol{B}_{\text{new}}$ by $\boldsymbol{I}_p - \boldsymbol{B}\left(\boldsymbol{B}^\top\boldsymbol{B}\right)^{-1}\boldsymbol{B}^\top$.
- ss3 update $\alpha_q$ for q = 1:K
- ss4 update $\beta_q$ for q = 1:K using gold searching
- ss5 calculate $\Gamma$ matrix for Gaussian mixture model
- ss6 compare where the 2-norms of difference for $\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{\alpha}, \boldsymbol{\beta}$ are smaller than the prespecified value. If not, go back to ss1. If yes, calculate the cost and move forward to ss7.
- ss7 If new cost decrease more than 10%, repeat **Step 2**, D = D+1. If not, Stop

### 3.2.4 One Dimension Case for multivariate response

$$\mathrm{C} = \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{p}(g_{ij} - a_i b_j)^2 + \sum_{m=1}^{M}(\sum_{i=1}^{n}\log\sigma_i^m + \frac{(x_i^m - \mu_i^m - \boldsymbol{\eta}_m \boldsymbol{x}_i^{-m})^2}{2(\sigma_i^m)^2})$$

$$\text{subject to} \quad \|B_1\|_2 = \sqrt{\sum_{j}^{p} b_j^2} = 1$$

$$\text{where} \quad \sigma_i^m = \sum_{q=1}^{K}\beta_q^m B_q(a_i), \quad \mu_i^m = \sum_{q=1}^{K}\alpha_q^m B_q(a_i)$$

$$\boldsymbol{\eta}_m = (\eta_m^1, ..., \eta_m^{m-1}, \eta_m^{m+1}, \eta_m^M)$$

For the above optimal question, we have four parameter groups $(a_i, b_j, \alpha_q^m, \beta_q^m, \eta_m)$ of interest. Since we can not write the closed form of the solution for four parameter groups simultaneously, we apply the component-wise algorithm to update

four parameter groups, i.e., update a specific parameter group when keeping the other three groups constant and move forward until all four groups have been updated in this loop. Then start the new loop. For four parameter groups, we initially consider the gradients:

$$\frac{\partial \mathrm{C}}{\partial a_i} = \sum_{j=1}^{p}(a_i b_j - g_{ij})b_j$$
$$+ \sum_{m=1}^{M} \frac{\sigma_i^{m'}}{\sigma_i^m} + \frac{(\mu_i^m + \boldsymbol{\eta}_m \boldsymbol{x}_i^{-m} - x_i^m)(\sigma_i^m \mu_i^{m'} - \sigma_i^{m'} \mu_i^m + \sigma_i^{m'} x_i - \sigma_i^{m'} \boldsymbol{\eta}_m \boldsymbol{x}_i^{-m})}{\sigma_i^{m3}}$$
$$\frac{\partial \mathrm{C}}{\partial \beta_q^m} = \sum_{i=1}^{n} \frac{B_q(u_i^m)}{\sigma_i^m} - \frac{(x_i^m - \mu_i^m - \boldsymbol{\eta}_m \boldsymbol{x}_i^{-m})^2 B_q(u_i^m)}{\sigma_i^{m3}}$$
$$\frac{\partial \mathrm{C}}{\partial \alpha_q^m} = \sum_{i=1}^{n} \frac{(\mu_i^m + \boldsymbol{\eta}_m \boldsymbol{x}_i^{-m} - x_i^m) B_q(\mu_i^m)}{\sigma_i^{m2}}$$
$$\frac{\partial \mathrm{C}}{\partial \eta_m^l} = \sum_{i=1}^{n} \frac{(\mu_i^m + \boldsymbol{\eta}_m \boldsymbol{x}_i^{-m} - x_i^m)(x_i^{-m})^l}{\sigma_i^{m2}}$$
$$\frac{\partial \mathrm{C}}{\partial b_j} = \sum_{i=1}^{n}(b_j a_i - g_{ij})a_i$$

### 3.3 multi-response and its test

A specific gene sequence might be responsible for multiple expression results in the real case. Thus, we should consider the multi-response case. We still hope $X \mid \boldsymbol{G}$ follows the mixture multivariate normal distribution, where the number of mixed distributions equals to the dimension of decomposed genotype data. Remember we have used the notation $d$ to represent the dimension of decomposed genotype data. Thus,

$$\boldsymbol{X} = I(i=1) \cdot \boldsymbol{X}^{(1)} + I(i=2) \cdot \boldsymbol{X}^{(2)} + \ldots + I(i=d) \cdot \boldsymbol{X}^{(d)}$$

Where each $\boldsymbol{X}^{(d^*)}$ follows the multivariate normal distribution. If we still consider the residual regression framework, we only need to update the newest $\boldsymbol{X}$. To reduce the complexity of notation, we denote the newest $\boldsymbol{X} = (\boldsymbol{X}_1, ..., \boldsymbol{X}_m)$.

Since all the expression results are transcribed from the same sequence, they should be naturally correlated, which means that the covariance matrix of the multivariate normal distribution should not be diagonal. If we consider the structure of the covariance, the computation would be complex. Thus, we consider the following conditional model:

$$\boldsymbol{X}_{m^*}^{(d^*)} \sim \mathcal{N}\left(\Theta_{m^*}^{(d^*)\top} g(\boldsymbol{A}_{d^*}) + \eta_{m^*}^{(d^*)\top} \boldsymbol{X}_{-m^*}, \sigma_{m^*}^2\right)$$

The above form can be rewritten as:

$$\boldsymbol{X}^{(d^*)} \sim \mathcal{N}\left(\begin{pmatrix} {\Theta_1^{(d^*)'}}^\top g(\boldsymbol{A}_{d^*}) \\ ... \\ {\Theta_m^{(d^*)'}}^\top g(\boldsymbol{A}_{d^*}) \end{pmatrix}, \Sigma\right)$$

where

$$\Sigma = \begin{pmatrix} (\sigma_1')^2 & \sigma_1'\sigma_2'\rho & \dots & \sigma_1'\sigma_m'\rho \\ \sigma_2'\sigma_1'\rho & (\sigma_2')^2 & \dots & \sigma_2'\sigma_m'\rho \\ & & ... & \\ \sigma_m'\sigma_1'\rho & \sigma_m'\sigma_2'\rho & \dots & (\sigma_m')^2 \end{pmatrix}$$

$$\begin{pmatrix} \Theta_1^{(d^*)} \\ \Theta_2^{(d^*)} \\ ... \\ \Theta_m^{(d^*)} \end{pmatrix} = \begin{pmatrix} 1 & -\eta_{1,1}^{(d^*)} & \dots & -\eta_{1,m-2}^{(d^*)} & -\eta_{1,m-1}^{(d^*)} \\ -\eta_{2,1}^{(d^*)} & 1 & \dots & -\eta_{2,m-2}^{(d^*)} & -\eta_{2,m-1}^{(d^*)} \\ & & ... & & \\ -\eta_{m,1}^{(d^*)} & -\eta_{m,2}^{(d^*)} & \dots & -\eta_{m,m-1}^{(d^*)} & 1 \end{pmatrix} \cdot \begin{pmatrix} \Theta_1^{(d^*)'} \\ \Theta_2^{(d^*)'} \\ ... \\ \Theta_m^{(d^*)'} \end{pmatrix}$$

$$\begin{pmatrix} \sigma_1 \\ \sigma_2 \\ ... \\ \sigma_m \end{pmatrix} = \rho \begin{pmatrix} 1 & -\eta_{1,1}^{(d^*)} & \dots & -\eta_{1,m-2}^{(d^*)} & -\eta_{1,m-1}^{(d^*)} \\ -\eta_{2,1}^{(d^*)} & 1 & \dots & -\eta_{2,m-2}^{(d^*)} & -\eta_{2,m-1}^{(d^*)} \\ & & ... & & \\ -\eta_{m,1}^{(d^*)} & -\eta_{m,2}^{(d^*)} & \dots & -\eta_{m,m-1}^{(d^*)} & 1 \end{pmatrix} \cdot \begin{pmatrix} \sigma_1' \\ \sigma_2' \\ ... \\ \sigma_m' \end{pmatrix}$$

# 4 Result

## 4.1 Simulation

We use data GTEx_Y_DDX11 for testing. There are 670 subjects and 3909 SNPs are recorded for each subject. We are interested in the performance of the algorithm in two aspects. First, whether our proposed method could outperform the Generalized Linear Model (GLM) and Neural Network (NN). Intuitively, we hope the proposed method should at least be better than the GLM since GLM simply neglects the high correlation and inner clustering structure in the genotype data. One common problem for NN is that the training sample size should as large as possible while the genotype data and expression level data are relatively scarce due to the difficulty of sampling and high cost. In this case, we believe our proposed algorithm should be also better than NN method. Second, we want compare the performance of our proposed algorithm in homogeneous and heterogeneous case.

Since the length of SNPs is pretty long, we assumed that 0.01 of they truly cause the expression level, and defined this subset of SNPs the causal part $\boldsymbol{G}_c$. The position of casual part are mutually random. For homogeneous case, we simulated the expression level of each subject under the assumption that expression level is the linear combination of the causal part with the coefficient $\boldsymbol{\gamma_{\text{homo}}}$.

$$\boldsymbol{X} = {\boldsymbol{\gamma_{\text{homo}}}}^\top \boldsymbol{G}_c + \boldsymbol{\epsilon}_1$$

For heterogeneous case, we simulated the expression level of each subject assuming that expression levels are also correlated with the interaction term of random error and causal part with the coefficient $\boldsymbol{\gamma_{\text{hete}}}$.

$$\boldsymbol{X} = \boldsymbol{\gamma_{\text{hete}}}^\top \boldsymbol{G}_c + \frac{1}{2}\boldsymbol{\gamma_{\text{hete}}}^\top \boldsymbol{G}_c * \boldsymbol{\epsilon}_1 + \boldsymbol{\epsilon}_1$$

**cleaning on $\boldsymbol{G}$**

We test our algorithm on the simulated homogeneous data and heterogeneous data. Training and testing data sets are separated with a probability of 0.7. Tuning parameter $\lambda$ of Lasso in the update of $\boldsymbol{B}$ are considered to be 0, 100, 125, 150. We consider the three types of measurements. For visualization we consider the QQplot. To better evaluate the algorithm, we apply the bootstrap method to repeat the partition and algorithm for 20 time. Within each repetition, KL divergence (KL) and Wasserstein distance (WD) are considered.

QQplot

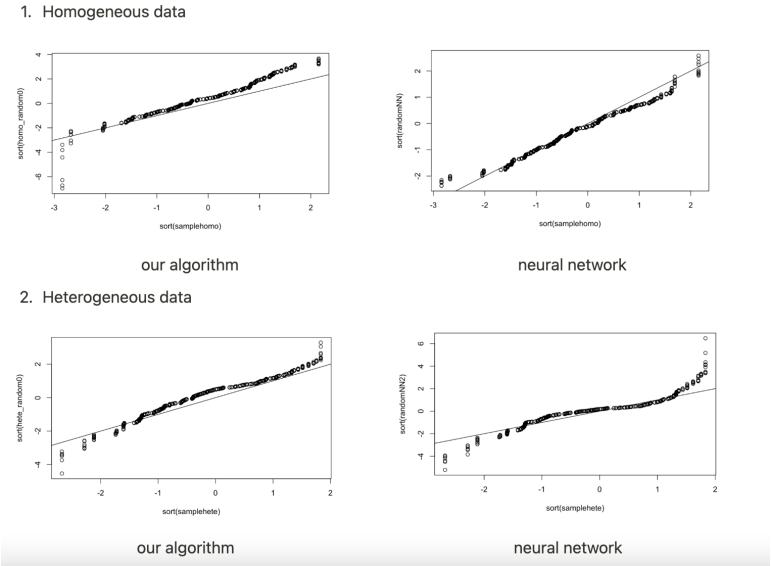


Figure 1: QQplot

KL and WD

We conclude that NN performs better on homogeneous data than on heterogeneous data, our algorithm seems to perform better on heterogeneous data. Our algorithm slightly outperforms neural networks on heterogeneous but performs worse on homogeneous data.

| | homo | homo | hetero | hetero |
|---|---|---|---|---|
| | algorithm | neural network | algorithm | neural network |
| KL divergence | 2.153113 | 2.032694 | 1.950047 | 2.071552 |
| Wasserstein distance | 0.4933297 | 0.1311285 | 0.3047337 | 0.3044585 |

Figure 2: QQplot

### 4.2 Real Data

## 5 Theory

### 5.1 Approximation

Gaussian mixture and B spline function

### 5.2 Asymptotic

$\alpha_q$ and $\beta_q$ matrix decomposition

## 6 Reference

From GWAS to Gene: Transcriptome-Wide Association Studies and Other Methods to Functionally Understand GWAS Discoveries

A gene-based association method for mapping traits using reference transcriptome data

Integrative approaches for large-scale transcriptome-wide association studies

A group bridge approach for variable selection

Group Variable Selection via a Hierarchical Lasso and Its Oracle Property

Sufficient direction factor model and its application to gene expression quantitative trait loci discovery