UNIVERSITY OF CALIFORNIA

Los Angeles

Statistical methods for molecular quantitative trait locus analysis

A dissertation submitted in partial satisfaction

of the requirements for the degree

Doctor of Philosophy in Statistics

by

Heather Zhou

2023

ABSTRACT OF THE DISSERTATION

Statistical methods for molecular quantitative trait locus analysis

by

Heather Zhou
Doctor of Philosophy in Statistics
University of California, Los Angeles, 2023
Professor Jingyi Li, Chair

Molecular quantitative trait locus (molecular QTL, henceforth “QTL”) analysis investigates the relationship between genetic variants and molecular traits, helping explain findings in genome-wide association studies. This dissertation addresses two major problems in QTL analysis: hidden variable inference problem and eGene identification problem.

Estimating and accounting for hidden variables is widely practiced as an important step in QTL analysis for improving the power of QTL identification. However, few benchmark studies have been performed to evaluate the efficacy of the various methods developed for this purpose. In my first project, I benchmark popular hidden variable inference methods including surrogate variable analysis (SVA), probabilistic estimation of expression residuals (PEER), and hidden covariates with prior (HCP) against principal component analysis (PCA)—a well-established dimension reduction and factor discovery method—via 362 synthetic and 110 real data sets. I show that PCA not only underlies the statistical methodology behind the popular methods but is also orders of magnitude faster, better performing, and much easier to interpret and use. To help researchers use PCA in their QTL analysis, I provide an R package PCAForQTL along with a detailed guide, both of which are available at `https://github.com/heatherjzhou/PCAForQTL`. I believe that using PCA rather than SVA, PEER, or HCP will substantially improve and simplify hidden variable inference in QTL mapping as well as increase the transparency and reproducibility of QTL research.

A central task in expression quantitative trait locus (eQTL) analysis is to identify cis-eGenes (henceforth "eGenes"), i.e., genes whose expression levels are regulated by at least one local genetic variant. Among the existing eGene identification methods, FastQTL is considered the gold standard but is computationally expensive as it requires thousands of permutations for each gene. Alternative methods such as eigenMT and TreeQTL have lower power than FastQTL. In my second project, I propose ClipperQTL, which reduces the number of permutations needed from thousands to 20 for data sets with large sample sizes ($>$ 450) by using the contrastive strategy developed in Clipper; for data sets with smaller sample sizes, it uses the same permutation-based approach as FastQTL. I show that ClipperQTL performs as well as FastQTL and runs about 500 times faster if the contrastive strategy is used and 50 times faster if the conventional permutation-based approach is used. The R package ClipperQTL is available at `https://github.com/heatherjzhou/ClipperQTL`. This project demonstrates the potential of the contrastive strategy developed in Clipper and provides a simpler and more efficient way of identifying eGenes.

The dissertation of Heather Zhou is approved.

Wei Li

Chad J. Hazlett

Mark S. Handcock

Jingyi Li, Committee Chair

University of California, Los Angeles

2023

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGMENTS

This work is funded in part by NSF DGE-1829071 and NIH/NHLBI T32HL139450.

I would like to thank my advisor, Dr. Jingyi Jessica Li, who trusted me, always made herself available, and continually inspires me with her passion for research.

I would like to thank the professors who taught me at my undergraduate institution, Swarthmore College, including Dr. Steve Wang, Dr. Kelly McConville, and Dr. Charles Grinstead. They taught me tirelessly and instilled in me the principles of mathematics and statistics.

I am also very grateful for my collaborators Dr. Wei Li from UC Irvine (who is also on my committee), Dr. Yumei Li from UC Irvine, Dr. Lei Li from Shenzhen Bay Laboratory, and Dr. Xinzhou Ge from UCLA. Dr. Wei Li, Dr. Yumei Li, and Dr. Lei Li helped introduce me to the field of molecular quantitative trait locus analysis and gave me essential help in my first project. Dr. Xinzhou Ge played a crucial role in conceiving my second project.

In addition, I would like to thank my committee members Dr. Mark Handcock, Dr. Chad Hazlett, Dr. Wei Li, and Dr. Janet Sinsheimer for their time and support. Dr. Janet Sinsheimer unfortunately passed away recently. She was a bright presence in my life even though we had limited interactions.

Last but not least, I would like to thank my family and peers, without whom I could not have completed this dissertation.

VITA

2012–2016	B.A. with majors in mathematics and economics, Swarthmore College, Swarthmore, Pennsylvania.
2018–2022	M.S. in statistics, University of California, Los Angeles, Los Angeles, California.
2018–2020	Recipient of Graduate Dean's Scholar Award, University of California, Los Angeles, Los Angeles, California.
2019–2020	Recipient of NSF DGE-1829071 Award as part of the UCLA Modeling and Understanding Human Behavior (MENTOR) training program.
2020–2021	Teaching assistant, University of California, Los Angeles, Los Angeles, California.
2021–2023	Recipient of NIH/NHLBI T32HL139450 Award as part of the UCLA Integrated Data Science in Cardiovascular Medicine (iDISCOVER) training program.
2023–2023	Graduate Student Researcher, University of California, Los Angeles, Los Angeles, California.

PUBLICATIONS

Heather J. Zhou, Xinzhou Ge, and Jingyi Jessica Li. ClipperQTL: ultrafast and powerful eGene identification method. bioRxiv, 2023.

Heather J. Zhou, Lei Li, Yumei Li, Wei Li, and Jingyi Jessica Li. PCA outperforms popular hidden variable inference methods for molecular QTL mapping. Genome Biology, 23(1):210, 2022.

Heather J. Zhou. Capturing hidden covariates with linear factor models and other statistical methods in differential gene expression and expression quantitative trait locus studies. UCLA Electronic Theses and Dissertations, 2022.

Yidan Eden Sun, **Heather J. Zhou**, Jingyi Jessica Li. Bipartite tight spectral clustering (BiTSC) algorithm for identifying conserved gene co-clusters in two species. Bioinformatics, 37(9):1225-1233, 2021.

Steve C. Wang, Philip J. Everson, **Heather J. Zhou**, Dasol Park, David J. Chudzicki. Adaptive credible intervals on stratigraphic ranges when recovery potential is unknown. Paleobiology, 42(2):240-256, 2016.

CHAPTER 1

Introduction

1.1 Hidden variable inference problem

Genome-wide association studies (GWASs) have identified thousands of genetic variants associated with human traits or diseases [21–24]. However, the majority of GWAS variants are located in non-coding regions of the genome, making it challenging to interpret the GWAS associations [25, 26]. In response to this, molecular quantitative trait locus (molecular QTL, henceforth "QTL") analysis has emerged as an important field in human genetics, interrogating the relationship between genetic variants and intermediate, molecular traits and potentially explaining GWAS findings [27, 28].

Based on the type of molecular phenotype studied, QTL analyses can be categorized into gene expression QTL (eQTL) analyses [3, 4], alternative splicing QTL (sQTL) analyses [4], three prime untranslated region alternative polyadenylation QTL ($3'$aQTL) analyses [2], and so on [27, 28]. Among these categories, eQTL analyses, which investigate the association between genetic variants and gene expression levels, are the most common. To date, most (single-tissue) QTL studies are carried out using regression-based methods such as Matrix eQTL [29] and FastQTL [17].

In QTL analysis, a major challenge is that measurements of gene expression levels and other molecular phenotypes can be affected by a number of technical or biological variables other than the genetic variants, such as batch, sex, and age. If these variables are known, then they can be directly included in the QTL pipeline as covariates. However, many of these variables may be unknown or unmeasured. Therefore, it has become standard practice to *first* infer the hidden variables and *then* include the inferred variables as covariates or otherwise account for them in the QTL pipeline [2–4, 30–39] (see Section 2.2.3 for a numerical example). This type of approach has been shown to both improve the power of QTL identification in simulation settings [5] and empirically increase the number of discoveries in QTL studies [3, 4, 32, 37–39].

Surrogate variable analysis (SVA) [13, 14] is one of the first popular hidden variable inference methods for large-scale genomic analysis. Although initially proposed as a hidden variable inference method for both QTL mapping and differential expression (DE) analysis, currently SVA is primarily used in DE and similar analyses as opposed to QTL mapping [40–43]. We believe this is partly because the SVA package [44] is difficult to apply in QTL settings in that it requires the user to input at least one variable of interest and using too many variables of interest causes the package to fail (Figure 2.1; Section 3.4); while there are usually at most a few variables of interest in a DE study, there are often millions of single nucleotide polymorphisms (SNPs; variables of interest) in a QTL study. Historically, there have been two versions of the SVA method: two-step SVA [13] and iteratively reweighted SVA (IRW-SVA) [14]; the latter supersedes the former. Therefore, we focus on IRW-SVA in this work.

Probabilistic estimation of expression residuals (PEER) [1, 5] is currently the most popular hidden variable inference method for QTL mapping by far. It is used in the Genotype-Tissue Expression (GTEx) project [3, 4] and many other high-impact studies [2, 30–37]. The PEER method has two main perceived advantages: (1) it can take known covariates into account when estimating the hidden covariates, and (2) its performance does not deteriorate as the number of inferred covariates increases (i.e., it does not "overfit"). One drawback of PEER, though, is that there is no consensus in the literature on *how* it should be used. For example, when there are known covariates available, PEER can be run with or without the known covariates—Stegle et al. [1] do not give an explicit recommendation as to which approach should be used, and both approaches are used in practice (e.g., [3, 4] vs. [2, 32]). Further, PEER outputs both inferred covariates and residuals of the inputted molecular phenotypes (Figure 2.1), so the user needs to decide which set of outputs to use (Section 3.4; we refer to the approach using the inferred covariates as the "factor approach" and the approach using the residuals as the "residual approach"). Such "flexibility" of PEER could be considered a benefit, but we believe it not only leads to confusion for practitioners who try to use the method but also reduces the transparency and reproducibility of published QTL research.

Hidden covariates with prior (HCP) [16] is another popular hidden variable inference method for QTL mapping. Though less popular than PEER, it has also been used in some high-impact studies [38, 39]. To determine which method is the best and whether PEER indeed has the perceived advantages, we thoroughly evaluate SVA, PEER, and HCP for the first time in the literature. Given that principal component analysis (PCA) [6–10] underlies the methodology behind each of these methods (Section 2.1.4) and has indeed been used for the same purpose [45, 46], we also include PCA in our evaluation. Through simulation studies (Section 2.1.1) and real data analysis (Sections 2.1.2, 2.1.3 and 2.1.5), we show that PCA is orders of magnitude faster, better-performing, and much easier to interpret and use (Figure 2.1).

1.2 eGene identification problem

Molecular quantitative trait locus (molecular QTL, henceforth "QTL") analysis investigates the relationship between genetic variants and molecular traits, helping explain findings in genome-wide association studies [27, 28]. Based on the type of molecular phenotype studied, QTL analyses can be categorized into gene expression QTL (eQTL) analyses [3, 4], alternative splicing QTL (sQTL) analyses [4], three prime untranslated region alternative polyadenylation QTL ($3'$aQTL) analyses [2], and so on [27, 28]. Among these categories, eQTL analyses, which investigate the association between genetic variants and gene expression levels, are the most common. Therefore, in this work, we focus on eQTL analyses as an example, although everything discussed in this work is applicable to other types of QTL analyses as well.

A central task in eQTL analysis is to identify cis-eGenes (henceforth "eGenes"), i.e., genes whose expression levels are regulated by at least one local genetic variant. This presents a multiple-testing challenge as not only are there many candidate genes, each gene can have up to tens of thousands of local genetic variants, and the local genetic variants are often in linkage disequilibrium (i.e., associated) with one another.

Existing eGene identification methods include FastQTL [17], eigenMT [47], and TreeQTL [48]. All three methods share the same two-step approach: first, obtain a gene-level p-value for each gene; second, apply a false discovery rate (FDR) control method on the gene-level p-values to call eGenes. The key difference between the three methods lies in how the gene-level p-values are obtained.

Among the existing eGene identification methods, FastQTL [17] is considered the gold standard and is currently the most popular. It uses permutations to obtain gene-level p-values. There are four main ways to use FastQTL, depending on (1) whether the direct or the adaptive permutation scheme is used and (2) whether proportions or beta approximation is used (Table 2.1). The default way of using FastQTL is to use the adaptive permutation scheme with beta approximation [4, 17]. The adaptive permutation scheme means the number of permutations is chosen adaptively for each gene (between 1000 and 10,000 by default [4, 17]); the beta approximation helps produce higher-resolution gene-level p-values given the numbers of permutations (Algorithm 5). The main drawback of FastQTL is the lack of computational efficiency as it requires thousands of permutations for each gene. A faster implementation of FastQTL named tensorQTL has been developed [49], but it relies on graphics processing units (GPUs), which are not universally available.

eigenMT [47] and TreeQTL [48] have been proposed as faster alternatives to FastQTL.

Neither method uses permutations. In a nutshell, eigenMT uses Bonferroni correction to calculate a gene-level p-value for each gene but estimates the *effective* number of local genetic variants for each gene by performing a principal component analysis (conceptually speaking; instead of using the *actual* number of local genetic variants). On the other hand, TreeQTL uses Simes' rule [50] to calculate a gene-level p-value for each gene. Our analysis shows that both eigenMT and TreeQTL have lower power than FastQTL (Figures 4.1 and 4.3).

Clipper [51] is a p-value-free FDR control method. Given a large number of features (e.g., genes), a number of measurements under the experimental (e.g., treatment) condition, and a number of measurements under the background (e.g., control) condition, Clipper works as the following: first, obtain a contrast score for each feature based on the experimental and background measurements (for example, the contrast score may be the average experimental measurement minus the average background measurement); second, given a target FDR (e.g., 0.05), obtain a cutoff for the contrast scores; lastly, call the features with contrast scores above the cutoff as discoveries. The idea is that the contrast scores of the uninteresting features (e.g., genes whose expected expression levels are *not* increased by the treatment) will be roughly symmetrically distributed around zero, and the outlying contrast scores in the right tail likely belong to interesting features. Notably, Clipper produces a q-value for each feature (similar to Storey's q-values [52]), so that the features can be ranked from the most significant to the least significant.

In this work, we propose ClipperQTL for eGene identification [53], which reduces the number of permutations needed from thousands to 20 for data sets with large sample sizes (> 450) by using the contrastive strategy developed in Clipper; for data sets with smaller sample sizes, it uses the same permutation-based approach as FastQTL. Unlike tensorQTL, our ClipperQTL software does not rely on GPUs. We show that ClipperQTL performs as well as FastQTL and runs about 500 times faster if the contrastive strategy is used and 50 times faster if the conventional permutation-based approach is used (we refer to the two variants of ClipperQTL as the Clipper variant and the standard variant, respectively; Section 4.2.2).

CHAPTER 2

PCA outperforms popular hidden variable inference methods for molecular QTL mapping

2.1 Results

2.1.1 Comprehensive simulation studies show that PCA is faster and better-performing

We compare the runtime and performance of 15 methods (Table 2.1), including Ideal (assuming the hidden covariates are known), Unadjusted (not estimating or accounting for the hidden covariates), and 13 variants of PCA, SVA, PEER, and HCP, based on two simulation studies. In the first simulation study (Simulation Design 1; Section 3.2), we follow the data simulation in Stegle et al. [5]—the original PEER publication—while addressing its data analysis and overall design limitations (Section 3.1). In the second simulation study (Simulation Design 2; Section 3.3), we further address the data simulation limitations of Stegle et al. [5] (Section 3.1) by simulating the data in a more realistic and comprehensive way, roughly following Wang et al. [20]—the SuSiE publication—but introducing the existence of known and hidden covariates. A summary of the main differences between the two simulation designs is provided in Table 3.1. The key difference is that in Simulation Design 1, the gene expression levels are primarily driven by trans-regulatory effects rather than cis-regulatory effects or covariate effects (Table 3.2), inconsistent with the common belief that trans-regulatory effects are generally weaker than cis-regulatory effects. In contrast, in Simulation Design 2, we focus on cis-QTL detection and carefully control the genotype effects and covariate effects in 176 experiments with two replicates per experiment (Section 3.3).

The details of the 15 methods are described in Section 3.4, and the evaluation metrics are described in Section 2.2.1. For convenience, we refer to the simulated molecular phenotypes as gene expression levels throughout our simulation studies; however, they can be interpreted as any type of molecular phenotype after data preprocessing and transformation, e.g., alternative splicing phenotypes and alternative polyadenylation phenotypes (Table 3.3).

The results from our simulation studies are summarized in Figures 2.2, 2.3, 3.1, 3.3, and 3.4. We find that PCA and HCP are orders of magnitude faster than SVA, which in turn is orders of magnitude faster than PEER, and that PCA outperforms SVA, PEER, and HCP in terms of the area under the precision-recall curve (AUPRC) of the QTL result (Figures 2.2 and 2.3). On a dataset-by-dataset basis, PCA outperforms the other methods in terms of AUPRC in 11% to 88% of the simulated data sets and underperforms them in close to 0% of the simulated data sets in Simulation Design 2 (Figure 3.3(d)). In addition, PCA has the highest average concordance scores, a metric for the concordance between the true hidden covariates and the inferred covariates (Section 2.2.1; Figures 3.1 and 3.4), which explains why PCA performs the best in terms of AUPRC.

To contrast the results in Stegle et al. [5], we also compare the powers of the different methods in Simulation Design 1 (Figure 3.1). We find that PCA is more powerful than SVA, PEER, and HCP. Notably, SVA and PEER have very low power in identifying trans-QTL relations—an especially unfavorable result for SVA and PEER, considering that the gene expression levels are primarily driven by trans-regulatory effects in Simulation Design 1 (Table 3.2).

Incidentally, Figures 2.2 and 3.3 also provide us with the following insights into the different ways of using PEER (Section 3.4). First, running PEER *with* the known covariates has no advantage over running PEER *without* the known covariates in terms of AUPRC, given the choice of K (the number of inferred covariates) and the choice between the factor approach and the residual approach. In fact, running PEER *with* the known covariates significantly increases the runtime of PEER in real data (Section 2.1.3). Second, contrary to claims in Stegle et al. [1, 5], the performance of PEER *does* deteriorate as the number of PEER factors increases. The only exception is when the residual approach is used in Simulation Design 1 (Figure 2.2). But given that Simulation Design 2 is more realistic than Simulation Design 1 and that the factor approach is more popular than the residual approach [2–4, 33–36], the take-home message should be that in general, the performance of PEER is worse when we use a large K rather than the true K. Third, whether the factor approach or the residual approach performs better depends on the choice of K. When we use the true K, the factor approach performs better, but when we use a large K, the residual approach performs better. All in all, PCA outperforms all different ways of using PEER in both of our simulation studies (Figure 2.2).

2.1.2 PEER factors sometimes fail to capture important variance components of the molecular phenotype data

For our real data analysis, we examine the most recent GTEx eQTL and sQTL data [4] (Sections 2.1.3 and 2.1.5) and the $3'$aQTL data prepared by Li et al. [2] from GTEx RNA-seq reads [3] (Section 2.1.2). While the exact data analysis pipelines are different (Table 3.3), these studies all choose PEER as their hidden variable inference method.

Unlike PCs, which are always uncorrelated (Section 3.5.1), PEER factors are not guaranteed to be uncorrelated. Here we show through the above-mentioned $3'$aQTL data that PEER factors can be highly correlated with each other (to the extent that many or all of them are practically identical) and thus fail to capture important variance components of the molecular phenotype data.

Given a post-imputation alternative polyadenylation phenotype matrix (each entry is between zero and one, representing a proportion), Li et al. [2] run PEER without further data transformation using the number of PEER factors chosen by GTEx [3] (Table 3.3). To assess the impact of data transformation on the PEER factors, we also run PEER after transforming the data in three ways: (1) center and scale (to unit variance) each feature, (2) apply inverse normal transform (INT) [18] to each feature ("INT within feature"), and (3) apply INT to each sample ("INT within sample"). Among these methods, GTEx [3, 4] uses "INT within feature" for its eQTL data and "INT within sample" for its sQTL data (Table 3.3). To quantify how many "distinct" or "nonrepetitive" PEER factors there are, given a set of PEER factors, we group them into clusters such that in each cluster, the correlation between any two PEER factors is above a pre-defined threshold (0.99, 0.9, or 0.8) in absolute value (this is done via hierarchical clustering [54] with complete linkage and the distance defined as one minus the absolute value of the correlation). Therefore, the number of PEER factor clusters can be interpreted as the number of distinct or nonrepetitive PEER factors.

Our results show that in many cases, the number of distinct PEER factors is considerably smaller than the number of PEER factors requested (Figure 2.4), and when this issue is severe (e.g., "No transformation" and "INT within sample"), the PEER factors fail to capture important variance components of the molecular phenotype data (Figure 3.5). Since the numbers of discoveries increase substantially with the numbers of PEER factors in GTEx's eQTL analyses [3, 4], where the PEER factors are essentially identical to PCs (Section 2.1.3), it is possible that replacing the nearly-all-identical PEER factors with appropriate numbers of PCs in Li et al. [2]'s $3'$aQTL analysis can lead to more discoveries. This is a potential direction for a future study.

2.1.3 PEER factors are almost identical to PCs but take three orders of magnitude longer to compute in GTEx eQTL and sQTL data

We report the surprising finding that in both GTEx eQTL and sQTL data [4], the PEER factors obtained by GTEx and used in its QTL analyses are almost identical to PCs. Specifically, given a fully processed molecular phenotype matrix, there is almost always a near-perfect one-to-one correspondence between the PEER factors and the top PCs (Figure 2.5).

This means that after the variational Bayesian inference in PEER initializes with PCs [5], it does not update the PCs much beyond scaling them (see Section 2.1.4 for an explanation). Therefore, it is no surprise that replacing the PEER factors with PCs in GTEx's FastQTL pipeline [4, 17] does not change the QTL results much (Figures 3.6 and 3.7) because in linear regressions (the basis of both Matrix eQTL [29] and FastQTL [17]), scaling and/or shifting the predictors does not change the p-values of t-tests for non-intercept terms (neither does scaling and/or shifting the response, for that matter).

However, PEER is at least three orders of magnitude slower than PCA (Figure 3.6). For a given expression matrix, running PEER without the known covariates (GTEx's approach) takes up to about 32 hours, while running PCA (with centering and scaling; our approach) takes no more than a minute.

To draw a connection between our simulation results and real data results, we analyze them jointly in Figure 3.8 and make the following two key observations. First, we find that in the simulation studies, PCA almost always outperforms PEER in terms of AUPRC (confirming our results in Section 2.1.1), and the percentage of QTL discoveries shared between PEER and PCA is a good predictor of the relative performance of PEER versus PCA—the higher the percentage of QTL discoveries shared, the smaller the performance gap between PEER and PCA. Second, the percentages of QTL discoveries shared between the two methods in GTEx eQTL data [4] fall comfortably within the range of percentage of QTL discoveries shared in Simulation Design 2. These two observations together suggest that PCA likely outperforms PEER in GTEx eQTL data [4] even though the results largely overlap.

2.1.4 PCA, SVA, PEER, and HCP are closely related statistical methods

We report that PCA, SVA, PEER, and HCP are closely related statistical methods despite their apparent dissimilarities. In particular, the methodology behind SVA, PEER, and HCP can all be traced back to PCA (Figure 2.6). We have previously reviewed these methods in detail in Zhou [15]. Here we aim to provide a brief summary and highlight their connections.

PCA [6–10] is traditionally derived by optimizing some objective functions (either maximum variance or minimum reconstruction error; Section 3.5.1), but more recently, it is shown that PCA can be derived as a limiting case of probabilistic principal component analysis (PPCA) [11], which in turn is a special case of factor analysis [7, 12]—a dimension reduction method commonly used in psychology and the social sciences that is based on a *frequentist* probabilistic model.

PEER [1, 5] is based on a *Bayesian* probabilistic model and can be considered a Bayesian version of factor analysis (with the not-very-useful ability to explicitly model the known covariates; see Section 2.1.1 for why we do not find this ability useful). Inference is performed using variational Bayes and initialized with the PCA solution [5]. Given that PCA underlies the PEER model (Figure 2.6) and PEER initializes with PCs, it is not surprising that PEER factors are almost identical to PCs in GTEx eQTL and sQTL data [4] (Section 2.1.3).

SVA [13, 14] is purely algorithmic and is not defined based on a probabilistic model or objective function. The steps of the SVA algorithm are complicated [15], but in a nutshell, SVA iterates between two steps: (1) reweight the features of the molecular phenotype matrix, and (2) perform PCA on the resulting matrix (with centering but without scaling) [14].

Lastly, HCP [16] is defined by minimizing a loss function that is very similar to the minimum-reconstruction-error loss function of PCA (Section 3.5.2). The optimization is done through coordinate descent with one deterministic initialization (see source code of the HCP R package [16]). In short, SVA, PEER, and HCP can all be considered extensions or more complex versions of PCA, though we show that the complexity is a burden rather than a benefit (Figure 2.1).

2.1.5 PCA provides insight into the choice of K

Choosing K, the number of inferred covariates in the context of hidden variable inference or the number of dimensions or clusters in more general contexts, is always a difficult task. Nonetheless, based on the proportion of variance explained (PVE) by each PC (Section 3.5.1), PCA offers convenient ways of choosing K such as the elbow method and the Buja and Eyuboglu (BE) algorithm [55] (more details below). Since SVA is heavily based on PCA (Section 2.1.4), it is able to adapt and make use of the BE algorithm. In contrast, PEER and HCP do not offer easy ways of choosing K; for lack of a better method, users of PEER and HCP often choose K by maximizing the number of discoveries [3, 4, 32, 37–39]. Not only is this approach of choosing K extremely computationally expensive and theoretically questionable, here we also show from the perspective of PCA that it may yield inappropriate choices of K.

Recall from Section 2.1.3 that PEER factors are almost identical to PCs in GTEx eQTL data [4] (the number of PEER factors is chosen by maximizing the number of discovered cis-eGenes for each pre-defined sample size bin; Table 3.3). Therefore, for each tissue type, we compare the number of PEER factors selected by GTEx to (1) the number of PCs chosen via an automatic elbow detection method (Algorithm 2) and (2) the number of PCs chosen via the BE algorithm (Algorithm 3; the default parameters are used). The BE algorithm is a permutation-based approach for choosing K in PCA. Intuitively, it retains PCs that explain more variance in the data than by random chance and discards those that do not. Hence, based on the statistical interpretation of the BE algorithm and the scree plots (examples shown in Figure 2.7), we believe that the number of PCs chosen via BE should be considered an upper bound of the reasonable number of PCs to choose in GTEx eQTL data [4].

Our results show that the number of PEER factors selected by GTEx is almost always greater than the number of PCs chosen via *BE*, which in turn is almost always greater than the number of PCs chosen via *elbow* (Figure 2.7). In particular, the number of PEER factors selected by GTEx far exceeds the number of PCs chosen via *BE* for many tissue types with sample size above 350, suggesting that the number of PEER factors selected by GTEx may be too large. This hypothesis is further supported by the fact that we can reduce the number of inferred covariates to between 20% and 40% of the number of PEER factors selected by GTEx without significantly reducing the number of discovered cis-eGenes (Figure 2.7).

2.2 Methods

2.2.1 Evaluation metrics

Given a simulated data set, we evaluate each of the 15 methods in Table 2.1 mainly in three ways (when applicable): runtime, AUPRC, and adjusted R^2 measures (including adjusted R^2, reverse adjusted R^2, and concordance score).

First, we record the runtime of the hidden variable inference step (Section 3.4; not applicable for Ideal and Unadjusted).

Second, we calculate the area under the precision-recall curve (AUPRC) of the QTL result. We use AUPRC rather than the area under the receiver operating characteristic curve (AUROC) because AUPRC is more appropriate for data sets with imbalanced classes (there are far more negatives than positives in our simulated data sets and in QTL settings in general). Since AUPRC measures the trade-off between the true positive rate (i.e., power) and the false discovery rate (i.e., one minus precision), it is a more comprehensive metric than power. However, to contrast the results in Stegle et al. [5], we also compare the powers of the different methods in Simulation Design 1.

Third, for each simulated data set, each method except Ideal and Unadjusted gets an adjusted R^2 score (short as "adjusted R^2"), a reverse adjusted R^2 score (short as "reverse adjusted R^2"), and a concordance score. The adjusted R^2 score summarizes how well the true hidden covariates can be captured by the inferred covariates; the reverse adjusted R^2

score summarizes how well the inferred covariates can be captured by the true hidden covariates (a low score indicates that the inferred covariates are invalid or "meaningless"); lastly, the concordance score is the average of the previous two scores and thus measures the concordance between the true hidden covariates and the inferred covariates. Specifically, given m true hidden covariates and n inferred covariates, first, we calculate m adjusted R^2's (regressing each true hidden covariate against the inferred covariates) and n reverse adjusted R^2's (regressing each inferred covariate against the true hidden covariates); then, we average the m adjusted R^2's to obtain the adjusted R^2 score and average the n reverse adjusted R^2's to obtain the reverse adjusted R^2 score; finally, we define the concordance score as the average of the adjusted R^2 score and the reverse adjusted R^2 score.

2.2.2 Selection of representative methods for detailed comparison

Here we describe how we select a few representative methods from the 15 methods for detailed comparison in Simulation Design 2 (Table 2.1). From Figures 2.2(d) and 3.3, we see that the two PCA methods perform almost identically, so for simplicity, we select PCA_direct_screeK. The two SVA methods perform almost identically as well, so we select SVA_BE. For PEER, whether the known covariates are inputted when PEER is run has little effect on the AUPRC. Further, we observe that when we use the true K, the factor approach outperforms the residual approach, but when we use a large K, the residual approach outperforms the factor approach. Therefore, we select PEER_withCov_trueK_factors and PEER_withCov_largeK_residuals as the representative PEER methods. In addition, Ideal, Unadjusted, and HCP_trueK are selected.

2.2.3 A numerical example

Here we provide a simple numerical example of QTL analysis with hidden variable inference by summarizing the setup of GTEx's cis-eQTL analysis for Colon - Transverse [4].

Let Y denote the $n \times p$ fully processed gene expression matrix with $n = 368$ samples and $p = 25{,}379$ genes. Let X_1 denote the $n \times K_1$ known covariate matrix with $K_1 = 8$ known covariates, which include the top five genotype PCs, WGS sequencing platform (HiSeq 2000 or HiSeq X), WGS library construction protocol (PCR-based or PCR-free), and donor sex. Let X_{inferred} denote the $n \times K$ inferred covariate matrix with $K = 60$ PEER factors, which are obtained by running PEER on Y (Table 3.3). For gene j, $j = 1, \cdots, p$, the relevant genotype data is stored (conceptually speaking) in S_j, the $n \times q_j$ genotype matrix, where each column of S_j corresponds to a local common SNP for gene j, and q_j is typically under 15,000.

Given these input data, the nominal pass (the first step) of FastQTL [17], or equivalently, Matrix eQTL [29], performs a linear regression for each gene and each of its local common SNPs. Specifically, for $j = 1, \cdots, p$, $l = 1 \cdots, q_j$, the linear regression represented by the following R `lm()` formula is run:

$$Y[\,,\, j] \;\sim\; S_j[\,,\, l] \,+\, X_1 \,+\, X_{\text{inferred}}$$

(where $Y[\,,\, j]$ denotes the jth column of Y, and $S_j[\,,\, l]$ denotes the lth column of S_j), and the p-value for the null hypothesis that the coefficient corresponding to $S_j[\,,\, l]$ is zero (given the covariates) is retained. The top five genotype PCs in X_1 are included in the analysis to correct for population stratification [3, 4] and are typically considered known covariates (see Section 6.1).

2.3 Tables and figures

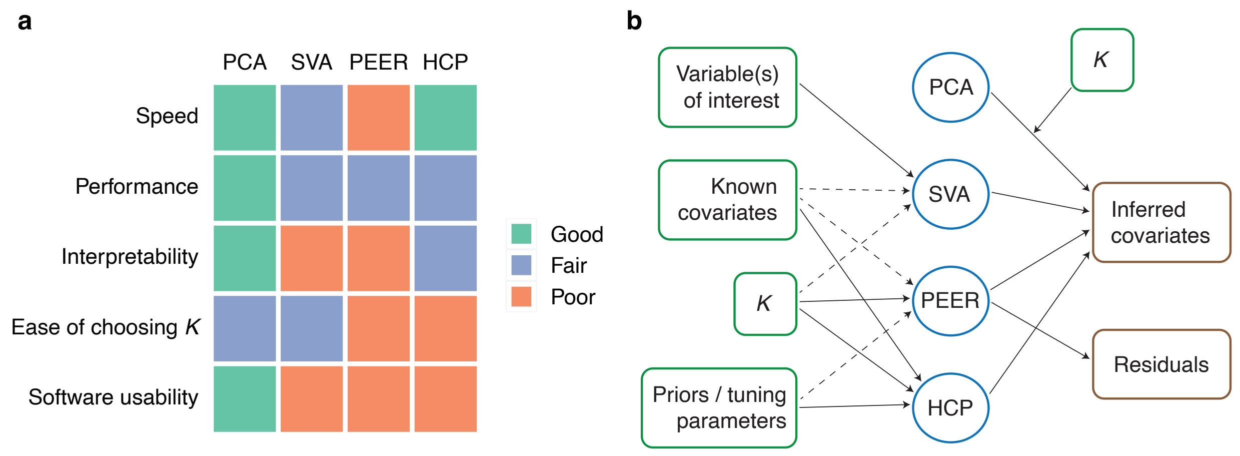

Figure 2.1: Overall comparison of PCA, SVA, PEER, and HCP and summary of their inputs and outputs. In this work, we use K to denote the number of inferred covariates, which are called PCs, SVs, PEER factors, and HCPs in PCA, SVA, PEER, and HCP, respectively. **a** PCA is faster, better-performing, and much easier to interpret and use. For speed and performance comparison, see Section 2.1.1 (and to a lesser extent, Sections 2.1.2 and 2.1.3). For interpretability and ease of choosing K, see Sections 2.1.4 and 2.1.5, respectively. In terms of software usability, SVA is difficult to apply in QTL settings (Section 3.4), PEER is difficult to install, and HCP is poorly documented. In addition, PEER suffers from the disadvantage that there is no consensus in the literature on how it should be used (Section 3.4). **b** Inputs (green boxes) and outputs (brown boxes) of the four methods. The fully processed molecular phenotype matrix (after the effects of the known covariates are regressed out in the case of PCA_resid; Table 2.1) is a required input for all four methods and is thus omitted in the diagram. Dashed arrows indicate optional inputs. PEER outputs both inferred covariates and residuals of the inputted molecular phenotype matrix [1].

	Inference method	**Method**	**Response, covariates**	**Method abbr. (if selected)**
	(A)	(B)	(C)	(D)
1		**Ideal**	Y, $X_1 + X_2$	Ideal
2		**Unadjusted**	Y, X_1	Unadjusted
3	PCA_direct	**PCA**_direct_screeK	Y, X_1 (filtered) + top PCs	PCA
4	PCA_resid	**PCA**_resid_screeK	Y, X_1 + top PCs	
5	SVA_trueK	**SVA**_trueK	Y, X_1 + SVs	
6	SVA_BE	**SVA**_BE	Y, X_1 + SVs	SVA
7	PEER_noCov_trueK	**PEER**_noCov_trueK_factors	Y, X_1 (filtered) + PEER factors	
8		**PEER**_noCov_trueK_residuals	Y_{resid}, `NULL`	
9	PEER_noCov_largeK	**PEER**_noCov_largeK_factors	Y, X_1 (filtered) + PEER factors	
10		**PEER**_noCov_largeK_residuals	Y_{resid}, `NULL`	
11	PEER_withCov_trueK	**PEER**_withCov_trueK_factors	Y, X_1 + PEER factors	PEER, true K, factors
12		**PEER**_withCov_trueK_residuals	Y_{resid}, `NULL`	
13	PEER_withCov_largeK	**PEER**_withCov_largeK_factors	Y, X_1 + PEER factors	
14		**PEER**_withCov_largeK_residuals	Y_{resid}, `NULL`	PEER, large K, residuals
15	HCP_trueK	**HCP**_trueK	Y, X_1 + HCPs	HCP

Table 2.1: Summary of the 15 methods we compare based on simulation studies, including Ideal, Unadjusted, and 13 variants of PCA, SVA, PEER, and HCP (Section 3.4). Out of the 15 methods, we select a few representative methods (Section 2.2.2) for detailed comparison in Simulation Design 2, the abbreviations of which are shown in (D). Y denotes the gene expression matrix, Y_{resid} denotes the residual matrix outputted by PEER, X_1 denotes the known covariate matrix, and X_2 denotes the hidden covariate matrix. In Line 3, PCA is run on Y directly; in Line 4, PCA is run after the effects of X_1 are regressed out from Y (Section 3.4). The addition signs in (C) denote column concatenation. "filtered" means that we filter out the known covariates that are captured well by the inferred covariates (unadjusted $R^2 \geq 0.9$); this filtering is only needed when the hidden variable inference method in (A) does not explicitly take the known covariates into account.

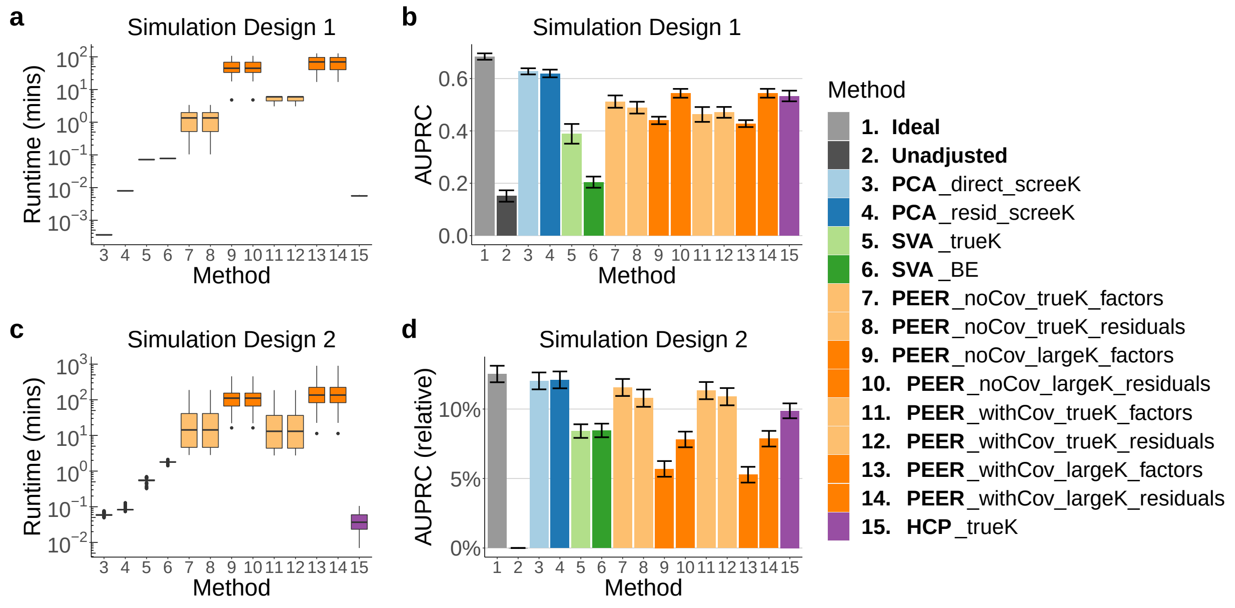

Figure 2.2: Runtime and AUPRC comparison of all 15 methods (Table 2.1) in Simulation Design 1 and Simulation Design 2. **a, c** PCA and HCP each takes within a few seconds, SVA takes up to a few minutes, and PEER takes up to about 1,000 minutes, equivalent to about 17 hours. In particular, PEER takes longer to run when K is larger (dark orange vs. light orange boxes). **b, d** PCA outperforms SVA, PEER, and HCP in terms of AUPRC. The height of each bar represents the average across simulated data sets. For ease of visualization, in **d**, the y-axis displays $(\text{AUPRC} - \text{AUPRC}_{\text{Unadjusted}}) / \text{AUPRC}_{\text{Unadjusted}}$. In this work, error bars indicate standard errors unless otherwise specified (whiskers in box plots are not considered error bars).

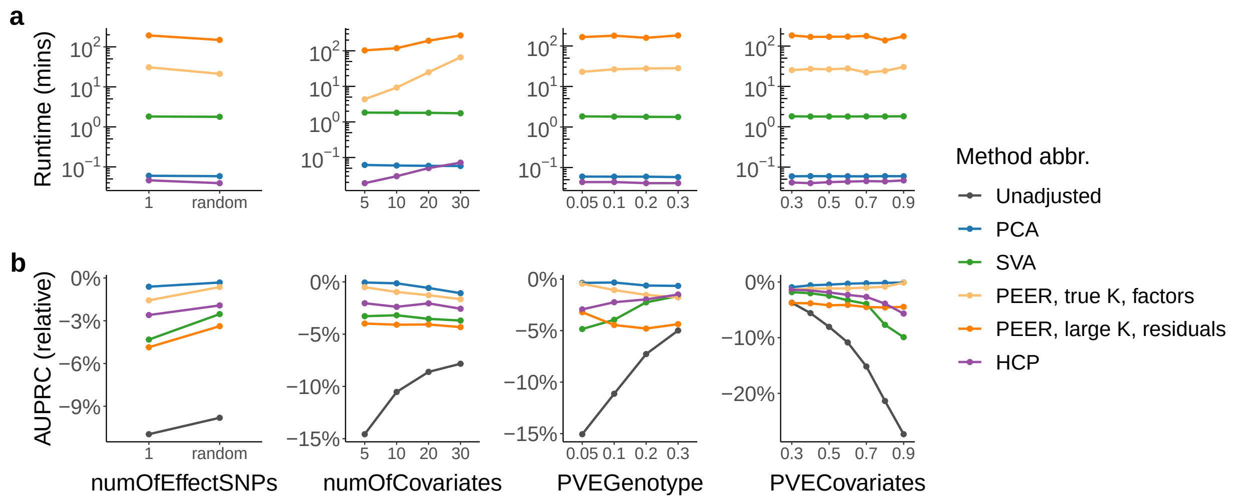

Figure 2.3: Detailed runtime and AUPRC comparison of the selected representative methods (Table 2.1) in Simulation Design 2. Each point represents the average across simulated data sets. The x-axes are: number of effect SNPs per gene (`numOfEffectSNPs`), number of simulated covariates (`numOfCovariates`; including known and hidden covariates), proportion of variance explained by genotype (`PVEGenotype`), and proportion of variance explained by covariates (`PVECovariates`) (Section 3.3). **a** PCA and HCP are orders of magnitude faster than SVA, which in turn is orders of magnitude faster than PEER. **b** PCA outperforms SVA, PEER, and HCP in terms of AUPRC across different simulation settings. For ease of visualization, the y-axis displays $(\text{AUPRC} - \text{AUPRC}_{\text{Ideal}})/\text{AUPRC}_{\text{Ideal}}$. Consistent with our expectation, the performance gap between Unadjusted and Ideal is the largest (and thus accounting for hidden covariates is the most important) when `numOfCovariates` is small, when `PVEGenotype` is small, and when `PVECovariates` is large.

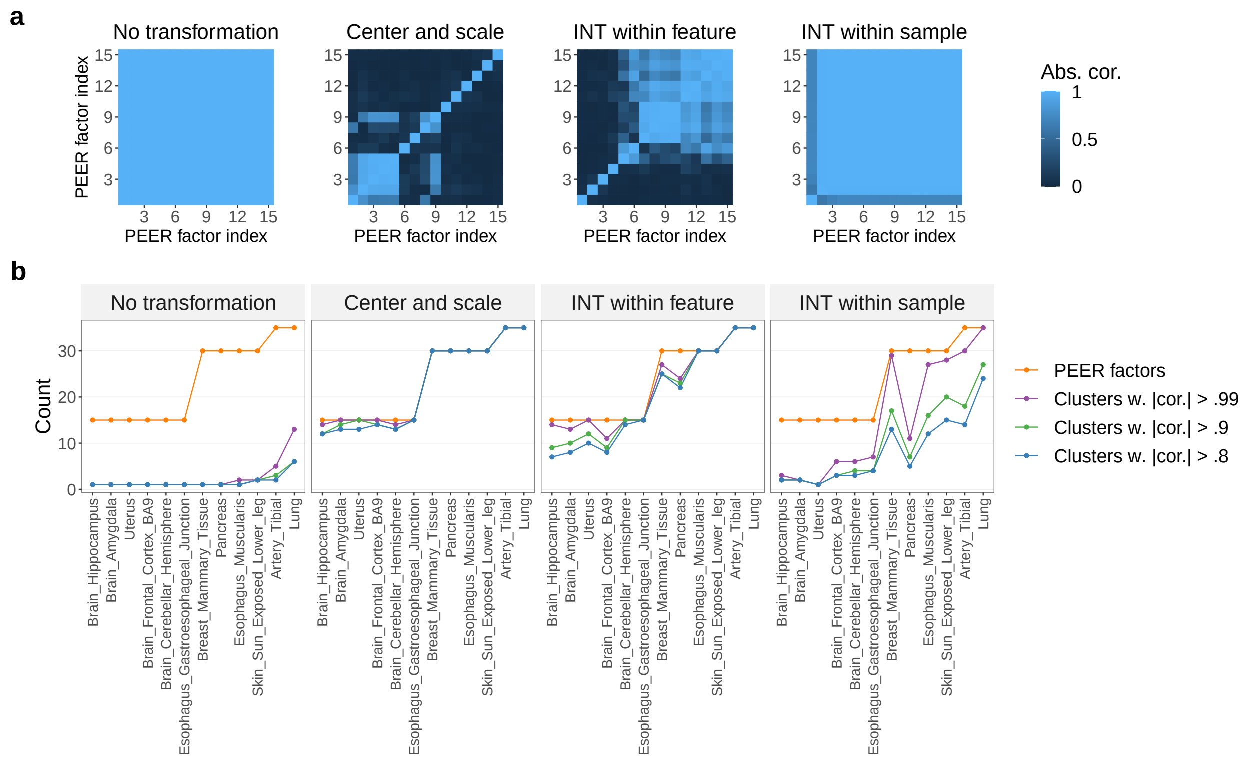

Figure 2.4: In the 3′aQTL data prepared by Li et al. [2] from GTEx RNA-seq reads [3], PEER factors can be highly correlated with each other to the extent that many or all of them are practically identical. **a** Correlation heatmaps of PEER factors for Brain_Hippocampus. For ease of visualization, the PEER factors are reordered based on results from hierarchical clustering (Section 2.1.2) in each heatmap. **b** The x-axis shows 12 randomly selected tissue types with increasing sample sizes. The y-axis shows the number of PEER factors requested (orange line) or the number of PEER factor clusters. Given a set of PEER factors, we group them into clusters such that in each cluster, the correlation between any two PEER factors is above 0.99, 0.9, or 0.8 in absolute value (Section 2.1.2). Therefore, the number of PEER factor clusters can be interpreted as the number of distinct or nonrepetitive PEER factors. We find that in many cases, the number of distinct PEER factors is considerably smaller than the number of PEER factors requested, and when this issue is severe (e.g., "No transformation" and "INT within sample"), the PEER factors fail to capture important variance components of the molecular phenotype data (Figure 3.5).

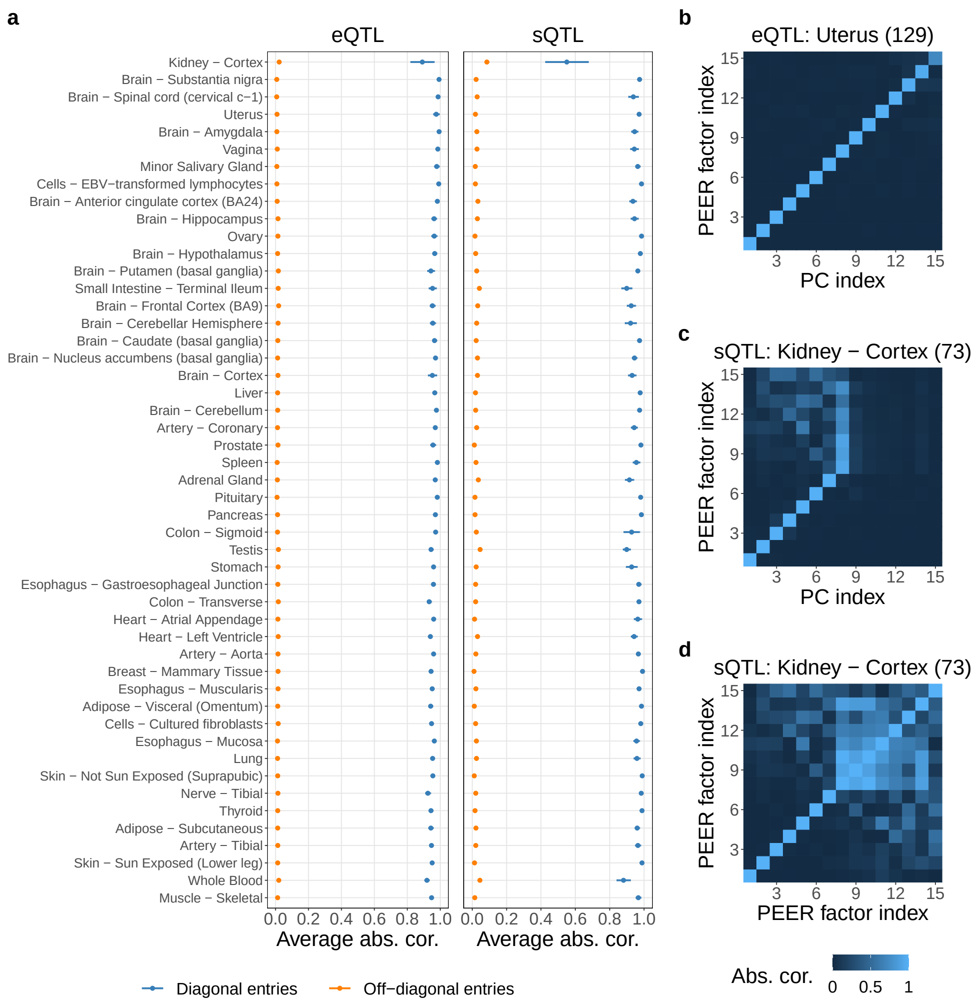

Figure 2.5: PEER factors are almost identical to PCs in GTEx eQTL and sQTL data [4]. **a** The y-axis shows all 49 tissue types with GTEx QTL analyses ordered by sample size (from small to large). Given a fully processed molecular phenotype matrix, we summarize the correlation matrix (in absolute value) between the PEER factors obtained and used by GTEx and the top PCs into two numbers: the average of the diagonal entries and the average of the off-diagonal entries. With the exception of Kidney - Cortex sQTL data, the diagonal entries have averages close to one, and the off-diagonal entries have averages close to zero (both have minimal standard errors). **b** A typical correlation heatmap showing near-perfect one-to-one correspondence between the PEER factors and the top PCs. **c** In Kidney - Cortex sQTL data, the PEER factors and the top PCs do not have a perfect one-to-one correspondence. The reason is because the PEER factors are highly correlated with each other (**d**), while PCs are always uncorrelated (Section 3.5.1). The numbers in parentheses represent sample sizes. To produce this figure, we reorder the PEER factors based on the PCs (Algorithm 1), although in almost all cases, this reordering does not change the original ordering of the PEER factors because PEER initializes with PCs [5].

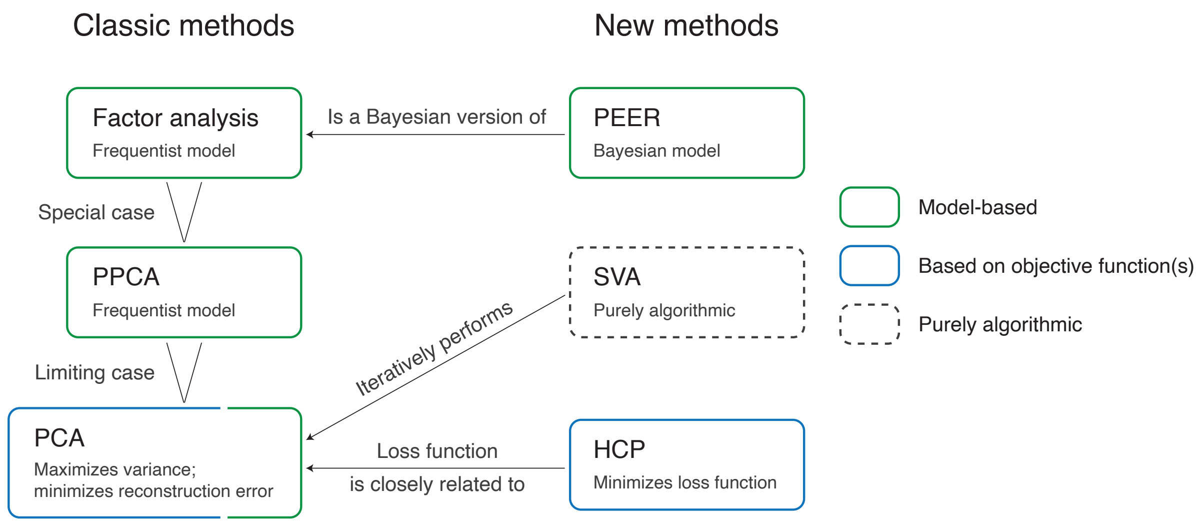

Figure 2.6: PCA, SVA, PEER, and HCP are closely related statistical methods despite their apparent dissimilarities. In particular, the methodology behind SVA, PEER, and HCP can all be traced back to PCA. PCA [6–10] is traditionally derived by optimizing some objective functions (either maximum variance or minimum reconstruction error; Section 3.5.1), but more recently, it is shown that PCA can be derived as a limiting case of probabilistic principal component analysis (PPCA) [11], which in turn is a special case of factor analysis [7, 12]. PEER [1, 5] is based on a Bayesian probabilistic model and can be considered a Bayesian version of factor analysis. SVA [13, 14] is purely algorithmic and is not defined based on a probabilistic model or objective function. The steps of the SVA algorithm are complicated [15], but in a nutshell, SVA iterates between two steps: (1) reweight the features of the molecular phenotype matrix, and (2) perform PCA on the resulting matrix (with centering but without scaling) [14]. Lastly, HCP [16] is defined by minimizing a loss function that is very similar to the minimum-reconstruction-error loss function of PCA (Section 3.5.2).

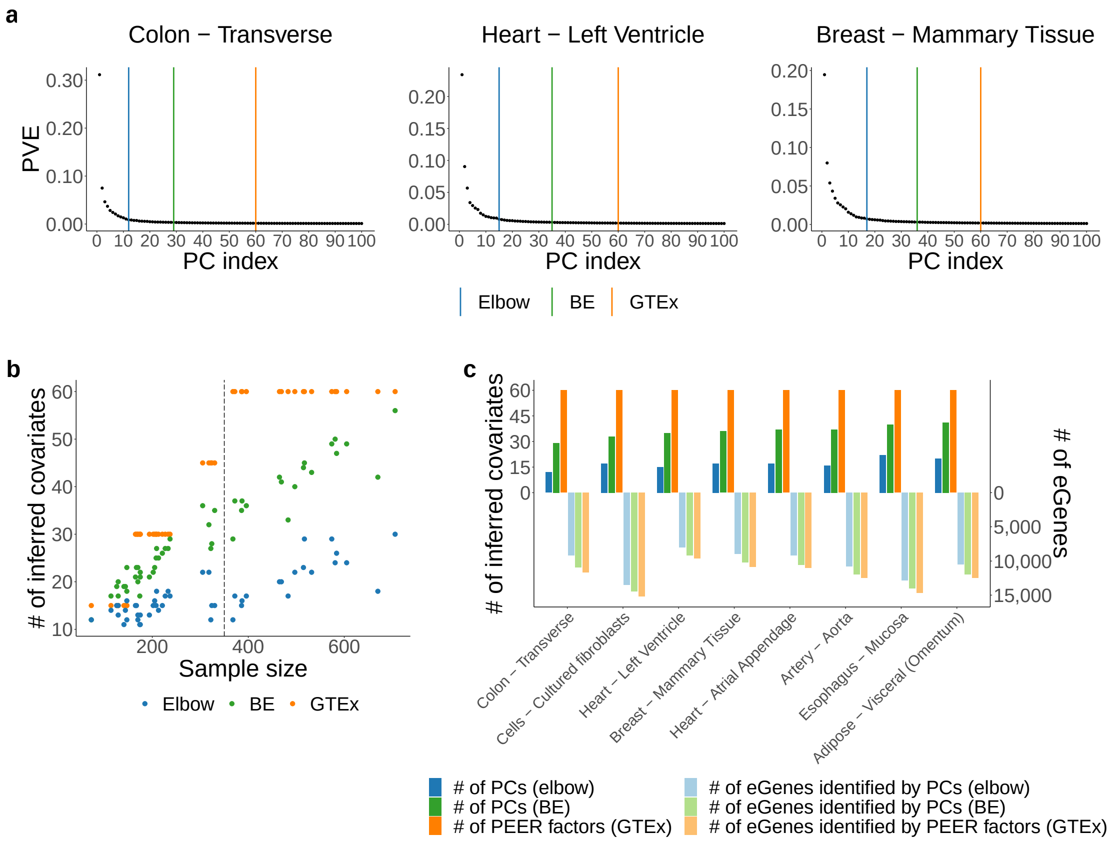

Figure 2.7: PCA provides insight into the choice of K. Recall from Section 2.1.3 that PEER factors are almost identical to PCs in GTEx eQTL data [4]. Therefore, for each tissue type, we compare the number of PEER factors selected by GTEx to (1) the number of PCs chosen via an automatic elbow detection method (Algorithm 2) and (2) the number of PCs chosen via the BE algorithm (Algorithm 3; the default parameters are used). **a** Example scree plots. **b** This scatter plot contains 49 dots of each color, corresponding to the 49 tissue types with GTEx eQTL analyses. The number of PEER factors selected by GTEx far exceeds the number of PCs chosen via BE for many tissue types with sample size above 350 (dashed line), suggesting that the number of PEER factors selected by GTEx may be too large. **c** For the eight tissue types with the largest absolute differences between the number of PEER factors chosen by GTEx and the number of PCs chosen via BE (all eight tissue types have sample size above 350), we replace the PEER factors with smaller numbers of PCs in GTEx's FastQTL pipeline [4, 17] and find that we can reduce the number of inferred covariates to between 20% ($12/60 = 20\%$, Colon - Transverse) and 40% ($22/60 \approx 36.67\%$, Esophagus - Mucosa) of the number of PEER factors selected by GTEx without significantly reducing the number of discovered cis-eGenes.

2.4 Availability of data and materials

The R package PCAForQTL and a tutorial on using PCA for hidden variable inference in QTL mapping are available at `https://github.com/heatherjzhou/PCAForQTL` [56]. The code used to generate the results in this work is available at `https://doi.org/10.5281/zenodo.6788888` [57]. In addition, this work makes use of the following data and software:

- GTEx V8 public data [4], including fully processed gene expression matrices, fully processed alternative splicing phenotype matrices, known covariates, PEER factors, and QTL results, are downloaded from `https://gtexportal.org/home/datasets`.
- GTEx V8 protected data [4], specifically, the whole genome sequencing (WGS) phased genotype data, are downloaded from the AnVIL repository with an approved dbGaP application (see `https://gtexportal.org/home/protectedDataAccess`).
- $3'$aQTL data prepared by Li et al. [2] from GTEx RNA-seq reads [3] are available from the authors by request.
- SVA R package Version 3.40.0 (`https://bioconductor.org/packages/sva/`, accessed on October 15, 2021).
- PEER R package Version 1.3 (`https://bioconda.github.io/recipes/r-peer/README.html`, accessed before October 15, 2021).
- HCP R package Version 1.6 (`https://rdrr.io/github/mvaniterson/Rhcpp/`, accessed on October 15, 2021).
- FastQTL (`https://github.com/francois-a/fastqtl`, accessed before October 15, 2021).

CHAPTER 3

Supplementary materials for PCA outperforms popular hidden variable inference methods for molecular QTL mapping

3.1 Limitations of the original PEER simulation study

The simulation study in the original PEER publication [5] is limited. We categorize its limitations into three categories: (1) data analysis limitations, (2) overall design limitations, and (3) data simulation limitations.

The data analysis limitations include:

(a) The study only compares PEER against the other methods in terms of power, not in terms of false positive rate or false discovery rate (see Section 2.2.1 for our evaluation metrics).

(b) The study does not use PCA or SVA properly (we do; Section 3.4).

(c) The study does not evaluate the different ways of using PEER (we do; Section 3.4).

(d) The study uses ad hoc priors for PEER that are different from the default priors (we use the default priors; Section 3.4).

The overall design limitations include:

(a) The study only simulates one replicate of one experiment. That is, the entire simulation study is based on one simulated data set (we simulate 10 replicates in our first simulation study; Section 3.2).

The data simulation limitations include (see Table 3.1 for our solutions):

(a) The data dimensions are minimal, with $q = 100$ SNPs in the entire genome.

(b) The SNP genotypes are simulated independently and identically with a target minor allele frequency (MAF) of 0.4, so there is no linkage disequilibrium (LD) and a higher average MAF than in real data (the average MAF in GTEx data [4], after SNPs with MAF under 0.01 are filtered out, is about 0.15; Section 3.3.1).

(c) The gene expression levels are primarily driven by trans-regulatory effects rather than cis-regulatory effects or covariate effects (Table 3.2), inconsistent with the common belief that trans-regulatory effects are generally weaker than cis-regulatory effects.

In addition, the simulation study in the original PEER publication [5] is imperfect in that the description of the data simulation and analysis is vague and inconsistent, and there is no reproducible code. In contrast, we describe our data simulation and analysis in detail (Sections 3.2 to 3.4) and provide the code we use to generate the results (see Availability of data and materials).

3.2 Simulation Design 1

3.2.1 Data simulation

In Simulation Design 1, we perform 10 replicates of the same experiment, where in each replicate, we follow the data simulation in Stegle et al. [5] as closely as possible.

In each replicate, we simulate a data set with $n = 80$ individuals, $p = 400$ genes, $q = 100$ SNPs in the entire genome, $K_1 = 3$ known covariates, and $K_2 = 7$ hidden covariates. Let i, j, l, and k be the indices of individuals, genes, SNPs, and covariates, respectively. That is, $i = 1, \cdots, n$; $j = 1, \cdots, p$; $l = 1, \cdots, q$; and $k = 1, \cdots, (K_1 + K_2)$. The data simulation consists of three steps.

In the first step, we simulate $Y_{\text{beforeDSE}}$, the gene expression matrix before downstream effect, based on

$$\underset{n\times p}{Y_{\text{beforeDSE}}} = \underset{n\times q}{S} \left(\underset{q\times p}{I_1} \odot \underset{q\times p}{B_1} \right) + \underset{n\times(K_1+K_2)}{X} \quad \underset{(K_1+K_2)\times p}{B_2} + \underset{n\times p}{E}, \tag{3.2.1}$$

where $\odot$ denotes element-wise multiplication. Specifically, in the genotype component, we have

- S: genotype matrix. Each entry is drawn independently from $Binom\,(2, \text{prob} = 0.4)$. That is, the target MAF is 0.4. In this work, all random sampling is independent unless otherwise specified.
- I_1: effect indicator matrix. Each entry is drawn from $Ber\,(0.01)$.
- B_1: effect size matrix. Each entry is drawn from $N(0, \text{var} = 4)$.

In the covariate component, we have

- X: covariate matrix. Each entry is drawn from $N(0, \text{var} = 0.6)$. The first K_1 columns are designated as the known covariates (X_1, $n \times K_1$), and the last K_2 columns are designated as the hidden covariates (X_2, $n \times K_2$).
- B_2: effect size matrix. First, we draw $\sigma_k^2 \sim 0.8\,(\Gamma\,(\text{shape} = 2.5, \text{rate} = 0.6))^2$, the covariate-specific effect size variance. Then, we draw $(B_2)_{kj} \sim N(0, \text{var} = \sigma_k^2)$.

Lastly, in the noise component, we have

- E: noise matrix. First, we draw $\tau_j \sim \Gamma\,(\text{shape} = 3, \text{rate} = 1)$, the gene-specific noise precision. Then, we draw $(E)_{ij} \sim N(0, \text{var} = 1/\tau_j)$.

In the second step, we simulate Y_{DSE}, the gene expression matrix due to the downstream effect of genes, based on

$$\underset{n\times p}{Y_{\text{DSE}}} = \underset{n\times p}{Y_{\text{beforeDSE}}} \left(\underset{p\times p}{I_3} \odot \underset{p\times p}{B_3} \right), \tag{3.2.2}$$

where we have

- I_3: effect indicator matrix. To simulate I_3, we start with a zero matrix. Then, we randomly choose three rows corresponding to genes with at least one cis-QTL (Section 3.2.2). For each of these three rows, we randomly assign 30 entries corresponding to genes other than the current gene in consideration (avoiding self-loops) to be one.
- B_3: effect size matrix. Each entry is drawn from $N(8, \text{var} = 0.8)$ for "strong downstream effects" [5].

As we see in Section 3.2.2, the downstream effect of genes induces trans-QTL relations.

In the third and last step, we define Y, the final, observed gene expression matrix, as

$$\underset{n\times p}{Y} = \underset{n\times p}{Y_{\text{beforeDSE}}} + \underset{n\times p}{Y_{\text{DSE}}} . \tag{3.2.3}$$

3.2.2 Definition of truth

In a simulated data set, the cis-QTL relations are encoded in the $q \times p$ binary matrix I_1. The lj-th entry being one means that SNP l and gene j form a cis-QTL pair (i.e., SNP l is a cis-QTL for gene j).

The trans-QTL relations are encoded in J, also a $q \times p$ binary matrix. J is defined based on I_1 and I_3. Specifically, SNP l and gene j form a trans-QTL pair if and only if SNP l is a cis-QTL for gene j' *and* gene j' has downstream effect on gene j, $j' \neq j$.

The overall truth is encoded in $\mathbb{1}\big((I_1 + J) \geq 1\big)$, again a $q \times p$ binary matrix. We use this matrix as the truth when calculating AUPRCs. The lj-th entry being one means that SNP l and gene j form a cis-QTL or trans-QTL pair (or both).

	Simulation Design 1	**Simulation Design 2**
Data simulation	Follows Stegle et al. [5]	Loosely based on Wang et al. [20]
# of experiments	1	176
# of replicates per experiment	10	2
# of simulated data sets	10	352
Genotype data	Simulated (no LD, high MAF)	Real genotype data from GTEx [4]
Cis-QTL relations present	✓	✓
Trans-QTL relations present	✓	✗
Source(s) of expression variation	Primarily trans-regulatory effects	Carefully controlled genotype effects and covariate effects
# of individuals	$n = 80$	$n = 838$
# of genes	$p = 400$	$p = 1{,}000$
# of SNPs	$q = 100$ SNPs in the entire genome	$q = 1{,}000$ local common SNPs per gene
# of known covariates	$K_1 = 3$	$K_1 = 2,\ 3,\ 5,$ or 8 depending on the experiment
# of hidden covariates	$K_2 = 7$	$K_2 = 3,\ 7,\ 15,$ or 22 depending on the experiment

Table 3.1: Summary of the main differences between Simulation Design 1 and Simulation Design 2. Highlighted in blue are the major data simulation limitations (Section 3.1) of Simulation Design 1, all of which we address in Simulation Design 2.

Replicate	Var $(Y_{\text{beforeDSE}})$	Var (Y_{DSE})	Var (Y)	Var $(Y_{\text{beforeDSE}})$ / Var (Y)	Var (Y_{DSE}) / Var (Y)
1	124.34	1757.07	1889.57	6.58%	92.99%
2	140.92	1505.17	1677.64	8.40%	89.72%
3	213.56	929.96	1169.18	18.27%	79.54%
4	71.85	761.01	855.39	8.40%	88.97%
5	123.07	2434.45	2574.51	4.78%	94.56%
6	74.94	1029.29	1092.65	6.86%	94.20%
7	148.61	2490.72	2628.93	5.65%	94.74%
8	79.36	796.55	868.05	9.14%	91.76%
9	54.62	1340.10	1390.72	3.93%	96.36%
10	65.64	831.89	895.90	7.33%	92.86%
Average				7.93%	**91.57%**

Table 3.2: In Simulation Design 1, which follows the data simulation in Stegle et al. [5] as closely as possible, the gene expression levels are primarily driven by trans-regulatory effects rather than cis-regulatory effects or covariate effects. Var $(Y_{\text{beforeDSE}})$ is defined as the variance of the $n \times p$ entries of $Y_{\text{beforeDSE}}$, and the other variances in the table are defined similarly. We find that Var (Y_{DSE}) / Var (Y) is above 90% on average.

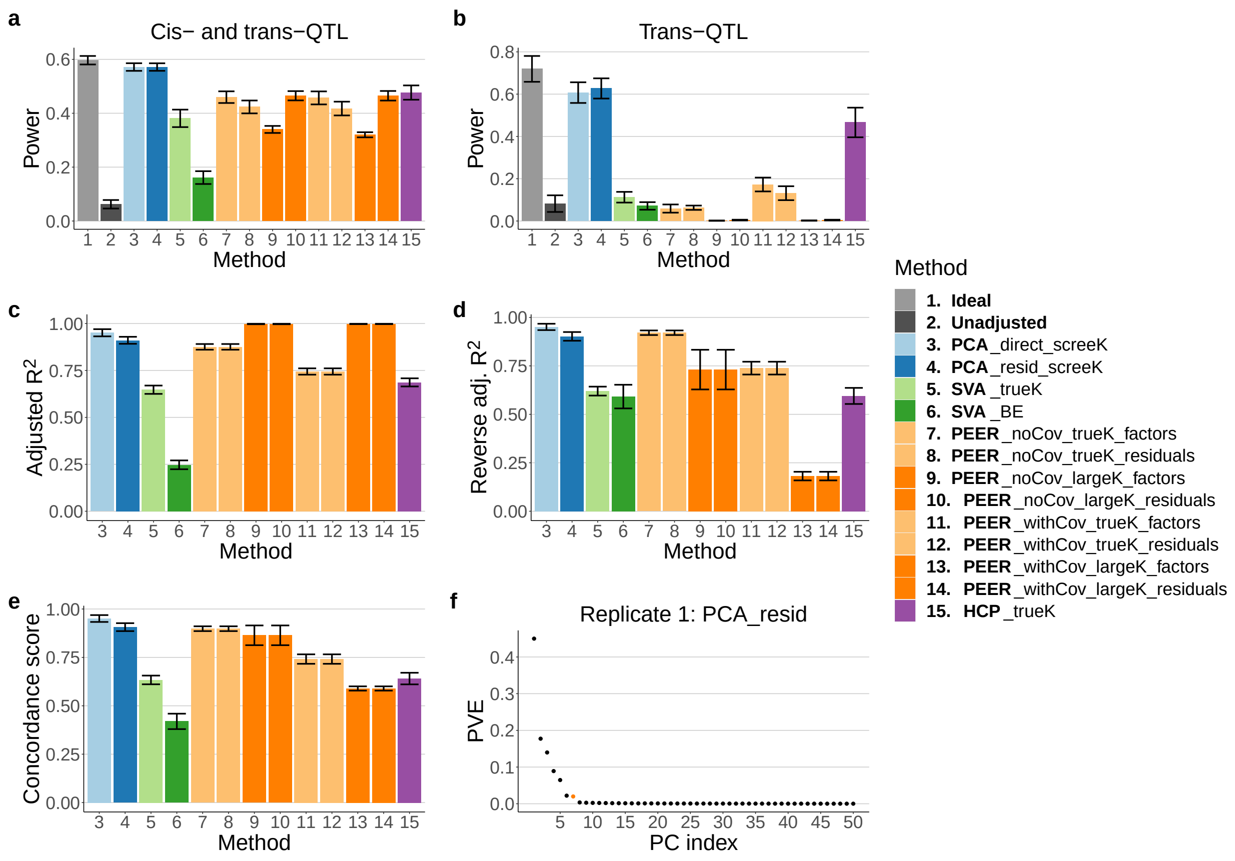

Figure 3.1: Comparison of all 15 methods (Table 2.1) in terms of power and adjusted R^2 measures in Simulation Design 1 (the height of each bar represents the average across simulated data sets) and an example scree plot. **a, b** PCA is more powerful than SVA, PEER, and HCP both when we consider all QTL relations (**a**) and when we focus on trans-QTL relations (**b**). Binary decisions are made based on p-values using the Benjamini-Hochberg (BH) procedure and a target false discovery rate of 0.05. **c, d, e** PCA performs the best in terms of concordance score. PEER with a large K (dark orange bars) performs well in terms of adjusted R^2 but less well in terms of reverse adjusted R^2. **f** An example scree plot that unambiguously suggests the true number of hidden covariates, seven in this case, as the reasonable number of PCs to choose (the y-axis represents the proportion of variance explained).

3.3 Simulation Design 2

3.3.1 Data simulation

In Simulation Design 2, we use real genotype data from GTEx [4], focus on cis-QTL detection, and carefully control the genotype effects and covariate effects in 176 experiments with two replicates per experiment. This simulation design takes inspiration from and is loosely based on Wang et al. [20].

In each experiment-replicate combination, we simulate a data set with $n = 838$ individuals, $p = 1{,}000$ genes, $q = 1{,}000$ local common SNPs per gene, K_1 known covariates, and K_2 hidden covariates (the values of K_1 and K_2 depend on the experiment; see below). Again, let i, j, l, and k be the indices of individuals, genes, SNPs, and covariates, respectively. That is, $i = 1, \cdots, n$; $j = 1, \cdots, p$; $l = 1, \cdots, q$; and $k = 1, \cdots, (K_1 + K_2)$.

We begin by obtaining $SArray$, the $n \times q \times p$ genotype array that remains constant throughout Simulation Design 2. $SArray[\,,\,,j]$, an $n \times q$ matrix, is the genotype matrix for the q local common SNPs for gene j. We obtain $SArray$ with the following steps:

(a) Download the whole genome sequencing (WGS) phased genotype data for $n = 838$ individuals from GTEx V8 [4].

(b) Randomly select $p = 1{,}000$ genes from the more than 20,000 genes on Chromosomes 1 to 22.

(c) For each gene, obtain the genotype data for the $q = 1{,}000$ SNPs with MAF$\geq$ 0.01 that are the closest to the gene's transcription start site (TSS); we find that these SNPs are almost always within 1 Mb of the TSS. The average MAF of $SArray$, calculated as $\overline{SArray}/2$, the average of all entries of $SArray$ divided by 2, is $0.1474385 \approx 0.15$.

Each experiment is characterized by four attributes:

(a) Number of effect SNPs per gene (`numOfEffectSNPs`): 1 or `random`.

(b) Number of covariates (`numOfCovariates`): 5, 10, 20, or 30.

    - Number of known covariates (K_1): 2, 3, 5, 8, respectively.
    - Number of hidden covariates (K_2): 3, 7, 15, 22, respectively.

(c) Proportion of variance explained by genotype (`PVEGenotype`): 0.05, 0.1, 0.2, or 0.3.

(d) Proportion of variance explained by covariates (`PVECovariates`): minimum 0.3, maximum $1-0.05-$`PVEGenotype`, in increments of 0.1. For example, when `PVEGenotype` $=$ 0.05, `PVECovariates` takes seven possible values: 0.3, 0.4, 0.5, $\cdots$, 0.9.

Therefore, we have a total of $2 \times 4 \times (7+6+5+4) = 8 \times 22 = 176$ experiments covering typical scenarios in QTL studies [20]. Following Wang et al. [20], we use the term "effect SNPs" to refer to SNPs that have a nonzero cis effect on a given gene.

Given `numOfEffectSNPs`, `numOfCovariates`, `PVEGenotype`, and `PVECovariates`, we simulate each data set based on

$$\underset{n\times p}{Y} = \underset{n\times q\times p}{SArray} \otimes \left(\underset{q\times p}{I} \odot \underset{q\times p}{B_1}\right) + \underset{n\times(K_1+K_2)}{X} \quad \underset{(K_1+K_2)\times p}{B_2} + \underset{n\times p}{E}\,, \tag{3.3.1}$$

where Y is the gene expression matrix, and $\otimes$ is defined as

$$\underset{n\times p}{C} = \underset{n\times q\times p}{A} \otimes \underset{q\times p}{B} \quad \Leftrightarrow \quad \underset{n\times 1}{C[\,,j]} = \underset{n\times q}{A[\,,\,,j]} \times \underset{q\times 1}{B[\,,j]}\,,\ j = 1,\cdots,p\,. \tag{3.3.2}$$

Specifically, in the genotype component, we have

- $SArray$: genotype array. $SArray[\,,\,,j]$, an $n \times q$ matrix, is the genotype matrix for the q local common SNPs for gene j (see above).
- I: effect indicator matrix.
    - If `numOfEffectSNPs` $= 1$, then for each column, we randomly assign one entry to be one while keeping the other entries zero.
    - If `numOfEffectSNPs` $=$ `random`, then each entry of I is drawn from $Ber\,(1/q)$. This means that for each gene, the number of effect SNPs is drawn from $Binom\,(q,\ \text{prob} = 1/q)$. This binomial distribution approximates the empirical distribution of the number

of independent cis-eQTLs per gene in GTEx data [4] well (Figure 3.2).

- B_1: effect size matrix. Each entry is drawn from $N(0, 1)$.

In the covariate component, we have

- X: covariate matrix. Each entry is drawn from $N(0, 1)$. As in Simulation Design 1, the first K_1 columns are designated as the known covariates (X_1 , $n \times K_1$), and the last K_2 columns are designated as the hidden covariates (X_2 , $n \times K_2$).
- B_2: effect size matrix. Each entry is drawn from $N(0, 1)$ and scaled (see below).

Lastly, in the noise component, we have

- E: noise matrix. Each entry is drawn from $N(0, 1)$ and scaled (see below).

Alternatively, (3.3.1) can be written as

$$\underset{n\times 1}{(Y)_j} = \underset{n\times q}{S_j}\ \underset{q\times 1}{(IB_1)_j} + \underset{n\times(K_1+K_2)}{X}\ \underset{(K_1+K_2)\times 1}{(B_2)_j} + \underset{n\times 1}{(E)_j}\ ,\ j = 1, \cdots, p\,, \tag{3.3.3}$$

where $(Y)_j$, $(IB_1)_j$, $(B_2)_j$, and $(E)_j$ denote the jth column of Y, $I \odot B_1$, B_2 , and E, respectively, and S_j denotes $SArray[\,,\,,\,j]$.

The scaling for B_2 and E is to ensure that `PVEGenotype` and `PVECovariates` are as desired. Specifically, for gene j, if $\mathrm{Var}\,(S_j\ (IB_1)_j) \neq 0$, then we scale $(B_2)_j$ so that

$$\frac{\mathrm{Var}\,(X\ (B_2)_j)}{\mathrm{Var}\,(S_j\ (IB_1)_j)} = \frac{\texttt{PVECovariates}}{\texttt{PVEGenotype}} \tag{3.3.4}$$

and separately scale $(E)_j$ so that

$$\frac{\mathrm{Var}\,((E)_j)}{\mathrm{Var}\,(S_j\ (IB_1)_j)} = \frac{1 - \texttt{PVEGenotype} - \texttt{PVECovariates}}{\texttt{PVEGenotype}}\,. \tag{3.3.5}$$

If $\mathrm{Var}\,(S_j\ (IB_1)_j) = 0$ (which is the case when $(IB_1)_j$ is a zero vector, i.e., when gene j has zero effect SNPs), then we only scale $(E)_j$ so that

$$\frac{\mathrm{Var}\left((E)_j\right)}{\mathrm{Var}\left(X\ (B_2)_j\right)} = \frac{1 - \texttt{PVECovariates}}{\texttt{PVECovariates}}\,. \tag{3.3.6}$$

3.3.2 Definition of truth

In a simulated data set, I is a $q \times p$ binary matrix. The lj-th entry being one means that the lth local common SNP for gene j is an effect SNP for gene j. However, due to LD, the expression level of a gene may be strongly associated with SNPs other than its effect SNPs.

Therefore, we define I_{cor}, also a $q \times p$ binary matrix, based on $SArray$ and I and use it as the truth when calculating AUPRCs. The lj-th entry of I_{cor} is defined as one if and only if the lth local common SNP for gene j is highly correlated with *any* of gene j's effect SNPs (correlation ≥ 0.9 in absolute value).

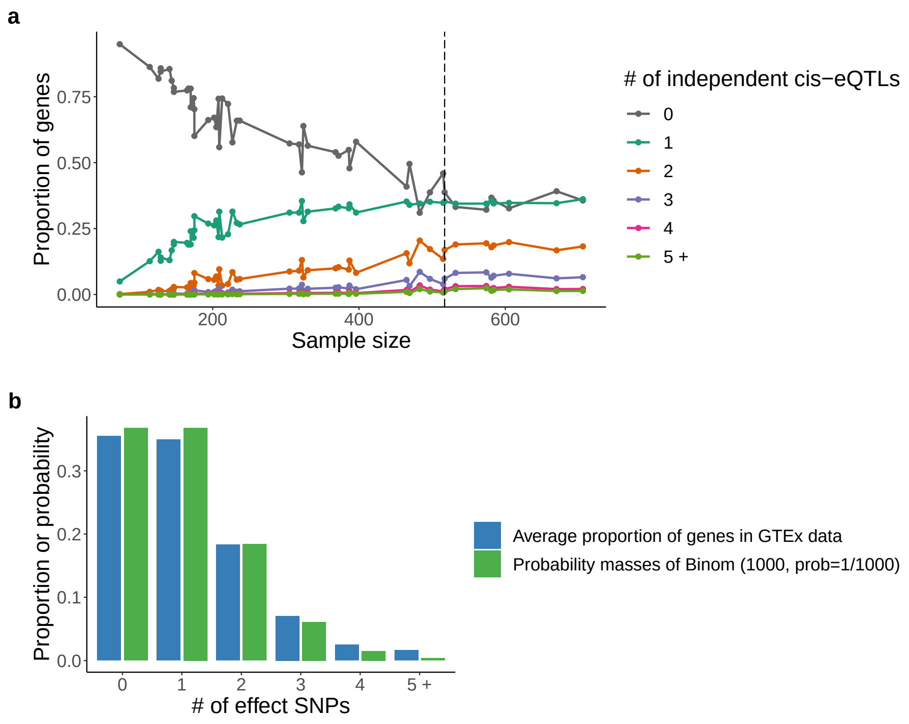

Figure 3.2: In Simulation Design 2, we find that $Binom\,(1000, \text{prob} = 1/1000)$ approximates the empirical distribution of the number of independent cis-eQTLs per gene in GTEx data [4] well. **a** Given a tissue type, which corresponds to a sample size, we plot the proportion of genes with 0, 1, 2, 3, 4, or 5 or more independent cis-eQTLs (the proportions add up to one; data from GTEx [4]). We find that the proportions stabilize once the sample size reaches about 517 (dashed line). **b** For the eight tissue types with sample size ≥ 517, we take the average proportion of genes with 0 independent cis-eQTLs, 1 independent cis-eQTL, etc. and plot them in the blue bars. The green bars represent the probability mass function of $Binom\,(1000, \text{prob} = 1/1000)$ (with the tail probabilities combined together).

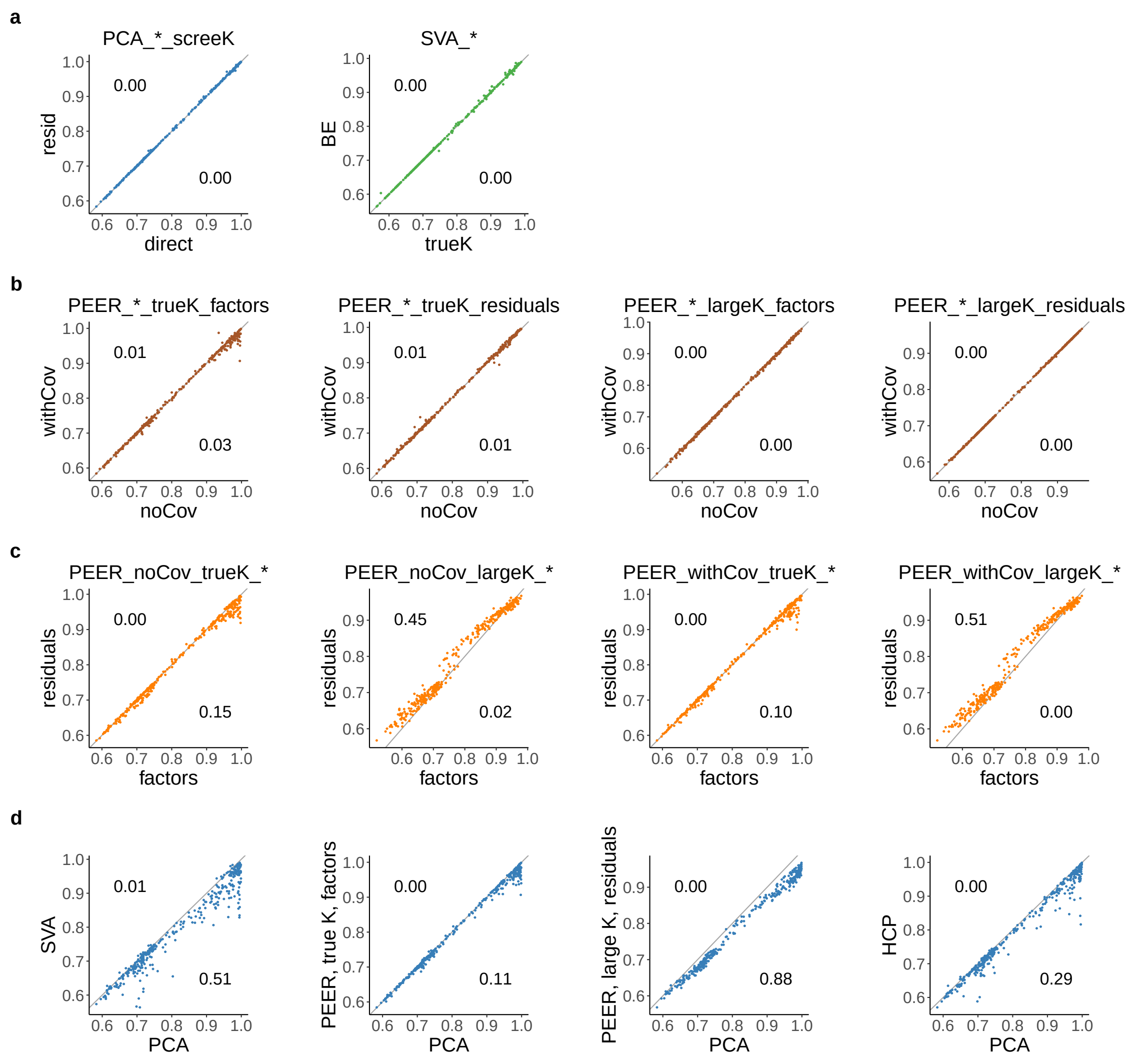

Figure 3.3: This figure shows how we select a few representative methods from the 15 methods for detailed comparison in Simulation Design 2 (**a, b, c**) and a dataset-by-dataset comparison of the selected representative methods (**d**). The x-axis and y-axis both represent AUPRCs of different methods. Each scatter plot contains 352 points, each of which corresponds to a simulated data set in Simulation Design 2. The number on the upper-left corner of each scatter plot represents the proportion of points that satisfy $y > 1.02\ x$, and the number on the lower-right corner represents the proportion of points that satisfy $x > 1.02\ y$, where x and y denote the coordinates of each point. **a** The two PCA methods perform almost identically, so for simplicity, we select PCA_direct_screeK. The two SVA methods perform almost identically as well, so we select SVA_BE. **b** Whether the known covariates are inputted when PEER is run has little effect on the AUPRC. **c** When we use the true K, the factor approach outperforms the residual approach, but when we use a large K, the residual approach outperforms the factor approach. Therefore, we select PEER_withCov_trueK_factors and PEER_withCov_largeK_residuals as the representative PEER methods. **d** Among the selected representative methods, PCA outperforms SVA, PEER, and HCP in terms of AUPRC in 11% to 88% of the simulated data sets and underperforms them in close to 0% of the simulated data sets.

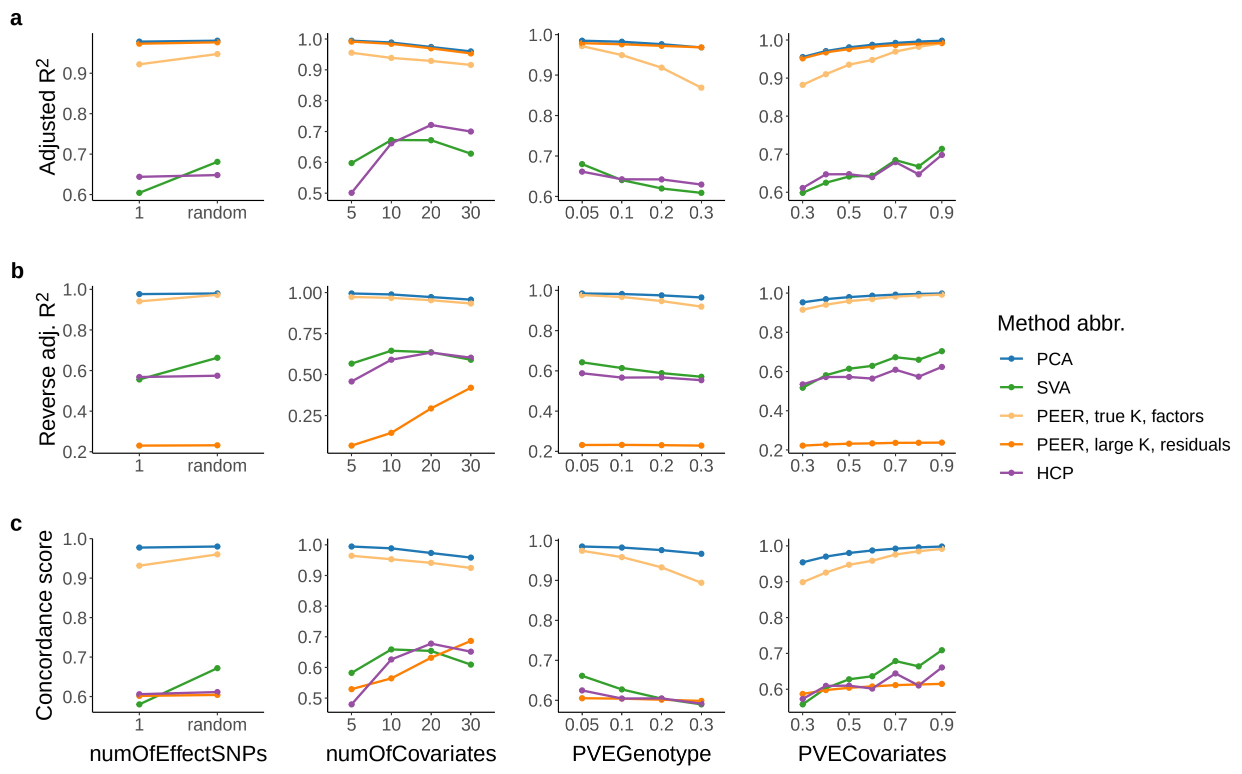

Figure 3.4: Detailed adjusted R^2, reverse adjusted R^2, and concordance score comparison of the selected representative methods (Table 2.1) in Simulation Design 2. Each point represents the average across simulated data sets. PCA performs the best in all three regards. PEER with a large K (dark orange line) performs well in terms of adjusted R^2 but falls short in terms of reverse adjusted R^2.

3.4 Compared methods

We compare the runtime and performance of 15 methods based on simulation studies, including Ideal, Unadjusted, and 13 variants of PCA, SVA, PEER, and HCP (Table 2.1). The details of Simulation Design 1 and Simulation Design 2 are described in Sections 3.2 and 3.3, respectively. Recall that in each simulated data set, Y denotes the gene expression matrix ($n \times p$, sample by gene), X_1 denotes the known covariate matrix ($n \times K_1$, sample by covariate), and X_2 denotes the hidden covariate matrix ($n \times K_2$, sample by covariate). The genotype information is stored in S in Simulation Design 1 and $SArray$ in Simulation

Design 2. In this work, we use K to denote the number of inferred covariates, which are called PCs, SVs, PEER factors, and HCPs in PCA, SVA, PEER, and HCP, respectively.

Given a simulated data set, each of the 15 methods consists of two steps: hidden variable inference step (not applicable for Ideal and Unadjusted) and QTL step. In the hidden variable inference step, we run PCA, SVA, PEER, or HCP to obtain the inferred covariates (and the expression residuals in the case of PEER; Figure 2.1). In the QTL step, given a gene-SNP pair, we run a linear regression with the gene expression vector (or the residual vector from PEER) as the *response* and the genotype vector and covariates as *predictors*, where the choice of the response and covariates depends on the method (Table 2.1); thus we obtain the p-value for the null hypothesis that the coefficient corresponding to the genotype vector is zero given the covariates. In Simulation Design 1, we investigate the association between each gene's expression level and each SNP in the *entire genome* for a simultaneous detection of cis-QTL and trans-QTL relations. In Simulation Design 2, we investigate the association between each gene's expression level and each of the gene's *local common SNPs* for a cis-QTL analysis.

For Ideal, we assume that X_2 is known. Therefore, we use X_1 and X_2 as covariates in the QTL step. For Unadjusted, we use X_1 as the covariates.

We devise two ways to use PCA to account for the hidden covariates. For PCA_direct_screeK, we run PCA on Y directly. For PCA_resid_screeK, we first residualize Y against X_1 and then run PCA on the residual matrix. In this work, PCA is run *with* centering and scaling unless otherwise specified; given A, an $n \times p_1$ matrix, and B, an $n \times p_2$ matrix, both observation by feature, to *residualize A against B* means to take each column of A, regress it against B, and replace the original column of A with the residuals from the linear regression. For both methods, since the scree plots always suggest the true "number of hidden covariates" ($K_1 + K_2$ for PCA_direct_screeK, K_2 for PCA_resid_screeK) as the reasonable number of PCs to choose within plus or minus one (usually exactly; Figure 3.1), we set the number of PCs to be the true "number of hidden covariates". For PCA_direct_screeK, we filter out the known covariates that are captured well by the top PCs (unadjusted $R^2 \geq 0.9$) and use the remaining known covariates along with the top PCs as covariates in the QTL step. For PCA_resid_screeK, no filtering is needed.

Here we describe the two hidden variable inference methods for SVA: SVA_trueK and SVA_BE. Since the SVA package [44] requires the user to input at least one variable of interest (Figure 2.1) and using too many variables of interest causes the package to fail, when running SVA, we input the top PC of the genotype matrix (S in Simulation Design 1, collapsed version of $SArray$ in Simulation Design 2) as the variable of interest. We also input X_1 as the known covariates because the package documentation indicates that the known covariates should be provided if available. The SVA package allows the user to specify K. Alternatively, it can automatically choose K using a slightly modified version of the Buja and Eyuboglu (BE) algorithm [15, 55]. Therefore, in SVA_trueK, we set $K = K_2$, and in SVA_BE, we let the package choose K automatically. In both cases, we use X_1 and the surrogate variables (SVs) as covariates in the QTL step.

There are several different ways to use PEER [1] but no consensus in the literature on which one is the best. In the hidden variable inference step, PEER can be run with or without the known covariates when there are known covariates available (Stegle et al. [1] do not give an explicit recommendation as to which approach should be used, and both approaches are used in practice [2–4, 32]), and K has to be specified by the user (Stegle et al. [1, 5] claim that the performance of PEER does not deteriorate as K increases). In the QTL step, one can include the PEER factors as covariates (we call this the "factor approach") or use the expression residuals outputted by PEER as the response (and not use any known or inferred covariates; we call this the "residual approach"). For completeness, we compare $2^3 = 8$ ways of using PEER (the default priors are always used): PEER is run with or without the known covariates; PEER is run using the true "number of hidden covariates" ($K_1 + K_2$ when PEER is run without the known covariates, K_2 when PEER is run with the known covariates) or using a large K (K=50); and either the factor approach or the residual approach is used in the QTL step.

The HCP package requires the user to specify K and three tuning parameters: λ_1, λ_2, and λ_3 (Section 3.5.2). The package documentation suggests choosing K and the tuning parameters via a grid search. However, no specific recommendations are given regarding the choice of the score function. In practice, users of HCP often choose K and the tuning parameters by maximizing the number of discoveries [38, 39]. For our simulation studies, such an approach would be computationally prohibitive. Therefore, for simplicity, we set $K = K_2$ and $\lambda_1 = \lambda_2 = \lambda_3 = 1$; the latter is because we do not want to give more weight to the penalty terms than the main term in the objective function (Section 3.5.2).

Reference	**GTEx data version**	**QTL analysis**	**Data transformation**	**Known covariates inputted**	**# of PEER factors**	**Factor or residual approach**
(A)	(B)	(C)	(D)	(E)	(F)	(G)
GTEx Consortium [3]	V6p	eQTL (cis and trans)	INT within feature	No	Maximizes cis-eGenes	Factor
GTEx Consortium [4]	V8	eQTL (cis and trans)	INT within feature	No	Maximizes cis-eGenes	Factor
		sQTL (cis and trans)	INT within sample	No	15	Factor
Li et al. [2]	V7	$3'$aQTL (cis)	No transformation	Yes	Follows GTEx [3]	Factor

Table 3.3: Summary of QTL analyses performed by GTEx [3, 4] and Li et al. [2]. "INT" in (D) stands for "inverse normal transform" [18]. (E), (F), and (G) summarize how PEER is used (Section 3.4) in each study. GTEx [3, 4] chooses the number of PEER factors for its eQTL analyses (including cis and trans) by maximizing the number of discovered cis-eGenes for each pre-defined sample size bin. The number of PEER factors selected is 15 for $n < 150$, 30 for $n \in [150, 250)$, and 35 for $n \geq 250$ for GTEx V6p eQTL analyses [3] and 15 for $n < 150$, 30 for $n \in [150, 250)$, 45 for $n \in [250, 350)$, and 60 for $n \geq 350$ for GTEx V8 eQTL analyses [4], where n denotes the sample size. Li et al. [2] use the numbers of PEER factors chosen by GTEx [3].

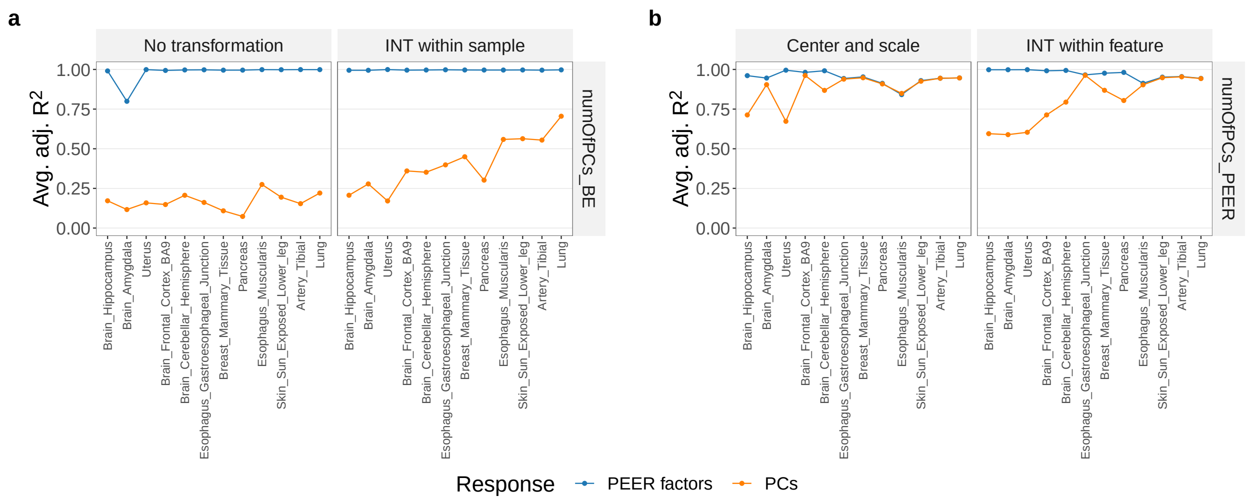

Figure 3.5: In the 3′aQTL data prepared by Li et al. [2] from GTEx RNA-seq reads [3], PEER factors fail to capture important variance components of the molecular phenotype data when the data transformation method is “No transformation” or “INT within sample” (**a**; the numbers of PCs are chosen via BE (Algorithm 3)). On the other hand, PEER factors span roughly the same linear subspace as the top PCs when the data transformation method is “Center and scale” or “INT within feature”, but the top PCs can almost always capture the PEER factors better than the PEER factors can capture the top PCs (**b**; the numbers of PCs are equal to the numbers of PEER factors). Given m PEER factors and n PCs from the same post-transformation molecular phenotype matrix ($m \geq n$ in **a**, $m = n$ in **b**), we calculate m adjusted R^2's by regressing each PEER factor against the PCs and plot the average in blue. Similarly, we calculate n adjusted R^2's by regressing each PC against the PEER factors and plot the average in orange.

Algorithm 1: Reordering of PEER factors based on PCs (Figure 2.5).

```
  Inputs:
    • $K$ PEER factors.
    • $K$ PCs.
  Output: $K$ PEER factors (reordered).
1 for $k \leftarrow 1$ to $K$ do
2     Select the PEER factor that is the most highly correlated with the $k$th PC from
      the PEER factors that have not been selected yet.
3 end
4 return the PEER factors in the order that they were selected in.
```

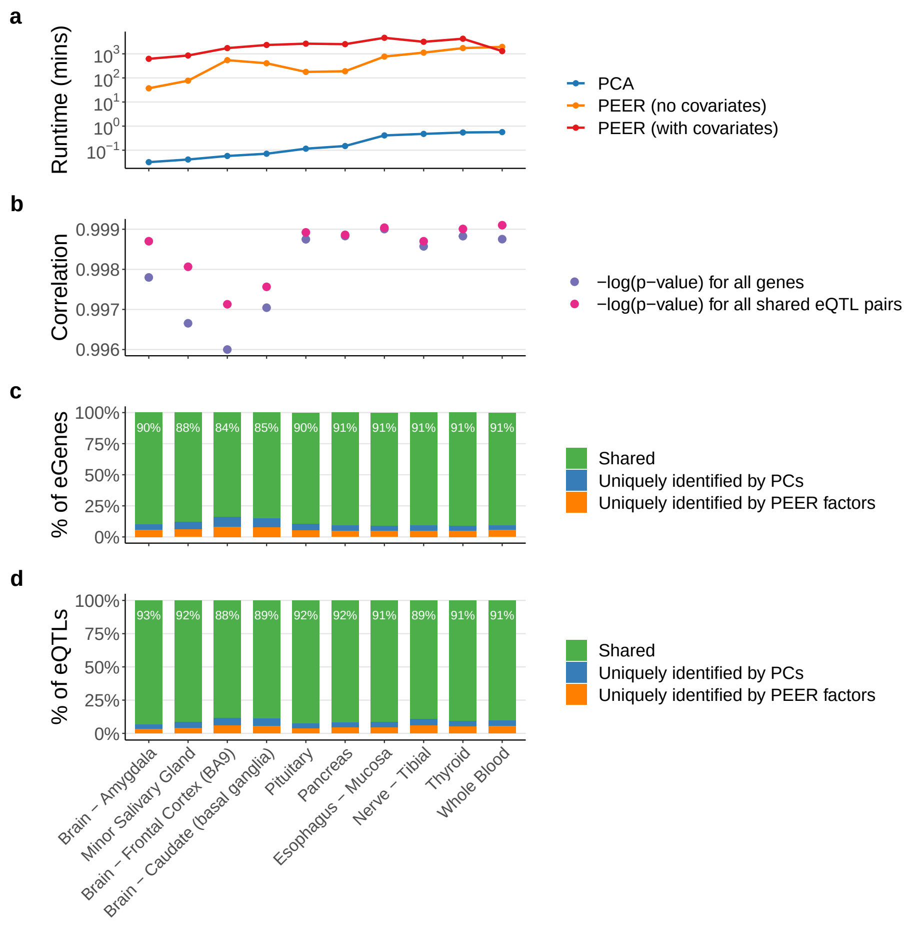

Figure 3.6: In GTEx eQTL data [4], PEER is at least three orders of magnitude slower than PCA (**a**), and replacing the PEER factors with PCs in GTEx's FastQTL pipeline [4, 17] does not change the cis-eQTL results much (**b, c, d**). The x-axis shows 10 randomly selected tissue types with increasing sample sizes. **a** For a given gene expression matrix, running PEER without the known covariates (GTEx's approach) takes up to about 1,900 minutes (equivalent to about 32 hours; Whole Blood), while running PCA (with centering and scaling; our approach) takes no more than a minute. For comparison, we also run PEER *with* the known covariates using the numbers of PEER factors selected by GTEx. This approach takes even longer (up to about 4,600 minutes, equivalent to about 77 hours; Esophagus - Mucosa). **b** The p-values produced by GTEx's approach and our approach are highly correlated (correlations between the negative common logarithms are shown). **c, d** The overlap of the identified eGenes and eQTL pairs between the two approaches is generally around 90% (see Figure 3.7 for more detail).

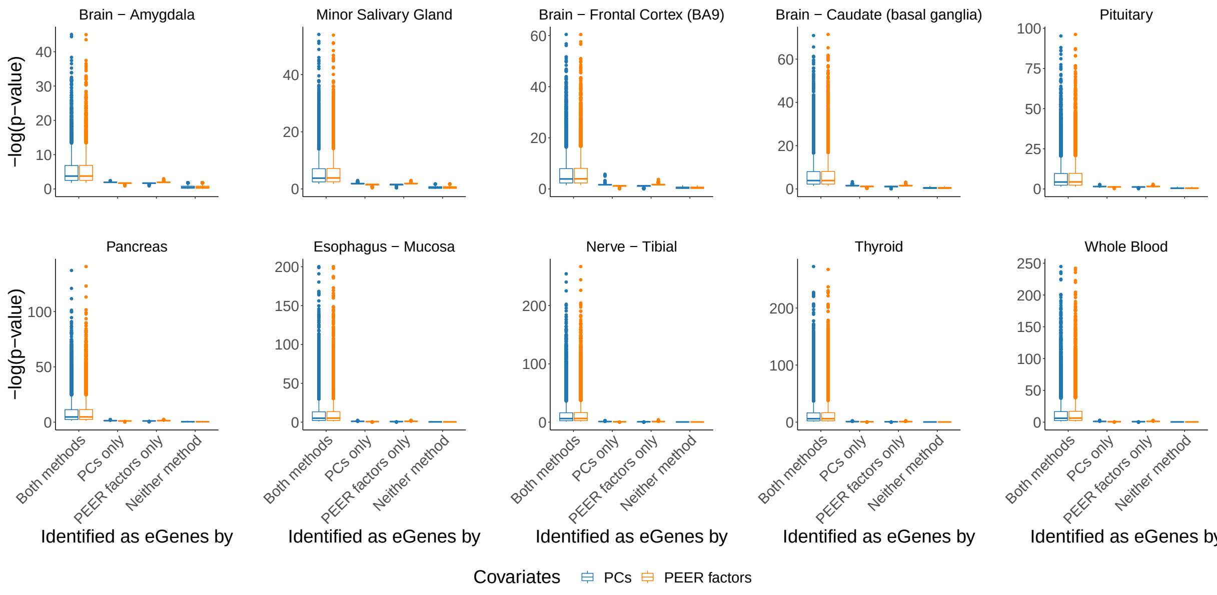

Figure 3.7: Following the analysis in Figure 3.6, we find that the eGenes uniquely identified by PCs or PEER factors have marginal p-values compared to those identified by both methods.

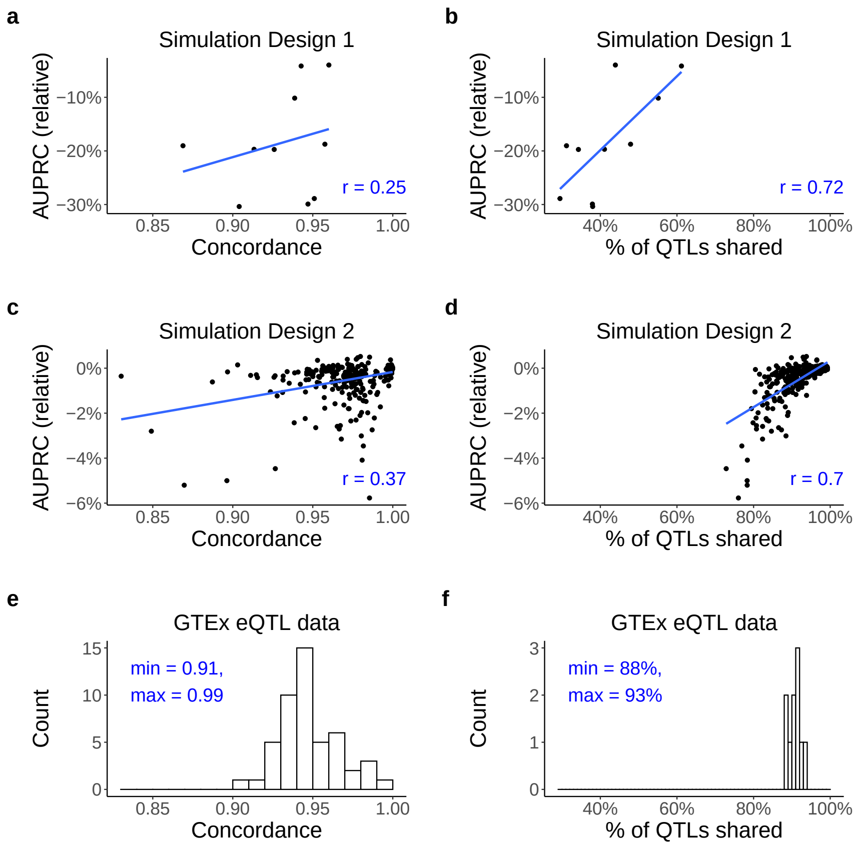

Figure 3.8: Joint analysis of results from Simulation Design 1, Simulation Design 2, and GTEx eQTL data [4]. The x-axes of **a**, **c**, and **e** show the concordance between PEER factors and top PCs (defined analogously as the concordance score; Section 2.2.1). The x-axes of **b**, **d**, and **f** show the percentage of QTL discoveries shared between PEER and PCA (in **b** and **d**, for each method, binary decisions are made based on p-values using the Benjamini-Hochberg (BH) procedure and a target false discovery rate of 0.05). In **a** through **d**, the y-axes show $(\text{AUPRC}_{\text{PEER}} - \text{AUPRC}_{\text{PCA}})/\text{AUPRC}_{\text{PCA}}$, the blue lines are the simple linear regression lines, and the Pearson correlation coefficients are shown on the bottom right. **a** and **b** each contains 10 data points, corresponding to the 10 simulated data sets in Simulation Design 1. **c** and **d** each contains 352 data points, corresponding to the 352 simulated data sets in Simulation Design 2. The methods compared in **a** through **d** are PCA_direct_screeK and PEER_noCov_trueK_factors. **e** presents similar information as Figure 2.5; the total count is 49, which is the number of tissue types with GTEx eQTL analyses. **f** is based on Figure 3.6(d); the total count is 10, which is the number of tissue types randomly selected for analysis in Figure 3.6. We find that the percentage of QTL discoveries shared is a good predictor of the relative performance of PEER versus PCA and is a better predictor than concordance. This plot is also evidence that Simulation Design 2 is more realistic than Simulation Design 1 because the ranges that concordance and percentage of QTL discoveries shared fall in in **e** and **f** agree better with those in **c** and **d** than those in **a** and **b**.

3.5 Theory of PCA and HCP

3.5.1 Principal component analysis (PCA)

Principal component analysis (PCA) [6, 7] is a well-established dimension reduction method with many applications. Here we aim to provide a brief summary of its algorithm, derivation, and interpretation.

Let X denote the $n \times p$ observed data matrix that is observation by feature, e.g., a molecular phenotype matrix. We use X instead of Y here to be consistent with standard PCA notations. We assume that the columns of X have been centered and scaled. That is, X satisfies

$$\frac{1}{n}\sum_{i=1}^{n} x_{ij} = 0\,,\ j = 1, \cdots, p \tag{3.5.1}$$

and

$$\frac{1}{n-1}\sum_{i=1}^{n} x_{ij}^2 = 1\,,\ j = 1, \cdots, p\,, \tag{3.5.2}$$

where x_{ij} denotes the ij-th entry of X.

The PCA algorithm consists of two steps. In the first step, we calculate the sample covariance matrix $\widehat{\Sigma}$ and perform eigendecomposition on it:

$$\begin{aligned} \widehat{\Sigma} &= \frac{1}{n}X^\top X && \text{definition of sample covariance matrix} && (3.5.3)\\ &:= Q\Lambda Q^\top\,, && \text{eigendecomposition} && (3.5.4) \end{aligned}$$

where

$$\underset{p\times p}{Q} = \begin{bmatrix} | & & | \\ q_1 & \cdots & q_p \\ | & & | \end{bmatrix} \tag{3.5.5}$$

is an orthogonal matrix whose columns are eigenvectors of $\widehat{\Sigma}$, and

$$\underset{p\times p}{\Lambda} = \begin{bmatrix} \lambda_1 & & \\ & \ddots & \\ & & \lambda_p \end{bmatrix}, \ \lambda_1 \geq \cdots \geq \lambda_p \geq 0\,, \tag{3.5.6}$$

is a diagonal matrix whose diagonal entries are the corresponding eigenvalues of $\widehat{\Sigma}$. We know that $\widehat{\Sigma}$ is orthogonally diagonalizable because it is a symmetric matrix (recall the spectral theorem for real matrices [58]: a real matrix is orthogonally diagonalizable if and only if it is symmetric). The eigenvalues are all non-negative because $\widehat{\Sigma}$ is positive semidefinite.

In the second step, we calculate Z as

$$Z = XQ\,, \tag{3.5.7}$$

where the columns of Z are called the *principal components* (PCs) or *scores*, and Q is called the *loading matrix* or *rotation matrix*. It is worth noting that some authors may refer to $q_1, \cdots, q_p$ as the PCs. This use of terminology is confusing and should be avoided [8].

The above two steps conclude the PCA algorithm. In practice, however, singular value decomposition (SVD) of the data matrix is often used as a more computationally efficient way of finding the loading matrix and the PCs [6].

The most common derivation of PCA is based on maximum variance [9]. First, we define

$\alpha_1^*, \cdots, \alpha_p^* \in \mathbb{R}^p$ sequentially as

$$\alpha_1^* = \arg\max_{\alpha_1 \in \mathbb{R}^p} \operatorname{Var}(X\alpha_1) \qquad \text{subject to } \|\alpha_1\|_2 = 1\,, \tag{3.5.8}$$

$$\alpha_2^* = \arg\max_{\alpha_2 \in \mathbb{R}^p} \operatorname{Var}(X\alpha_2) \qquad \text{subject to } \|\alpha_2\|_2 = 1\,,\ \alpha_2^\top \alpha_1^* = 0\,, \tag{3.5.9}$$

$$\vdots$$

$$\alpha_p^* = \arg\max_{\alpha_p \in \mathbb{R}^p} \operatorname{Var}(X\alpha_p) \qquad \text{subject to } \|\alpha_p\|_2 = 1\,,\ \alpha_p^\top \alpha_j^* = 0\ \ \forall\, j < p\,. \tag{3.5.10}$$

Then, we define the PCs of X as $X\alpha_1^*, \cdots, X\alpha_p^*$. That is, the PCs are defined sequentially as the linear combinations of the columns of X with maximum variances, subject to certain constraints. It can then be shown that $\alpha_1^*, \cdots, \alpha_p^*$ are given by $q_1, \cdots, q_p$ respectively, where $q_1, \cdots, q_p$ are eigenvectors of $\widehat{\Sigma}$ as defined in (3.5.5).

A complementary property of PCA, which is closely related to the original discussion of Pearson [10], is the minimum reconstruction error property. Given $K < p$, we define Q_K as the matrix that contains the first K columns of Q. That is,

$$\underset{p\times K}{Q_K} := \begin{bmatrix} | & & | \\ q_1 & \cdots & q_K \\ | & & | \end{bmatrix}\,. \tag{3.5.11}$$

The minimum reconstruction error property of PCA states that Q_K is a global minimizer of the loss function

$$\mathcal{J}\left(\widetilde{Q}_K\right) := \left|\left|\left| X - X\widetilde{Q}_K\widetilde{Q}_K^\top \right|\right|\right|_F^2 \tag{3.5.12}$$

$$= \sum_{i=1}^n \left\| x_i^\top - x_i^\top \widetilde{Q}_K \widetilde{Q}_K^\top \right\|_2^2 = \sum_{i=1}^n \left\| x_i - \widetilde{Q}_K \widetilde{Q}_K^\top x_i \right\|_2^2\,, \tag{3.5.13}$$

where $\widetilde{Q}_K$ denotes an arbitrary $p \times K$ matrix whose columns are orthonormal, $|||\cdot|||_F$ denotes the Frobenius norm of a matrix, and $x_i^\top$ denotes the ith row of X. Since $\widetilde{Q}_K\widetilde{Q}_K^\top x_i$

represents the (orthogonal) projection of x_i onto the subspace spanned by the columns of $\widetilde{Q}_K$, (3.5.13) measures the total squared ℓ_2 error when approximating each x_i with its projection onto the subspace spanned by the columns of $\widetilde{Q}_K$.

A central idea of PCA is the proportion of variance explained by each PC. To establish this concept, we claim that

$$\sum_{j=1}^{p} \mathrm{Var}\left(X_j\right) = \sum_{j=1}^{p} \mathrm{Var}\left(Z_j\right) \,, \tag{3.5.14}$$

$$\mathrm{Var}\left(Z_j\right) = \lambda_j \,,\ j = 1, \cdots, p \,, \tag{3.5.15}$$

and

$$\mathrm{Cov}\left(Z_j \,,\ Z_{j'}\right) = 0 \,,\ j, j' = 1, \cdots, p \,,\ j \neq j' \,, \tag{3.5.16}$$

where X_j denotes the jth column of X (the jth original variable) and Z_j denotes the jth column of Z (the jth PC). (3.5.16) means that the PCs are uncorrelated with each other.

We prove (3.5.15) and (3.5.16) by calculating $\widehat{\Sigma}_Z$, the sample covariance matrix of Z (we know that the columns of Z are centered by (3.5.1) and (3.5.7)):

$$\begin{aligned}
\widehat{\Sigma}_Z &= \frac{1}{n} Z^\top Z && \text{definition of sample covariance matrix} && (3.5.17)\\
&= \frac{1}{n} \left(XQ\right)^\top XQ && \text{plugging in (3.5.7)} && (3.5.18)\\
&= Q^\top \left(\frac{1}{n} X^\top X\right) Q && && (3.5.19)\\
&= Q^\top \left(Q \Lambda Q^\top\right) Q && \text{plugging in (3.5.4)} && (3.5.20)\\
&= \Lambda \,. && && (3.5.21)
\end{aligned}$$

(3.5.14) can be proven by the following:

$$\begin{aligned}
\sum_{j=1}^{p} \operatorname{Var}(X_j) &= \operatorname{Tr}\left(\widehat{\Sigma}\right) && \text{definition of trace and } \widehat{\Sigma} && (3.5.22)\\
&= \operatorname{Tr}\left(Q\Lambda Q^\top\right) && \text{plugging in (3.5.4)} && (3.5.23)\\
&= \operatorname{Tr}\left(\Lambda Q^\top Q\right) && \text{cyclic property of trace} && (3.5.24)\\
&= \operatorname{Tr}(\Lambda) && && (3.5.25)\\
&= \sum_{j=1}^{p} \operatorname{Var}(Z_j) \ . && \text{by (3.5.15)} && (3.5.26)
\end{aligned}$$

Because of (3.5.14) and (3.5.15), we may define the proportion of variance in the original data explained by the jth PC as

$$\frac{\lambda_j}{\sum_{j'=1}^{p} \operatorname{Var}(X_{j'})} = \frac{\lambda_j}{\sum_{j'=1}^{p} \operatorname{Var}(Z_{j'})} = \frac{\lambda_j}{\sum_{j'=1}^{p} \lambda_{j'}} \ , \tag{3.5.27}$$

which provides a basis for deciding the number of PCs to keep (e.g., Algorithms S2 and S3).

3.5.2 Hidden covariates with prior (HCP) and its connection to PCA

Hidden covariates with prior (HCP) [16] is a popular hidden variable inference method for QTL mapping defined by minimizing a loss function. Neither Mostafavi et al. [16] nor the HCP package documents the HCP method well. For example, the squares in the loss function (3.5.28) are missing in both Mostafavi et al. [16] and the package documentation, but one can deduce that the squares are there by inspecting the coordinate descent steps in the source code of the R package. Here we aim to provide a better, more accurate documentation of the HCP method and point out its connection to PCA.

Given Y, the molecular phenotype matrix ($n \times p$, sample by feature), X_1, the known covariate matrix ($n \times K_1$, sample by covariate), K, the number of inferred covariates (HCPs), and $\lambda_1,\ \lambda_2,\ \lambda_3 > 0$, the tuning parameters, HCP looks for

$$\underset{X_2,\ W_1,\ W_2}{\arg\min} \left\{ \left|\left|\left| \underset{n\times p}{Y} - \underset{n\times K}{X_2} \underset{K\times p}{W_2} \right|\right|\right|_F^2 + \lambda_1 \left|\left|\left| \underset{n\times K}{X_2} - \underset{n\times K_1}{X_1} \underset{K_1\times K}{W_1} \right|\right|\right|_F^2 + \lambda_2 |||W_1|||_F^2 + \lambda_3 |||W_2|||_F^2 \right\}, \tag{3.5.28}$$

where $|||\cdot|||_F$ denotes the Frobenius norm of a matrix, X_2 is the hidden covariate matrix, and W_1 and W_2 are weight matrices of the appropriate dimensions. The name of the method, "hidden covariates with prior", comes from the second term in (3.5.28), where the method informs the hidden covariates with the known covariates. The optimization is done through coordinate descent with one deterministic initialization (see source code of the HCP R package [16]). The columns of the obtained X_2 are reported as the HCPs.

From (3.5.28), we see that the HCP method is closely related to PCA. The first term in (3.5.28) is very similar to (3.5.12), the only difference being that the rows of W_2 in (3.5.28) are not required to be orthonormal and X_2 is not required to be equal to $YW_2^\top$.

Algorithm 2: The elbow method for choosing K in PCA (based on distance to diagonal line).

```
  Input: $X$, $n \times p$ observed data matrix, observation by feature.
  Output: $K$, the number of PCs selected.
1 Define $d = \texttt{min}(n, p)$. This is the total number of PCs.
2 Run PCA on $X$ with centering and scaling.
3 Obtain the proportion of variance explained by each PC, $t_1, \cdots, t_d$.   // $\sum_{j=1}^d t_j = 1$.
4 Consider $(1, t_1), \cdots, (d, t_d) \in \mathbb{R}^2$.   // $d$ points in $\mathbb{R}^2$.
5 Select $K$ by choosing the point that is the farthest from the diagonal line, i.e., the
    line that passes through the first point, $(1, t_1)$, and the last point, $(d, t_d)$.
    Specifically, the distance from $(x_0, y_0)$ to the line that passes through $(x_1, y_1)$ and
    $(x_2, y_2)$ is given by
    $|\ (x_2 - x_1)(y_1 - y_0) - (x_1 - x_0)(y_2 - y_1)\ |\ /((x_2 - x_1)^2 + (y_2 - y_1)^2)^{1/2}$.
6 return $K$.
```

Algorithm 3: The Buja and Eyuboglu (BE) algorithm [55] for choosing K in PCA.

```
Inputs:
  • $X$, $n \times p$ observed data matrix, observation by feature.
  • $B$, number of permutations (default is 20).
  • $\alpha$, significance level (default is 0.05).
Output: $K$, the number of PCs selected.
1  Define $d = \texttt{min}\,(n, p)$. This is the total number of PCs.
2  Run PCA on $X$ with centering and scaling.
3  Obtain the proportion of variance explained (PVE) by each PC, $t_1, \cdots, t_d$.
     // $\sum_{j=1}^{d} t_j = 1$.  Observed test statistics.
4  for $b \leftarrow 1$ to $B$ do
5      Obtain $X^{(b)}$ by permuting each column of $X$.   // Permute the observations in each feature.
6      Run PCA on $X^{(b)}$ with centering and scaling.
7      Obtain the PVE of each PC, $t_1^{(b)}, \cdots, t_d^{(b)}$.   // $\sum_{j=1}^{d} t_j^{(b)} = 1$.
8  end
9  The $p$-value for the $j$th PC is calculated as
     $p_j = \left(\sum_{b=1}^{B} \mathbb{1}\{t_j^{(b)} \geq t_j\} + 1\right) / (B+1)\,,\; j = 1, \cdots, d\,.$   // $p_j$ is calculated as,
     roughly speaking, the proportion of permutations where the PVE of the $j$th PC is
     greater than or equal to the PVE of the $j$th original PC (the added ones in the
     numerator and denominator are mainly for avoiding $p$-values that are exactly
     zero).  The greater this proportion is, the larger the $p$-value is, and the less
     significant the PC is.
10 for $j \leftarrow 2$ to $d$ do
11     If $p_j \leq p_{j-1}$, then set $p_j = p_{j-1}$.   // Enforce monotone increase of the $p$-values.
12 end
13 Set $K$ to be the maximum $j$ such that $p_j \leq \alpha$.
14 return $K$.
```

CHAPTER 4

ClipperQTL: ultrafast and powerful eGene identification method

4.1 Results

4.1.1 Real data results

We compare the performance and run time of different variants of FastQTL, eigenMT, TreeQTL, and ClipperQTL (Table 4.1) on the most recent GTEx expression data [4]. The 49 tissues with sample sizes above 70 are considered [4]. For each gene, we consider single nucleotide polymorphisms (SNPs) within one megabase (Mb) of the transcription start site (TSS) of the gene [4]; we use 0.01 as the threshold for the minor allele frequency (MAF) of a SNP and 10 as the threshold for the number of samples with at least one copy of the minor allele (MA samples) [17]. We include eight known covariates and a number of top expression PCs (principal components) as inferred covariates [19]. The eight known covariates are the top five genotype PCs, WGS sequencing platform (HiSeq 2000 or HiSeq X), WGS library construction protocol (PCR-based or PCR-free), and donor sex [4]. The number of expression PCs is chosen via the Buja and Eyuboglu (BE) algorithm [19, 55] for each tissue. We use the BE algorithm because we find that in our simulated data (Section 5.2), the BE algorithm can recover the true number of covariates well. The target FDR for eGene identification is set at 0.05. We do not include Matrix eQTL [29] in our real data comparison because both our simulation study (Section 4.1.2) and Huang et al. [59] show that Matrix eQTL cannot control the FDR in the eGene identification problem.

The results from our real data analysis are summarized in Figures 4.1, 4.2, and 5.2. We find that the four variants of FastQTL produce almost identical results as one another. Specifically, the numbers of eGenes identified by the four methods are almost identical (Figure 4.1), and the identified eGenes highly overlap (Figure 5.2). This means the adaptive permutation scheme and the beta approximation of FastQTL (Section 5.1.2) are not critical to the performance of FastQTL; the simplest variant, FastQTL_1K_prop, is sufficient. Further, we find that eigenMT and TreeQTL methods identify fewer eGenes than FastQTL (Figure 4.1). In contrast, ClipperQTL methods produce almost identical results as FastQTL in tissues with the appropriate sample sizes (Section 4.2.2; Figures 4.1 and 5.2).

In terms of run time comparison (Figure 4.2), we find that eigenMT has almost no computational advantage over FastQTL, and TreeQTL has no computational advantage over the standard variant of ClipperQTL (which is slower than the Clipper variant of ClipperQTL). Both the standard variant and the Clipper variant of ClipperQTL are orders of magnitude faster than FastQTL. In particular, the standard variant of ClipperQTL is about five times faster than FastQTL_1K_prop—the simplest FastQTL method—even though the algorithms are equivalent (Section 4.2.2); we attribute this to differences in software implementation. Compared to the default FastQTL method, the standard variant and the Clipper variant of ClipperQTL are about 50 times and 500 times faster, respectively.

4.1.2 Simulation results

In our simulation study, we roughly follow the data simulation in the second, more realistic simulation design of Zhou et al. [19], which roughly follows the data simulation in Wang et al. [20]. We simulate three data sets in total. Each data set is simulated according to Algorithm 9 with sample size $n = 838$, number of genes $p = 1000$, number of covariates $\widetilde{K} = 20$, proportion of variance explained by genotype in eGenes `PVEGenotype` $= 0.02$, and proportion of variance explained by covariates `PVECovariates` $= 0.5$. All covariates are assumed to be known covariates.

The results from our simulation study are summarized in Figure 4.3. We confirm the finding in Huang et al. [59] that Matrix eQTL cannot control the FDR in the eGene identification problem. All other methods can approximately control the FDR. Further, FastQTL and ClipperQTL methods have higher power than eigenMT and TreeQTL methods, consistent with our real data results (Section 4.1.1).

4.2 Methods

4.2.1 Problem

Here we describe the eGene identification problem and introduce the notations for this work.

The input data are as follows. Let Y denote the $n \times p$ fully processed gene expression matrix with n samples and p genes. For gene j, $j = 1, \cdots, p$, the relevant genotype data is stored in S_j, the $n \times q_j$ genotype matrix, where each column of S_j corresponds to a local common SNP for gene j (conceptually speaking; in reality, all genotype data may be stored in one file). Let X denote the $n \times K$ covariate matrix with K covariates. Using our analysis of GTEx's Colon - Transverse expression data [4] (Section 4.1.1) as an example, we have $n = 368$, $p = 25{,}379$, q_j typically under 15,000, and $K = 37$, including eight known covariates and 29 inferred covariates (the number of inferred covariates is chosen via the BE algorithm [19, 55]; Section 4.1.1).

The assumption is that for $j = 1, \cdots, p$, $Y[\,,\,j]$, the jth column of Y, is a realization of the following random vector:

$$\underset{n\times 1}{\mathbb{1}}\ \underset{1\times 1}{\beta_{0j}} + \underset{n\times q_j}{S_j}\ \underset{q_j\times 1}{\beta_{1j}} + \underset{n\times \widetilde{K}}{\widetilde{X}}\ \underset{\widetilde{K}\times 1}{\beta_{2j}} + \underset{n\times 1}{\epsilon_j}\ , \tag{4.2.1}$$

where $\mathbb{1}$ denotes the $n \times 1$ matrix of ones, S_j is defined as above, $\widetilde{X}$ is the true covariate matrix (which X tries to capture), all entries of β_{0j}, β_{1j}, and β_{2j} are fixed but unknown parameters, and ϵ_j is the random noise. In particular, it is assumed that at most a small number of entries of β_{1j} are nonzero [20]. If all entries of β_{1j} are zero, then gene j *is not* an eGene. On the other hand, if at least one entry of β_{1j} is nonzero, then gene j *is* an eGene. The goal is to identify which of the p genes are eGenes given Y, $\{S_j\}_{j=1}^{p}$, and X.

4.2.2 ClipperQTL

We propose two main variants of ClipperQTL: the standard variant and the Clipper variant. The standard variant is equivalent to FastQTL with the direct permutation scheme and proportions (Algorithm 5) and is suitable for a wide range of sample sizes. The Clipper variant uses the contrastive strategy developed in Clipper [51] (Algorithm 4) and is only recommended for data sets with large sample sizes (> 450). The development of ClipperQTL is discussed in Section 5.3. A key technical difference between the standard variant and the Clipper variant is that in the standard variant, gene expression is permuted first and then residualized, whereas in the Clipper variant, gene expression is residualized first and then permuted.

The main input parameter of ClipperQTL under both variants is B, the number of permutations. For the standard variant, B is set at 1000 by default. For the Clipper variant, we recommend setting B between 20 and 100 (Figures 5.3 and 5.4).

Algorithm 4: The Clipper variant of ClipperQTL

```
Inputs:
  • $Y$, $\{S_j\}_{j=1}^p$, and $X$ (gene expression, genotype, and covariate data, respectively; Section 4.2.1).
  • $B > 1$, the number of permutations (default is 20).
1 for $j \leftarrow 1$ to $p$ do
2     Regress $Y[\,,\,j]$ against $X$ and denote the residuals as $(Y[\,,\,j])_{\text{resid}}$, an $n \times 1$ matrix.
3     Regress each column of $S_j$ against $X$ and save the residuals in $(S_j)_{\text{resid}}$, an $n \times q_j$ matrix.
4     Calculate $R_j := \texttt{abs}\Big(\texttt{cor}\big((Y[\,,\,j])_{\text{resid}}\,,(S_j)_{\text{resid}}\big)\Big)$, a $1 \times q_j$ matrix, where abs and cor denote the absolute value function and the correlation function in R, respectively. That is, the $l$th entry of $R_j$, $l = 1, \cdots, q_j$, is the absolute value of the correlation between $(Y[\,,\,j])_{\text{resid}}$ and the $l$th column of $(S_j)_{\text{resid}}$.
5     Define $r_j := \max(R_j)$, the maximum of all values in $R_j$. This is equivalent to $|r_{j(1)}|$ in Algorithm 5.
6     for $b \leftarrow 1$ to $B$ do
7         Permute $(Y[\,,\,j])_{\text{resid}}$ to obtain $(Y[\,,\,j])^b_{\text{resid}}$.
8         Calculate $R^b_j := \texttt{abs}\Big(\texttt{cor}\big((Y[\,,\,j])^b_{\text{resid}}\,,(S_j)_{\text{resid}}\big)\Big)$, a $1 \times q_j$ matrix.
9         Define $r^b_j := \max\left(R^b_j\right)$.
10    end
11 end
12 Run Clipper [51] to call eGenes using $\{r_j\}_{j=1}^p$ as measurements under the experimental condition and $\{r^1_j\}_{j=1}^p, \cdots, \{r^B_j\}_{j=1}^p$ as measurements under the background condition (Section 1.2); use enrichment analysis, GZ procedure, maximum contrast score, and $h = 1$ (Section 5.3).
```

4.3 Tables and figures

	Method category	**Method**	**Note**	**Method name for speed comparison**
	(A)	(B)	(C)	(D)
1	Matrix eQTL	Matrix eQTL		
2	FastQTL	FastQTL_1K-10K_prop		FastQTL_1K-10K
3		FastQTL_1K-10K_beta	Default FastQTL method	
4		FastQTL_1K_prop		FastQTL_1K
5		FastQTL_1K_beta		
6	eigenMT	eigenMT		eigenMT
7	TreeQTL	TreeQTL_BY	Default TreeQTL method	TreeQTL
8		TreeQTL_Storey		
9	ClipperQTL	ClipperQTL_standard_1K		ClipperQTL_standard_1K
10		ClipperQTL_Clipper_20		ClipperQTL_Clipper_20
11		ClipperQTL_Clipper_50		ClipperQTL_Clipper_50

Table 4.1: Summary of the 11 eGene identification methods we compare. Details of these methods can be found in Sections 4.2.2 and 5.1.

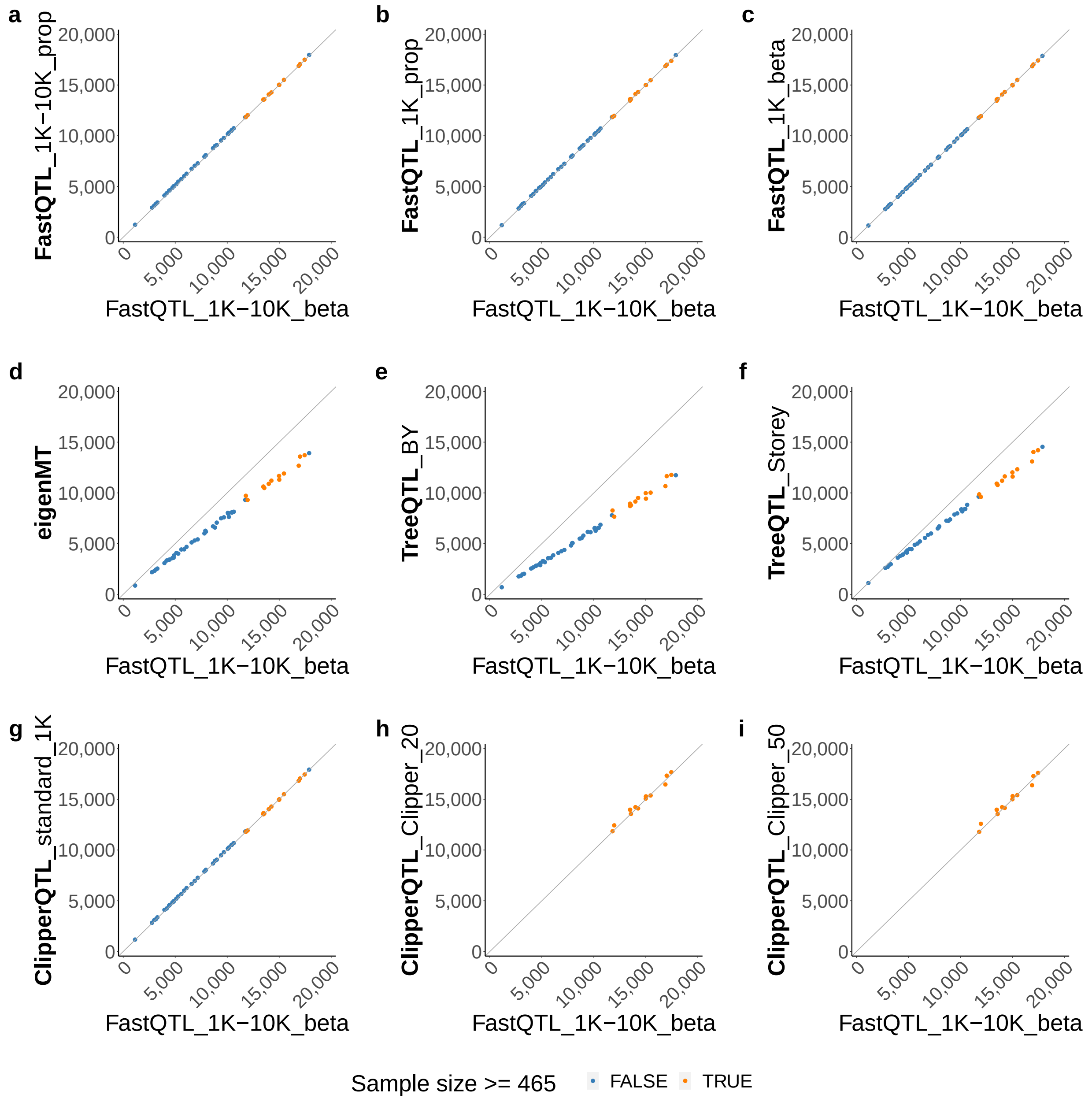

Figure 4.1: Number of eGenes comparison based on GTEx expression data [4] (Table 4.1; see Section 4.1.1 for the analysis details). Each dot corresponds to a tissue. The x-axis and y-axis both represent numbers of eGenes identified by different methods. Diagonal lines through the origin are shown to help with visualization. **a-c** The four variants of FastQTL identify almost the same numbers of eGenes as one another. **d-f** eigenMT and TreeQTL methods identify fewer eGenes than FastQTL. **g-i** ClipperQTL methods identify almost the same numbers of eGenes as FastQTL in tissues with the appropriate sample sizes (Section 4.2.2). We use 465 as the sample size cutoff because the next largest sample size is 396. See Figure 5.2 for an analysis of the overlap between identified eGenes.

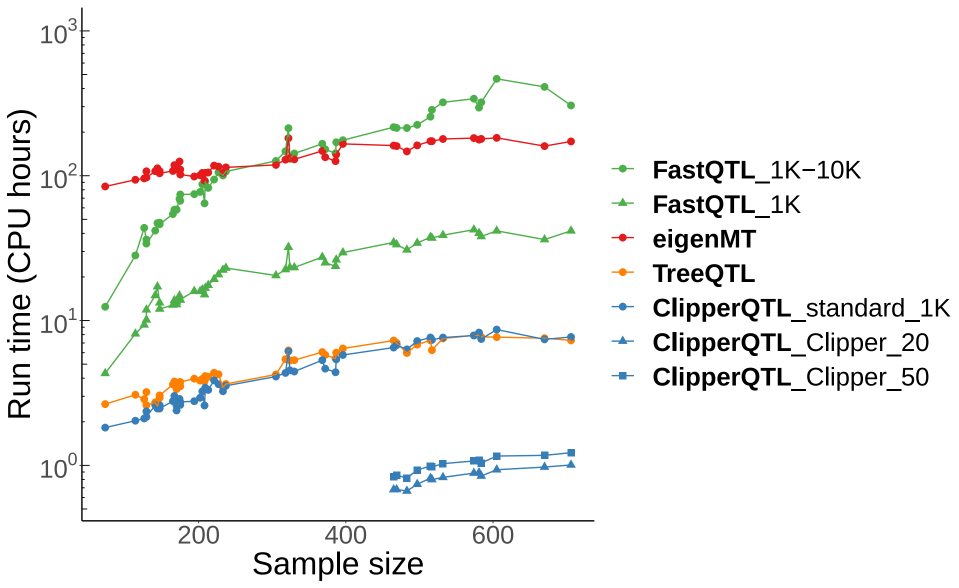

Figure 4.2: Run time comparison based on GTEx expression data [4] (Table 4.1; see Section 4.1.1 for the analysis details). Each dot corresponds to a tissue. FastQTL_1K-10K takes under 500 CPU hours. FastQTL_1K takes under 50 CPU hours. ClipperQTL_standard_1K takes under 10 CPU hours. ClipperQTL_Clipper_20 takes under 1 CPU hour. Run times of ClipperQTL_Clipper_20 and ClipperQTL_Clipper_50 are only shown for tissues with sample sizes ≥ 465 (Figure 4.1).

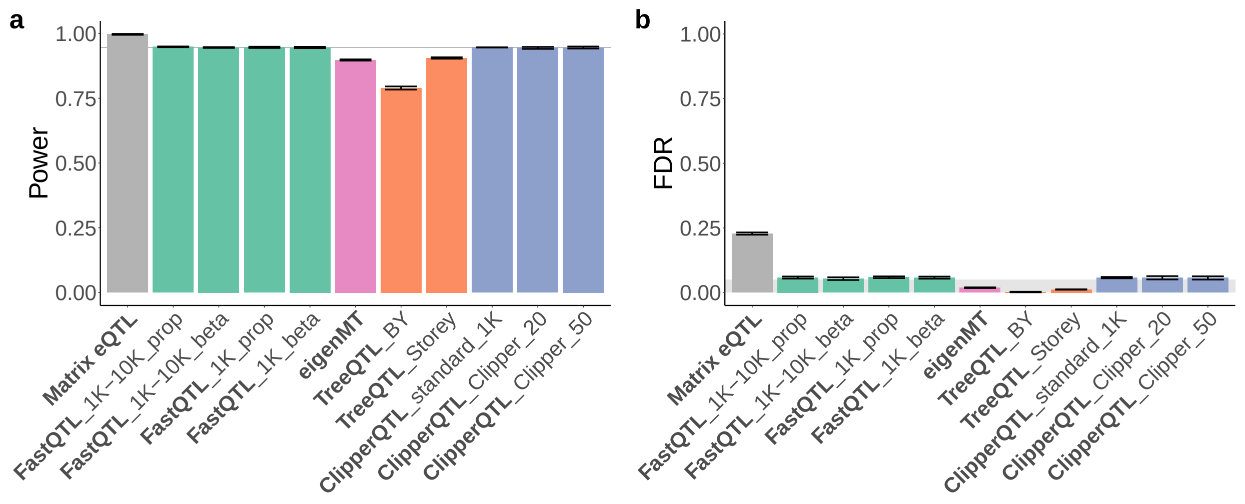

Figure 4.3: Power and FDR comparison of all 11 methods based on our simulation study (Table 4.1; Section 4.1.2). The target FDR is set at 0.05 (grey shaded area in **b**). The height of each bar represents the average across simulated data sets. Error bars indicate standard errors. In **a**, a horizontal line at the height of the bar for FastQTL_1K-10K_beta is shown to help with visualization. All methods except Matrix eQTL can approximately control the FDR. FastQTL and ClipperQTL methods have higher power than eigenMT and TreeQTL methods.

4.4 Availability of data and materials

The R package ClipperQTL is available at `https://github.com/heatherjzhou/ClipperQTL`. The code used to generate the results in this work is available at `https://doi.org/10.5281/zenodo.8259929`. In addition, this work makes use of the following data and software:

- GTEx V8 public data [4], including fully processed gene expression matrices and known covariates, are downloaded from `https://gtexportal.org/home/datasets`.
- GTEx V8 protected data [4], specifically, the whole genome sequencing (WGS) phased genotype data, are downloaded from the AnVIL repository with an approved dbGaP application (see `https://gtexportal.org/home/protectedDataAccess`).
- FastQTL (`https://github.com/francois-a/fastqtl`, accessed October 29, 2020).
- Matrix eQTL R package Version 2.3 (`https://cran.r-project.org/web/packages/`

`MatrixEQTL`, accessed March 6, 2023).

- eigenMT (`https://github.com/joed3/eigenMT`, accessed March 6, 2023).
- TreeQTL R package Version 2.0 (`https://bioinformatics.org/treeqtl`, accessed March 6, 2023).

CHAPTER 5

Supplementary materials for ClipperQTL: ultrafast and powerful eGene identification method

5.1 Existing eGene identification methods

In this section, we review the existing eGene identification methods and describe the variants that we compare in this work (Table 2.1).

Recall the notations from Section 4.2.1. Let Y denote the $n \times p$ fully processed gene expression matrix with n samples and p genes. For gene j, $j = 1, \cdots, p$, the relevant genotype data is stored in S_j, the $n \times q_j$ genotype matrix, where each column of S_j corresponds to a local common SNP for gene j. Let X denote the $n \times K$ covariate matrix with K covariates.

5.1.1 Matrix eQTL

Conceptually speaking, Matrix eQTL [29] works as follows: for $j = 1, \cdots, p$, $l = 1, \cdots, q_j$, run the linear regression represented by the following R `lm()` formula:

$$Y[\,,\,j] \;\sim\; S_j[\,,\,l] \;+\; X \tag{5.1.1}$$

and obtain the p-value for the null hypothesis that the coefficient corresponding to $S_j[\,,\,l]$ is zero given the covariates; denote this p-value as p_{jl}. Therefore, a total of $\sum_{j=1}^{p} q_j$ p-values are obtained, one for each gene-SNP pair. Matrix eQTL then uses the Benjamini-Hochberg (BH) procedure [60] on these p-values to call significant gene-SNP pairs [29]. To call eGenes using Matrix eQTL in this work, we call as eGenes all genes that appear at least once in the significant gene-SNP pairs.

In reality, Matrix eQTL uses the following equivalent approach to obtain the p-values, which is more computationally efficient due to the overlap of local common SNPs across genes. For $j = 1, \cdots, p$, $l = 1, \cdots, q_j$, first, regress the gene expression against the covariates:

$$Y[\,,\, j] \;\sim\; X. \tag{5.1.2}$$

Second, regress the genotype against the covariates:

$$S_j[\,,\, l] \;\sim\; X. \tag{5.1.3}$$

Then, calculate the Pearson correlation between the expression residuals from (5.1.2) and the genotype residuals from (5.1.3) and denote it as r_{jl}. This is the partial correlation between $Y[\,,\, j]$ and $S_j[\,,\, l]$ conditional on X.

Lastly, convert the partial correlation to a test statistic using

$$t_{jl} = r_{jl}\sqrt{\frac{n-2-K}{1-r_{jl}^2}} \tag{5.1.4}$$

and convert the test statistic to a p-value using

$$p_{jl} = 2 \times \mathbb{P}\left(T \geq |t_{jl}|\right), \;\; T \sim t_{n-2-K}\,, \tag{5.1.5}$$

where $\mathbb{P}$ denotes probability, $|t_{jl}|$ denotes the absolute value of t_{jl}, and T is a random variable following the t-distribution with $n-2-K$ degrees of freedom.

Notably, the larger $|r_{jl}|$ (the absolute value of r_{jl}), the smaller p_{jl}, and vice versa. Both

FastQTL (the variants using proportions; Section 5.1.2) and ClipperQTL (the standard variant; Section 4.2.2) make use of this fact.

5.1.2 FastQTL

There are four main ways to use FastQTL [17], depending on (1) whether the direct or the adaptive permutation scheme is used and (2) whether proportions or beta approximation is used. The direct permutation scheme with either proportions or beta approximation is summarized in Algorithm 5. The adaptive permutation scheme is identical except the number of permutations is chosen adaptively between $B_{\min}$ and $B_{\max}$ (two input parameters) for each gene rather than directly inputted (see Ongen et al. [17] for details).

The default way of using FastQTL is to use the adaptive permutation scheme ($B_{\min} = 1000$ and $B_{\max} = 10{,}000$) with beta approximation [4, 17]. In total, we compare four ways of using FastQTL in this work including the default approach: FastQTL_1K-10K_prop, FastQTL_1K-10K_beta (the default), FastQTL_1K_prop, and FastQTL_1K_beta (see Table 2.1). That is, the number of permutations is either fixed at 1000 or chosen adaptively between 1000 and 10,000 for each gene, and either proportions or beta approximation is used.

In addition to identifying eGenes, FastQTL can also output significant gene-SNP pairs. We summarize the algorithm for this in Algorithm 6.

Algorithm 5: The direct permutation scheme of FastQTL

```
Inputs:
  • $Y$, $\{S_j\}_{j=1}^{p}$, and $X$ (gene expression, genotype, and covariate data, respectively;
    Section 4.2.1).
  • $B$, the number of permutations.
1  for $j \leftarrow 1$ to $p$ do
2      Obtain $r_{j1}, \cdots, r_{jq_j}$, the partial correlation between $Y[\,, j]$ and
         $S_j[\,, 1], \cdots, S_j[\,, q_j]$ (respectively) conditional on $X$ (Section 5.1.1).
3      Denote the one with the largest absolute value as $r_{j(1)}$.
4      for $b \leftarrow 1$ to $B$ do
5          Permute $Y[\,, j]$ (leave $S_j$ unchanged).
6          Obtain $r^b_{j1}, \cdots, r^b_{jq_j}$, the partial correlation between $Y[\,, j]$ after permutation
             and $S_j[\,, 1], \cdots, S_j[\,, q_j]$ (respectively) conditional on $X$.
7          Denote the one with the largest absolute value as $r^b_{j(1)}$.
8      end
9      if using proportions then
10         $\widetilde{p}_j$, the gene-level $p$-value for gene $j$, is defined as
             $\widetilde{p}_j := \left(\sum_{b=1}^{B} \mathbb{1}\left\{|r^b_{j(1)}| \geq |r_{j(1)}|\right\} + 1\right) / (B+1)$.   // Roughly the proportion
             of permutations with more extreme outcomes.  The addition of one in the
             numerator and the denominator helps avoid $p$-values that are exactly zero.
11     else if using beta approximation then
12         Find $true\_df \in (0, \infty)$, which is to replace $n-2-K$ when converting $r_{j(1)}$
             and $\{r^b_{j(1)}\}_{b=1}^{B}$ to $p$-values using (5.1.4) and (5.1.5).   // See the source code
             of FastQTL for how $true\_df$ is defined.  In a nutshell, $true\_df$ minimizes
             the absolute difference between 1 and the method of moments estimate for
             the first shape parameter of the beta distribution from the $p$-values.
13         Convert $r_{j(1)}$ and $\{r^b_{j(1)}\}_{b=1}^{B}$ to $p$-values using (5.1.4) and (5.1.5) with
             $n-2-K$ replaced by $true\_df$. Denote these $p$-values as $p_{j(1)}$ and $\{p^b_{j(1)}\}_{b=1}^{B}$.
14         Fit a beta distribution to $\{p^b_{j(1)}\}_{b=1}^{B}$ using maximum likelihood estimation.
             Denote the cumulative distribution function of the fitted beta distribution
             as $F_j$.
15         $\widetilde{p}_j$, the gene-level $p$-value for gene $j$, is defined as $\widetilde{p}_j := F_j\left(p_{j(1)}\right)$.
16     end
17 end
18 Use Storey's $q$-value [52] on $\{\widetilde{p}_j\}_{j=1}^{p}$ to call eGenes.
```

```
Algorithm 6: Identification of significant gene-SNP pairs in FastQTL
  Input:
    • Intermediate and final results from Algorithm 5 using either the direct or the
      adaptive permutation scheme and beta approximation (rather than proportions).
1 Define $p_t$ (following the notation of the GTEx Consortium [4]) as the average of the
    gene-level $p$-value of the most significant non-eGene and the gene-level $p$-value of
    the least significant eGene (see the source code of FastQTL). That is, $p_t$ is defined
    as the average of two of $\widetilde{p}_1, \cdots, \widetilde{p}_p$.
2 for $j \leftarrow 1$ to $p$ do
3     if gene $j$ is identified as an eGene then
4         Define $threshold_j := F_j^{-1}(p_t)$, where $F_j^{-1}$ denotes the inverse function of $F_j$.
5         Convert $r_{j1}, \cdots, r_{jq_j}$ (Line 2 of Algorithm 5) to $p$-values using (5.1.4) and
            (5.1.5) (without replacing $n - 2 - K$ with $true\_df$).
6         If the $p$-value corresponding to a SNP is less than or equal to $threshold_j$,
            then gene $j$ and this SNP are together identified as a significant gene-SNP
            pair.
7     end
8 end
```

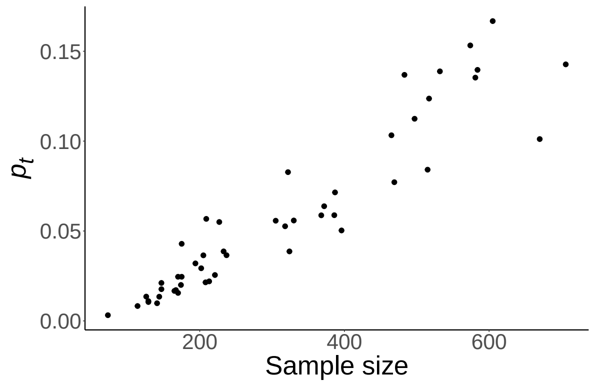

Figure 5.1: Scatter plot of p_t (Algorithm 6) from FastQTL_1K-10K_beta versus sample size in GTEx expression data [4] (see Section 4.1.1 for the analysis details). This scatter plot contains 49 dots, each corresponding to a tissue. We see that p_t increases roughly linearly with sample size.

5.1.3 eigenMT

We summarize eigenMT [47] in Algorithm 7. After obtaining the gene-level p-values, eigenMT does not specify what method to use to control the false discovery rate when calling eGenes. Therefore, in this work, we use Storey's q-value [52] following FastQTL (Section 5.1.2). In addition to producing the gene-level p-values, eigenMT also outputs the most significantly associated SNP for each gene.

5.1.4 TreeQTL

TreeQTL [48] uses Simes' rule [50] to calculate a gene-level p-value for each gene (Algorithm 8). After obtaining the gene-level p-values, TreeQTL allows the user to use Bonferroni correction, BH [60], or Benjamini-Yekutieli (BY) [62] to call eGenes (the default is BY).

We compare two variants of TreeQTL in this work: TreeQTL_BY (the default) and TreeQTL_Storey (see Table 2.1). In TreeQTL_Storey, we use Storey's q-value [52] on the gene-level p-values to call eGenes, following FastQTL (Section 5.1.2). We do not include variants of TreeQTL using Bonferroni correction or BH in our comparison because Bonferroni correction aims to control the family-wise error rate rather than the false discovery rate, and BH is more stringent than Storey's q-value (we show that even TreeQTL_Storey has lower power than FastQTL; Figures 4.1 and 4.3).

5.2 Data simulation

In our simulation study, we roughly follow the data simulation in the second, more realistic simulation design of Zhou et al. [19], which roughly follows the data simulation in Wang et al. [20]. We simulate three data sets in total. Each data set is simulated according to Algorithm 9 with the following attributes:

- Sample size, $n = 838$.

Algorithm 7: eigenMT

```
Inputs:
  • Results from Matrix eQTL, i.e., $p_{jl}$ , $j = 1, \cdots, p$ , $l = 1, \cdots, q_j$ (Section 5.1.1).
  • $\{S_j\}_{j=1}^{p}$ , genotype data (Section 4.2.1).
  • $W$, window size for partitioning genotype data (default is 200).
  • $C$, threshold for the cumulative proportion of variance explained (default is 0.99).
1  for $j \leftarrow 1$ to $p$ do
2      Consider $p_{j1}, \cdots, p_{jq_j}$ . Denote the smallest one as $p_{j(1)}$ .
3      Initialize $q_j^{\text{eff}}$ , the effective number of local common SNPs for gene $j$ , at 0. The
         goal is to calculate $q_j^{\text{eff}}$ from $S_j$ .
4      Break $S_j$ vertically into $\lceil \frac{q_j}{W} \rceil$ chunks, each chunk an $n \times W$ matrix except the
         last chunk, which may have fewer columns.
5      for $i \leftarrow 1$ to $\lceil \frac{q_j}{W} \rceil$ do
6          Denote the number of columns in the $i$th chunk of $S_j$ as $W_{ji}$ (if $i < \lceil \frac{q_j}{W} \rceil$,
             then $W_{ji} = W$).
7          if $W_{ji} = 1$ then
8              $q_j^{\text{eff}} \leftarrow q_j^{\text{eff}} + 1$.
9          else
10             Obtain the Ledoit-Wolf estimate [61] of the covariance matrix of the $i$th
                 chunk of $S_j$ using sklearn.covariance.LedoitWolf() in Python.
11             Convert the estimated covariance matrix to a correlation matrix.
12             Obtain the eigenvalues of the correlation matrix, $\lambda_1 \geq \cdots \geq \lambda_{W_{ji}}$ , using
                 scipy.linalg.eigvalsh() in Python.
13             Set all negative eigenvalues (if any) to 0. // See the source code of
                   eigenMT.
14             $q_j^{\text{eff}} \leftarrow q_j^{\text{eff}} + \arg\min_K \left( \frac{\sum_{k=1}^{K} \lambda_k}{W_{ji}} \geq C \right)$ . That is, increment $q_j^{\text{eff}}$ by the
                   minimum number of top eigenvalues required to pass the threshold for
                   the cumulative proportion of variance explained. // See the source
                   code of eigenMT.
15         end
16     end
17     $\widetilde{p}_j$ , the gene-level $p$-value for gene $j$ , is defined as $\widetilde{p}_j := \min\left(p_{j(1)} \times q_j^{\text{eff}}, 1\right)$ .
18 end
```

Algorithm 8: TreeQTL

```
Input:
    • Results from Matrix eQTL, i.e., $p_{jl}$, $j = 1, \cdots, p$, $l = 1, \cdots, q_j$ (Section 5.1.1).
1 for $j \leftarrow 1$ to $p$ do
2     Consider $p_{j1}, \cdots, p_{jq_j}$. Denote the order statistics as $p_{j(1)}, \cdots, p_{j(q_j)}$, with $p_{j(1)}$
        being the smallest and $p_{j(q_j)}$ being the largest.
3     $\widetilde{p}_j$, the gene-level $p$-value for gene $j$, is defined as $\widetilde{p}_j := \min_{l=1,\cdots,q_j} p_{j(l)} \frac{q_j}{l}$,
        following Simes' rule [50].
4 end
5 Use Bonferroni correction, BH [60], or BY [62] on $\{\widetilde{p}_j\}_{j=1}^{p}$ to call eGenes (the default
    is BY).
```

- Number of genes, $p = 1000$.
- Number of covariates, $\widetilde{K} = 20$.
- Proportion of variance explained by genotype in eGenes, `PVEGenotype` $= 0.02$.
- Proportion of variance explained by covariates, `PVECovariates` $= 0.5$.

# of effect SNPs	**Probability**
0	0.35483532
1	0.34962617
2	0.18326554
3	0.07072812
4	0.02498728
5	0.01655758

Table 5.1: In our data simulation (Algorithm 9), the number of effect SNPs [19, 20] for each gene is sampled based on this probability table (the second column sums to one). This table is summarized from GTEx's independent cis-eQTL analysis [4] (see Figure S2 of Zhou et al. [19]). A gene is an eGene if and only if its number of effect SNPs is greater than zero.

5.3 Development of ClipperQTL

Here we describe the development of ClipperQTL.

Clipper [51] has four main technical parameters:

- Analysis: enrichment vs. differential analysis.

Algorithm 9: Simulation of one data set

```
1  Randomly select $p$ genes from GTEx's Brain - Cortex expression data [4], avoiding genes from the
     X chromosome [19, 20].
2  The goal is to simulate $Y$, the $n \times p$ gene expression matrix.
3  Simulate $\widetilde{X}$, the $n \times \widetilde{K}$ true covariate matrix, by drawing each entry independently from $N(0,1)$.
     In this work, all random sampling is independent unless otherwise specified.
4  for $j \leftarrow 1$ to $p$ do
5      Obtain $S_j$ , the $n \times q_j$ genotype matrix for gene $j$ , by subsetting GTEx V8 genotype data [4].
         Each column of $S_j$ corresponds to a local common SNP for gene $j$ . "Local" means the SNP is
         on the same chromosome as the gene and is located within one megabase (Mb) of the
         transcription start site (TSS) of the gene. "Common" means the minor allele frequency
         (MAF) of the SNP is at least 0.01 and the number of samples with at least one copy of the
         minor allele (MA samples) is at least 10.
6      Sample $\widetilde{q}_j$ , the number of effect SNPs [19, 20] for gene $j$ , based on Table 5.1.
7      if $\widetilde{q}_j = 0$ then
8          Generate $Y[\,,j]$ based on
             $$Y[\,,j] = \underset{n\times\widetilde{K}}{\widetilde{X}} \; \underset{\widetilde{K}\times 1}{\beta_{2j}} + \underset{n\times 1}{e_j} \,, \qquad (5.2.1)$$
           where each entry of $\beta_{2j}$ is drawn from $N(0,1)$, and each entry of $e_j$ is drawn from $N(0,1)$
           and scaled. The scaling is to ensure that PVECovariates is as desired. Specifically, we
           scale $e_j$ so that
             $$\frac{\mathrm{Var}\,(e_j)}{\mathrm{Var}\left(\widetilde{X}\beta_{2j}\right)} = \frac{1 - \texttt{PVECovariates}}{\texttt{PVECovariates}} \,. \qquad (5.2.2)$$
9      else
10         Randomly select $\widetilde{q}_j$ columns of $S_j$ and designate them as the effect SNPs of gene $j$ .
11         Generate $Y[\,,j]$ based on
             $$Y[\,,j] = \underset{n\times q_j}{S_j} \; \underset{q_j\times 1}{\beta_{1j}} + \underset{n\times\widetilde{K}}{\widetilde{X}} \; \underset{\widetilde{K}\times 1}{\beta_{2j}} + \underset{n\times 1}{e_j} \,, \qquad (5.2.3)$$
           where entries of $\beta_{1j}$ that don't correspond to the effect SNPs of gene $j$ are set to 0, and
           entries of $\beta_{1j}$ that correspond to the effect SNPs of gene $j$ are each drawn from $N(0,1)$.
           Further, each entry of $\beta_{2j}$ is drawn from $N(0,1)$ and scaled, and each entry of $e_j$ is drawn
           from $N(0,1)$ and scaled. The scaling is to ensure that PVEGenotype and PVECovariates
           are as desired. Specifically, we scale $\beta_{2j}$ so that
             $$\frac{\mathrm{Var}\left(\widetilde{X}\beta_{2j}\right)}{\mathrm{Var}\,(S_j\beta_{1j})} = \frac{\texttt{PVECovariates}}{\texttt{PVEGenotype}} \qquad (5.2.4)$$
           and separately scale $e_j$ so that
             $$\frac{\mathrm{Var}\,(e_j)}{\mathrm{Var}\,(S_j\beta_{1j})} = \frac{1 - \texttt{PVEGenotype} - \texttt{PVECovariates}}{\texttt{PVEGenotype}} \,. \qquad (5.2.5)$$
12     end
13 end
14 A gene is an eGene if and only if its number of effect SNPs is greater than zero.
```

- Procedure: Barber-Candès (BC) vs. Gimenez-Zou (GZ) procedure.
- Contrast score: maximum vs. difference (i.e., minus) contrast score.
- h (only applicable under the GZ procedure, not the BC procedure).

In addition, in ClipperQTL, we can control B, the number of permutations (ClipperQTL terminology), i.e., the number replicates under the background condition (Clipper terminology).

In ClipperQTL, we use enrichment analysis rather than differential analysis because in identifying eGenes, the alternative hypothesis is that the expectation of the maximum absolute correlation from the original expression data is *greater than* (rather than merely different from) the expectation of the maximum absolute correlation from permuted expression data.

In ClipperQTL, the number of replicates under the experimental condition is fixed at one because we only have one set of the original, unpermuted expression data. Therefore, if $B = 1$, then we only need to consider the BC procedure (in enrichment analysis, if the number of replicates under the experimental condition and the number of replicates under the background condition are both one, then the GZ procedure with either maximum or difference contrast score reduces to the BC procedure with difference contrast score); if $B > 1$, then we only need to consider the GZ procedure (in enrichment analysis, the BC procedure is only applicable when the number of replicates under the experimental condition and the number of replicates under the background condition are equal [51]). In other words, B determines the procedure we need to consider.

Therefore, we explore different combinations of B, contrast score, and h (only applicable under the GZ procedure). We find that for data sets with small sample sizes (< 450), no combination works well consistently, but for data sets with large sample sizes (> 450), B between 20 and 100, maximum contrast score, and $h = 1$ works well ($B = 1$ and maximum contrast score works almost as well; details not shown). Therefore, the Clipper variant of ClipperQTL is only recommended for data sets with large sample sizes (> 450; Section 4.2.2). It always uses enrichment analysis, GZ procedure, maximum contrast score, and $h = 1$

(Algorithm 4), and the user is recommended to set B between 20 and 100 (Section 4.2.2).

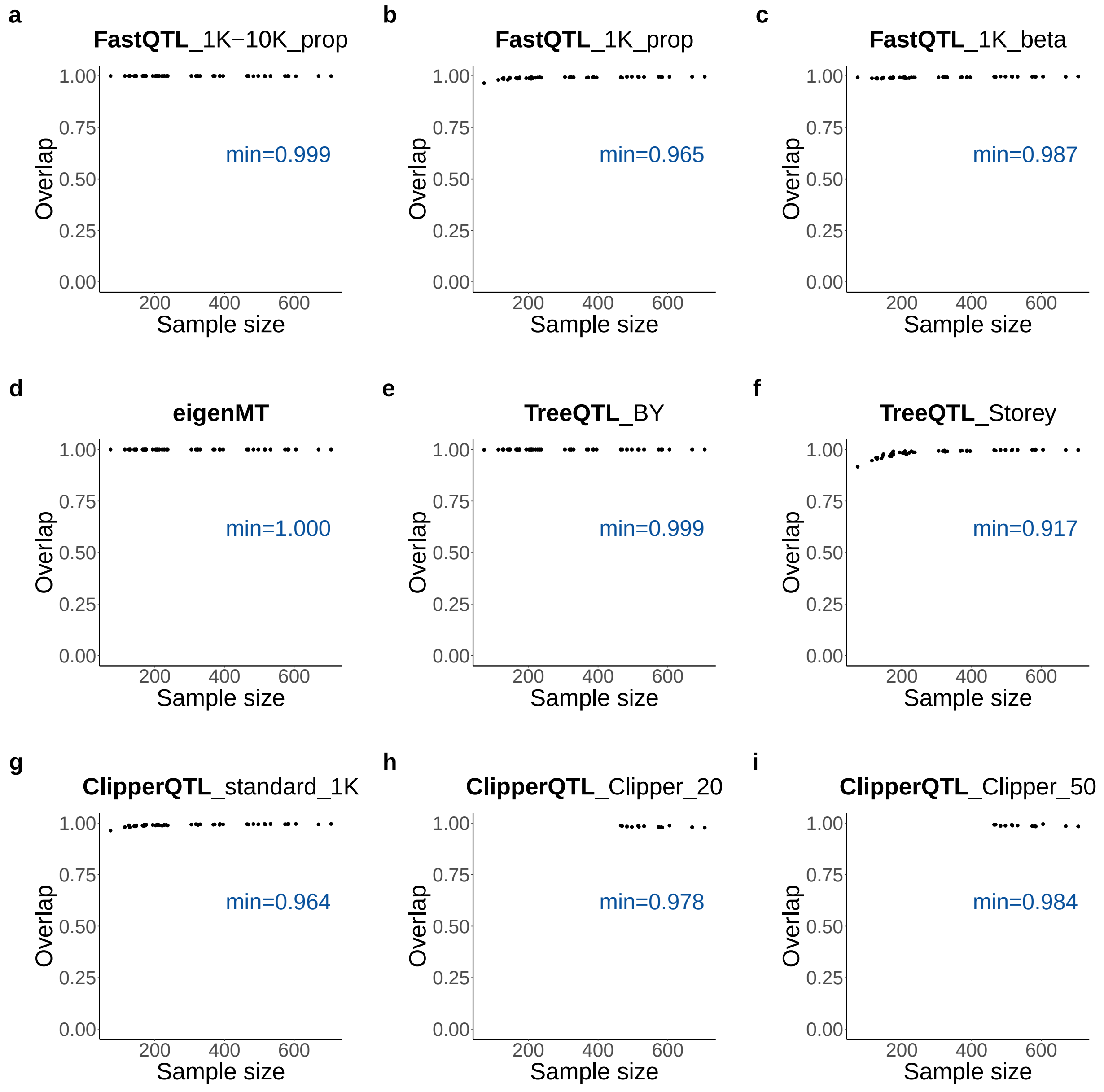

Figure 5.2: Overlap between eGenes identified by various methods and eGenes identified by FastQTL_1K-10K_beta—the default FastQTL method—in GTEx expression data [4] (Table 2.1; see Section 4.1.1 for the analysis details). Each dot corresponds to a tissue. Given two sets, A and B, the overlap is defined as $|A \cap B| / \min(|A|, |B|)$, where $|\cdot|$ denotes the cardinality of a set. That is, the overlap between two sets is defined as the size of the intersection divided by the size of the smaller set. **b**, **c**, **g** The overlap is slightly lower when the sample size is smaller. This can be explained by the fact that power is generally lower when the sample size is smaller [4]. **h**, **i** Only tissues with sample sizes ≥ 465 are shown (Figure 4.1).

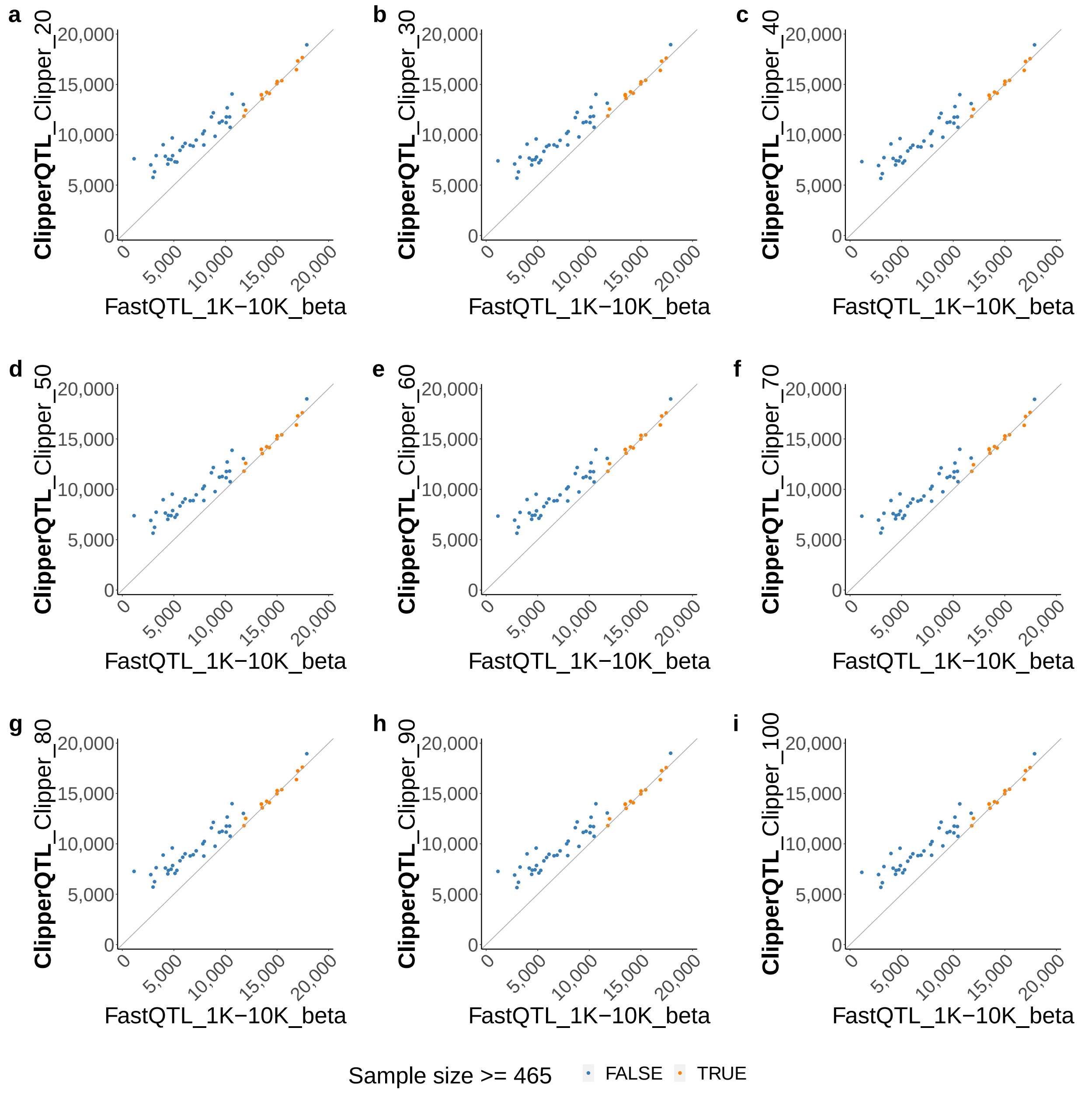

Figure 5.3: Number of eGenes comparison between ClipperQTL_Clipper with different B's and the default FastQTL method based on GTEx expression data [4] (the analysis details are as described in Section 4.1.1). Each dot corresponds to a tissue. The x-axis and y-axis both represent numbers of eGenes identified by different methods. Diagonal lines through the origin are shown to help with visualization. ClipperQTL_Clipper with B between 20 and 100 works well for tissues with large sample sizes (Section 4.2.2). We use 465 as the sample size cutoff because the next largest sample size is 396.

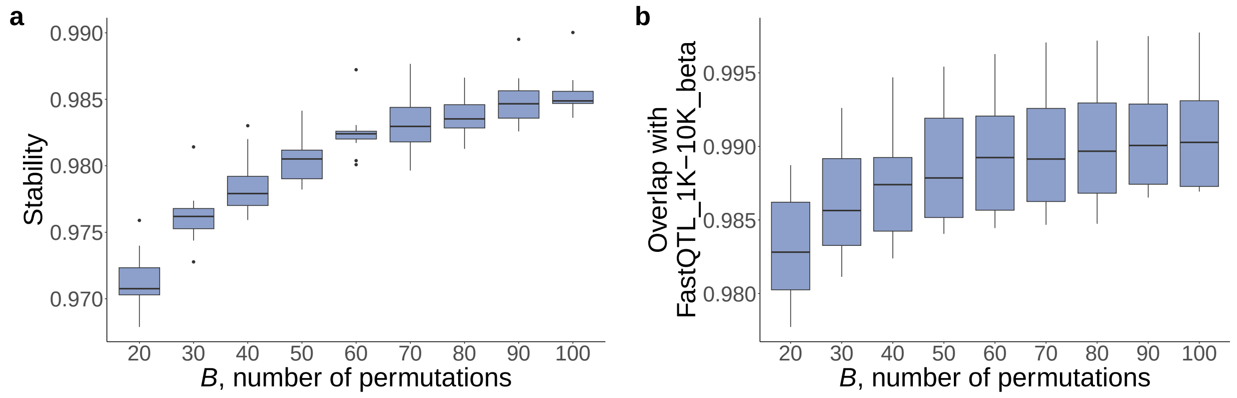

Figure 5.4: Comparison of ClipperQTL_Clipper with different B's. Each box plot contains 13 data points, corresponding to the 13 tissues in GTEx expression data [4] with sample sizes ≥ 465 (the analysis details are as described in Section 4.1.1). As B increases from 20 to 100, the stability of ClipperQTL_Clipper (**a**) and the discovery overlap with the default FastQTL method (**b**) both increase slightly. See Figure 5.2 for our definition of overlap. Stability is a measure of how much the result of a method depends on the random seed; the higher the stability, the less the result varies with respect to the random seed. Specifically, to calculate the stability of a method (e.g., ClipperQTL_Clipper_20), we run the method 10 times with 10 different seeds. We divide the 10 runs into 5 pairs. For each pair, we calculate the overlap between the two sets of identified eGenes. The stability of the method is calculated as the average of the 5 overlaps.

CHAPTER 6

Conclusions

6.1 Hidden variable inference problem

Hidden variable inference is widely practiced as an important step in QTL mapping for improving the power of QTL identification. Popular hidden variable inference methods include SVA, PEER, and HCP. In this work, we show that PCA not only underlies the statistical methodology behind the popular methods (Section 2.1.4) but is also orders of magnitude faster, better-performing, and much easier to interpret and use (Figure 2.1; relatedly, Malik and Michoel [63] have pointed out issues with the optimization algorithm used in PANAMA [64]—a variant of PEER, and the computational efficiency of PCA has been reported in other settings, including genomic selection [65]). Our conclusions are consistent with those from Cuomo et al. [66], who conclude that PCA is superior to alternative hidden variable inference methods for improving the power of single-cell eQTL analysis.

On the simulation front, we compare the runtime and performance of PCA, SVA, PEER, and HCP via two simulation studies (Section 2.1.1). In the first simulation study, we follow the data simulation in Stegle et al. [5], the original PEER publication, while addressing its data analysis and overall design limitations. In the second simulation study, we further address the data simulation limitations of Stegle et al. [5] by simulating the data in a more realistic and comprehensive way. Both simulation studies unanimously show that PCA is faster and better-performing. Further, they show that running PEER *with* the known covariates has no advantage over running PEER *without* the known covariates—in fact, running PEER *with* the known covariates makes PEER significantly slower (Figure 3.6)—and that contrary to claims in Stegle et al. [1, 5], the performance of PEER *does* deteriorate as the number of PEER factors increases (Section 2.1.1). One caveat of our simulation studies, though, is that the genotype and covariates all have linear effects on the gene expression levels (consistent with Stegle et al. [5] and Wang et al. [20]). But since PCA, SVA, PEER, and HCP are all linear methods or assume linearity [15], and so does linear regression, we do not believe our conclusions would change qualitatively if we simulated the data in a nonlinear fashion.

On the real data front, we examine the most recent GTEx eQTL and sQTL data [4] and the $3'$aQTL data prepared by Li et al. [2] from GTEx RNA-seq reads [3]. While the exact data analysis pipelines are different (Table 3.3), these studies all choose PEER as their hidden variable inference method (due to lack of data availability, we do not examine more real data sets). Our analysis shows that PEER, the most popular hidden variable inference method for QTL mapping currently, produces nearly identical results as PCA at best (Section 2.1.3), is at least three orders of magnitude slower than PCA (Figure 3.6), and can be full of pitfalls. Specifically, we show that in certain cases, PEER factors can be highly correlated with each other and thus fail to capture important variance components of the molecular phenotype data, leading to potential loss of power in QTL identification (Section 2.1.2). Further, we show from the perspective of PCA that choosing the number of PEER factors by maximizing the number of discoveries (a common approach used by practitioners) may yield inappropriate choices of K, leading to model overfit and potential loss of power and precision (Section 2.1.5).

Between the two PCA approaches, PCA_direct (running PCA on the fully processed molecular phenotype matrix *directly* and filtering out the known covariates that are captured well by the top PCs afterwards) and PCA_resid (running PCA after regressing out the effects of the known covariates from the molecular phenotype matrix) (Table 2.1; Section 3.4), we recommend PCA_direct because the two approaches perform similarly in our simulation studies and PCA_direct is simpler. In addition, PCA_direct can better hedge against the possibility that the known covariates are not actually important confounders because in PCA_direct, the known covariates do not affect the calculation of the PCs. We also advise the users to make sure to center and scale their data when running PCA unless they are experts and have a good reason not to.

In addition to the benefits discussed so far, using PCA rather than SVA, PEER, or HCP has another conceptual benefit. While SVA, PEER, and HCP are hidden variable inference (i.e., factor discovery) methods, PCA can be interpreted and used as both a *dimension reduction* and a *factor discovery* method. Therefore, PCs of the molecular phenotype data need not be considered inferred covariates; instead, they can be considered a dimension-reduced version of the molecular phenotype data—by including them as covariates, we are controlling for the effect of the overall gene expression profile on the expression level of any individual gene (taking expression phenotypes as an example). With this perspective, including *phenotype* PCs as covariates is analogous to including *genotype* PCs as covariates (which is commonly done to correct for population stratification [3, 4]). This perspective solves the conundrum that inferred covariates such as PEER factors are often difficult to interpret using known technical and biological variables [67].

To help researchers use PCA in their QTL analysis, we provide an R package PCAForQTL, which implements highly interpretable methods for choosing the number of PCs (Algorithms S2 and S3), a graphing function, and more, along with a detailed tutorial. Both resources are freely available at `https://github.com/heatherjzhou/PCAForQTL` [56]. We believe that using PCA rather than SVA, PEER, or HCP will substantially improve and simplify hidden variable inference in QTL mapping as well as increase the transparency and reproducibility of QTL research.

6.2 eGene identification problem

We have shown that ClipperQTL achieves a 500-fold or 50-fold increase in computational efficiency compared to FastQTL (depending on the variant used) without sacrificing power or precision. In contrast, other alternatives to FastQTL such as eigenMT and TreeQTL have lower power than FastQTL.

We propose two main variants of ClipperQTL: the standard variant and the Clipper variant. The standard variant is equivalent to FastQTL with the direct permutation scheme and proportions (Algorithm 5) and is suitable for a wide range of sample sizes. The Clipper variant uses the contrastive strategy developed in Clipper [51] (Algorithm 4) and is only recommended for data sets with large sample sizes (> 450).

Regarding which variant of ClipperQTL should be used when the sample size is large enough (> 450), we believe that if computational efficiency is a priority, then the Clipper variant should be used. However, if the study also contains smaller data sets, then the researcher may choose to use the standard variant on all data sets for consistency.

A possible extension of ClipperQTL lies in trans-eGene identification. Compared to cis-eGenes, trans-eGenes are currently identified in very small numbers [4], possibly due to the lack of power of existing approaches. Since the Clipper variant of ClipperQTL only needs 20 permutations for optimal performance and using only one permutation works almost as well (Section 5.3), there may be potential for ClipperQTL to be adapted for trans-eGene identification.

The R package ClipperQTL is available at `https://github.com/heatherjzhou/ClipperQTL`. Our work demonstrates the potential of the contrastive strategy developed in Clipper and provides a simpler and more efficient way of identifying cis-eGenes.

Bibliography

[1] Oliver Stegle, Leopold Parts, Matias Piipari, John Winn, and Richard Durbin. Using probabilistic estimation of expression residuals (PEER) to obtain increased power and interpretability of gene expression analyses. *Nature Protocols*, 7(3):500–507, 2012.

[2] Lei Li, Kai-Lieh Huang, Yipeng Gao, Ya Cui, Gao Wang, Nathan D. Elrod, Yumei Li, Yiling Elaine Chen, Ping Ji, Fanglue Peng, William K. Russell, Eric J. Wagner, and Wei Li. An atlas of alternative polyadenylation quantitative trait loci contributing to complex trait and disease heritability. *Nature Genetics*, 53(7):994–1005, 2021.

[3] GTEx Consortium. Genetic effects on gene expression across human tissues. *Nature*, 550(7675):204–213, 2017.

[4] GTEx Consortium. The GTEx Consortium atlas of genetic regulatory effects across human tissues. *Science*, 369(6509):1318–1330, 2020.

[5] Oliver Stegle, Leopold Parts, Richard Durbin, and John Winn. A Bayesian framework to account for complex non-genetic factors in gene expression levels greatly increases power in eQTL studies. *PLoS Computational Biology*, 6(5):e1000770, 2010.

[6] Ian T. Jolliffe. *Principal Component Analysis*. Springer, New York, 2nd edition, 2002.

[7] Richard A. Johnson and Dean W. Wichern. *Applied Multivariate Statistical Analysis*. Pearson Prentice Hall, Upper Saddle River, NJ, 6th edition, 2007.

[8] Ian T. Jolliffe and Jorge Cadima. Principal component analysis: A review and recent developments. *Philosophical Transactions of the Royal Society A*, 374(2065), 2016.

[9] Harold Hotelling. Analysis of a complex of statistical variables into principal components. *Journal of Educational Psychology*, 24(6):417–441, 1933.

[10] Karl Pearson. LIII. On lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 2(11): 559–572, 1901.

[11] Michael E. Tipping and Christopher M. Bishop. Probabilistic principal component analysis. *Journal of the Royal Statistical Society: Series B*, 61(3):611–622, 1999.

[12] K. V. Mardia, J. T. Kent, and J. M. Bibby. *Multivariate Analysis*. Academic Press, London, 1979.

[13] Jeffrey T. Leek and John D. Storey. Capturing heterogeneity in gene expression studies by surrogate variable analysis. *PLoS Genetics*, 3(9):e161, 2007.

[14] Jefferey T. Leek and John D. Storey. A general framework for multiple testing dependence. *Proceedings of the National Academy of Sciences*, 105(48):18718–18723, 2008.

[15] Heather J. Zhou. Capturing hidden covariates with linear factor models and other statistical methods in differential gene expression and expression quantitative trait locus studies. *UCLA Electronic Theses and Dissertations*, 2022.

[16] Sara Mostafavi, Alexis Battle, Xiaowei Zhu, Alexander E. Urban, Douglas Levinson, Stephen B. Montgomery, and Daphne Koller. Normalizing RNA-sequencing data by modeling hidden covariates with prior knowledge. *PLoS ONE*, 8(7):e68141, 2013.

[17] Halit Ongen, Alfonso Buil, Andrew Anand Brown, Emmanouil T. Dermitzakis, and Olivier Delaneau. Fast and efficient QTL mapper for thousands of molecular phenotypes. *Bioinformatics*, 32(10):1479–1485, 2016.

[18] T. Mark Beasley, Stephen Erickson, and David B. Allison. Rank-based inverse normal transformations are increasingly used, but are they merited? *Behavior Genetics*, 39(5): 580–595, 2009.

[19] Heather J. Zhou, Lei Li, Yumei Li, Wei Li, and Jingyi Jessica Li. PCA outperforms popular hidden variable inference methods for molecular QTL mapping. *Genome Biology*, 23(1):210, 2022.

[20] Gao Wang, Abhishek Sarkar, Peter Carbonetto, and Matthew Stephens. A simple new approach to variable selection in regression, with application to genetic fine mapping. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 82(5):1273–1300, 2020.

[21] Jacqueline MacArthur, Emily Bowler, Maria Cerezo, Laurent Gil, Peggy Hall, Emma Hastings, Heather Junkins, Aoife McMahon, Annalisa Milano, Joannella Morales, Zoe May Pendlington, Danielle Welter, Tony Burdett, Lucia Hindorff, Paul Flicek, Fiona Cunningham, and Helen Parkinson. The new NHGRI-EBI Catalog of published genome-wide association studies (GWAS Catalog). *Nucleic Acids Research*, 45(D1): D896–D901, 2017.

[22] Annalisa Buniello, Jacqueline A. L. MacArthur, Maria Cerezo, Laura W. Harris, James Hayhurst, Cinzia Malangone, Aoife McMahon, Joannella Morales, Edward Mountjoy, and Elliot Sollis. The NHGRI-EBI GWAS Catalog of published genome-wide association studies, targeted arrays and summary statistics 2019. *Nucleic Acids Research*, 47(D1): D1005–D1012, 2019.

[23] Hongyu Zhao. Roles of statistical modeling in characterizing the genetic basis of human diseases and traits. *Quantitative Biology*, 9(4):371–377, 2021.

[24] Loic Yengo, Julia Sidorenko, Kathryn E. Kemper, Zhili Zheng, Andrew R. Wood, Michael N. Weedon, Timothy M. Frayling, Joel Hirschhorn, Jian Yang, and Peter M. Visscher. Meta-analysis of genome-wide association studies for height and body mass index in 700 000 individuals of European ancestry. *Human Molecular Genetics*, 27(20): 3641–3649, 2018.

[25] Matthew T. Maurano, Richard Humbert, Eric Rynes, Robert E. Thurman, Eric Haugen, Hao Wang, Alex P. Reynolds, Richard Sandstrom, Hongzhu Qu, Jennifer Brody, Anthony Shafer, Fidencio Neri, Kristen Lee, Tanya Kutyavin, Sandra Stehling-Sun, Audra K. Johnson, Theresa K. Canfield, Erika Giste, Morgan Diegel, Daniel Bates, R. Scott Hansen, Shane Neph, Peter J. Sabo, Shelly Heimfeld, Antony Raubitschek, Steven Ziegler, Chris Cotsapas, Nona Sotoodehnia, Ian Glass, Shamil R. Sunyaev, Rajinder Kaul, and John A. Stamatoyannopoulos. Systematic localization of common disease-associated variation in regulatory DNA. *Science*, 337(6099):1190–1195, 2012.

[26] Edward Mountjoy, Ellen M. Schmidt, Miguel Carmona, Jeremy Schwartzentruber, Gareth Peat, Alfredo Miranda, Luca Fumis, James Hayhurst, Annalisa Buniello, Mohd Anisul Karim, Daniel Wright, Andrew Hercules, Eliseo Papa, Eric B. Fauman, Jeffrey C. Barrett, John A. Todd, David Ochoa, Ian Dunham, and Maya Ghoussaini. An open approach to systematically prioritize causal variants and genes at all published human GWAS trait-associated loci. *Nature Genetics*, 53(11):1527–1533, 2021.

[27] Eddie Cano-Gamez and Gosia Trynka. From GWAS to function: Using functional genomics to identify the mechanisms underlying complex diseases. *Frontiers in Genetics*, 11:424, 2020.

[28] Youqiong Ye, Zhao Zhang, Yaoming Liu, Lixia Diao, and Leng Han. A multi-omics perspective of quantitative trait loci in precision medicine. *Trends in Genetics*, 36(5): 318–336, 2020.

[29] Andrey A. Shabalin. Matrix eQTL: Ultra fast eQTL analysis via large matrix operations. *Bioinformatics*, 28(10):1353–1358, 2012.

[30] The Geuvadis Consortium, Tuuli Lappalainen, Michael Sammeth, Marc R. Friedländer, Peter A. C. ‘t Hoen, Jean Monlong, Manuel A. Rivas, Mar Gonzàlez-Porta, Natalja Kurbatova, Thasso Griebel, Pedro G. Ferreira, Matthias Barann, Thomas Wieland, Liliana

Greger, Maarten van Iterson, Jonas Almlöf, Paolo Ribeca, Irina Pulyakhina, Daniela Esser, Thomas Giger, Andrew Tikhonov, Marc Sultan, Gabrielle Bertier, Daniel G. MacArthur, Monkol Lek, Esther Lizano, Henk P. J. Buermans, Ismael Padioleau, Thomas Schwarzmayr, Olof Karlberg, Halit Ongen, Helena Kilpinen, Sergi Beltran, Marta Gut, Katja Kahlem, Vyacheslav Amstislavskiy, Oliver Stegle, Matti Pirinen, Stephen B. Montgomery, Peter Donnelly, Mark I. McCarthy, Paul Flicek, Tim M. Strom, Hans Lehrach, Stefan Schreiber, Ralf Sudbrak, Ángel Carracedo, Stylianos E. Antonarakis, Robert Häsler, Ann-Christine Syvänen, Gert-Jan van Ommen, Alvis Brazma, Thomas Meitinger, Philip Rosenstiel, Roderic Guigó, Ivo G. Gut, Xavier Estivill, and Emmanouil T. Dermitzakis. Transcriptome and genome sequencing uncovers functional variation in humans. *Nature*, 501(7468):506–511, 2013.

[31] Xianjun Dong, Zhixiang Liao, David Gritsch, Yavor Hadzhiev, Yunfei Bai, Joseph J. Locascio, Boris Guennewig, Ganqiang Liu, Cornelis Blauwendraat, Tao Wang, Charles H. Adler, John C. Hedreen, Richard L. M. Faull, Matthew P. Frosch, Peter T. Nelson, Patrizia Rizzu, Antony A. Cooper, Peter Heutink, Thomas G. Beach, John S. Mattick, Ferenc Müller, and Clemens R. Scherzer. Enhancers active in dopamine neurons are a primary link between genetic variation and neuropsychiatric disease. *Nature Neuroscience*, 21(10):1482–1492, 2018.

[32] Christopher E. Gillies, Rosemary Putler, Rajasree Menon, Edgar Otto, Kalyn Yasutake, Viji Nair, Paul Hoover, David Lieb, Shuqiang Li, Sean Eddy, Damian Fermin, Michelle T. McNulty, Nir Hacohen, Krzysztof Kiryluk, Matthias Kretzler, Xiaoquan Wen, Matthew G. Sampson, John Sedor, Katherine Dell, Marleen Schachere, Kevin Lemley, Lauren Whitted, Tarak Srivastava, Connie Haney, Christine Sethna, Kalliopi Grammatikopoulos, Gerald Appel, Michael Toledo, Laurence Greenbaum, Chia-shi Wang, Brian Lee, Sharon Adler, Cynthia Nast, Janine LaPage, Ambarish Athavale, Alicia Neu, Sara Boynton, Fernando Fervenza, Marie Hogan, John C. Lieske, Vladimir Chernitskiy, Frederick Kaskel, Neelja Kumar, Patricia Flynn, Jeffrey Kopp, Eveleyn

Castro-Rubio, Jodi Blake, Howard Trachtman, Olga Zhdanova, Frank Modersitzki, Suzanne Vento, Richard Lafayette, Kshama Mehta, Crystal Gadegbeku, Duncan Johnstone, Daniel Cattran, Michelle Hladunewich, Heather Reich, Paul Ling, Martin Romano, Alessia Fornoni, Laura Barisoni, Carlos Bidot, Matthias Kretzler, Debbie Gipson, Amanda Williams, Renee Pitter, Patrick Nachman, Keisha Gibson, Sandra Grubbs, Anne Froment, Lawrence Holzman, Kevin Meyers, Krishna Kallem, Fumei Cerecino, Kamal Sambandam, Elizabeth Brown, Natalie Johnson, Ashley Jefferson, Sangeeta Hingorani, Kathleen Tuttle, Laura Curtin, S. Dismuke, Ann Cooper, Barry Freedman, Jen Jar Lin, Stefanie Gray, Matthias Kretzler, Larua Barisoni, Crystal Gadegbeku, Brenda Gillespie, Debbie Gipson, Lawrence Holzman, Laura Mariani, Matthew G. Sampson, Peter Song, Johnathan Troost, Jarcy Zee, Emily Herreshoff, Colleen Kincaid, Chrysta Lienczewski, Tina Mainieri, Amanda Williams, Kevin Abbott, Cindy Roy, Tiina Urv, and John Brooks. An eQTL landscape of kidney tissue in human nephrotic syndrome. *The American Journal of Human Genetics*, 103(2):232–244, 2018.

[33] Satria P. Sajuthi, Peter DeFord, Yingchun Li, Nathan D. Jackson, Michael T. Montgomery, Jamie L. Everman, Cydney L. Rios, Elmar Pruesse, James D. Nolin, Elizabeth G. Plender, Michael E. Wechsler, Angel C. Y. Mak, Celeste Eng, Sandra Salazar, Vivian Medina, Eric M. Wohlford, Scott Huntsman, Deborah A. Nickerson, Soren Germer, Michael C. Zody, Gonçalo Abecasis, Hyun Min Kang, Kenneth M. Rice, Rajesh Kumar, Sam Oh, Jose Rodriguez-Santana, Esteban G. Burchard, and Max A. Seibold. Type 2 and interferon inflammation regulate SARS-CoV-2 entry factor expression in the airway epithelium. *Nature Communications*, 11(1):5139, 2020.

[34] Luz D. Orozco, Hsu-Hsin Chen, Christian Cox, Kenneth J. Katschke, Rommel Arceo, Carmina Espiritu, Patrick Caplazi, Sarajane Saturnio Nghiem, Ying-Jiun Chen, Zora Modrusan, Amy Dressen, Leonard D. Goldstein, Christine Clarke, Tushar Bhangale, Brian Yaspan, Marion Jeanne, Michael J. Townsend, Menno van Lookeren Campagne, and Jason A. Hackney. Integration of eQTL and a single-cell atlas in the human eye

identifies causal genes for age-related macular degeneration. *Cell Reports*, 30(4):1246–1259.e6, 2020.

[35] Jing Gong, Shufang Mei, Chunjie Liu, Yu Xiang, Youqiong Ye, Zhao Zhang, Jing Feng, Renyan Liu, Lixia Diao, An-Yuan Guo, Xiaoping Miao, and Leng Han. PancanQTL: Systematic identification of cis-eQTLs and trans-eQTLs in 33 cancer types. *Nucleic Acids Research*, 46(D1):D971–D976, 2018.

[36] Anubha Mahajan, Daniel Taliun, Matthias Thurner, Neil R. Robertson, Jason M. Torres, N. William Rayner, Anthony J. Payne, Valgerdur Steinthorsdottir, Robert A. Scott, Niels Grarup, James P. Cook, Ellen M. Schmidt, Matthias Wuttke, Chloé Sarnowski, Reedik Mägi, Jana Nano, Christian Gieger, Stella Trompet, Cécile Lecoeur, Michael H. Preuss, Bram Peter Prins, Xiuqing Guo, Lawrence F. Bielak, Jennifer E. Below, Donald W. Bowden, John Campbell Chambers, Young Jin Kim, Maggie C. Y. Ng, Lauren E. Petty, Xueling Sim, Weihua Zhang, Amanda J. Bennett, Jette Bork-Jensen, Chad M. Brummett, Mickaël Canouil, Kai-Uwe Ec kardt, Krista Fischer, Sharon L. R. Kardia, Florian Kronenberg, Kristi Läll, Ching-Ti Liu, Adam E. Locke, Jian'an Luan, Ioanna Ntalla, Vibe Nylander, Sebastian Schönherr, Claudia Schurmann, Loïc Yengo, Erwin P. Bottinger, Ivan Brandslund, Cramer Christensen, George Dedoussis, Jose C. Florez, Ian Ford, Oscar H. Franco, Timothy M. Frayling, Vilmantas Giedraitis, Sophie Hackinger, Andrew T. Hattersley, Christian Herder, M. Arfan Ikram, Martin Ingelsson, Marit E. Jørgensen, Torben Jørgensen, Jennifer Kriebel, Johanna Kuusisto, Symen Ligthart, Cecilia M. Lindgren, Allan Linneberg, Valeriya Lyssenko, Vasiliki Mamakou, Thomas Meitinger, Karen L. Mohlke, Andrew D. Morris, Girish Nadkarni, James S. Pankow, Annette Peters, Naveed Sattar, Alena Stančáková, Konstantin Strauch, Kent D. Taylor, Barbara Thorand, Gudmar Thorleifsson, Unnur Thorsteinsdottir, Jaakko Tuomilehto, Daniel R. Witte, Josée Dupuis, Patricia A. Peyser, Eleftheria Zeggini, Ruth J. F. Loos, Philippe Froguel, Erik Ingelsson, Lars Lind, Leif Groop, Markku Laakso, Francis S. Collins, J. Wouter Jukema, Colin N. A. Palmer, Harald Grallert, Andres Metspalu, Ab-

bas Dehghan, Anna Köttgen, Goncalo R. Abecasis, James B. Meigs, Jerome I. Rotter, Jonathan Marchini, Oluf Pedersen, Torben Hansen, Claudia Langenberg, Nicholas J. Wareham, Kari Stefansson, Anna L. Gloyn, Andrew P. Morris, Michael Boehnke, and Mark I. McCarthy. Fine-mapping type 2 diabetes loci to single-variant resolution using high-density imputation and islet-specific epigenome maps. *Nature Genetics*, 50(11): 1505–1513, 2018.

[37] Stephanie Feupe Fotsing, Jonathan Margoliash, Catherine Wang, Shubham Saini, Richard Yanicky, Sharona Shleizer-Burko, Alon Goren, and Melissa Gymrek. The impact of short tandem repeat variation on gene expression. *Nature Genetics*, 51(11): 1652–1659, 2019.

[38] Rebecca L. Walker, Gokul Ramaswami, Christopher Hartl, Nicholas Mancuso, Michael J. Gandal, Luis de la Torre-Ubieta, Bogdan Pasaniuc, Jason L. Stein, and Daniel H. Geschwind. Genetic control of expression and splicing in developing human brain informs disease mechanisms. *Cell*, 179(3):750–771, 2019.

[39] Alexis Battle, Sara Mostafavi, Xiaowei Zhu, James B. Potash, Myrna M. Weissman, Courtney McCormick, Christian D. Haudenschild, Kenneth B. Beckman, Jianxin Shi, Rui Mei, Alexander E. Urban, Stephen B. Montgomery, Douglas F. Levinson, and Daphne Koller. Characterizing the genetic basis of transcriptome diversity through RNA-sequencing of 922 individuals. *Genome Research*, 24(1):14–24, 2014.

[40] James C. Cronk, Anthony J. Filiano, Antoine Louveau, Ioana Marin, Rachel Marsh, Emily Ji, Dylan H. Goldman, Igor Smirnov, Nicholas Geraci, Scott Acton, Christopher C. Overall, and Jonathan Kipnis. Peripherally derived macrophages can engraft the brain independent of irradiation and maintain an identity distinct from microglia. *Journal of Experimental Medicine*, 215(6):1627–1647, 2018.

[41] Jeffrey W. Tyner, Cristina E. Tognon, Daniel Bottomly, Beth Wilmot, Stephen E. Kurtz, Samantha L. Savage, Nicola Long, Anna Reister Schultz, Elie Traer, Melissa

Abel, Anupriya Agarwal, Aurora Blucher, Uma Borate, Jade Bryant, Russell Burke, Amy Carlos, Richie Carpenter, Joseph Carroll, Bill H. Chang, Cody Coblentz, Amanda d'Almeida, Rachel Cook, Alexey Danilov, Kim-Hien T. Dao, Michie Degnin, Deirdre Devine, James Dibb, David K. Edwards, Christopher A. Eide, Isabel English, Jason Glover, Rachel Henson, Hibery Ho, Abdusebur Jemal, Kara Johnson, Ryan Johnson, Brian Junio, Andy Kaempf, Jessica Leonard, Chenwei Lin, Selina Qiuying Liu, Pierrette Lo, Marc M. Loriaux, Samuel Luty, Tara Macey, Jason MacManiman, Jacqueline Martinez, Motomi Mori, Dylan Nelson, Ceilidh Nichols, Jill Peters, Justin Ramsdill, Angela Rofelty, Robert Schuff, Robert Searles, Erik Segerdell, Rebecca L. Smith, Stephen E. Spurgeon, Tyler Sweeney, Aashis Thapa, Corinne Visser, Jake Wagner, Kevin Watanabe-Smith, Kristen Werth, Joelle Wolf, Libbey White, Amy Yates, Haijiao Zhang, Christopher R. Cogle, Robert H. Collins, Denise C. Connolly, Michael W. Deininger, Leylah Drusbosky, Christopher S. Hourigan, Craig T. Jordan, Patricia Kropf, Tara L. Lin, Micaela E. Martinez, Bruno C. Medeiros, Rachel R. Pallapati, Daniel A. Pollyea, Ronan T. Swords, Justin M. Watts, Scott J. Weir, David L. Wiest, Ryan M. Winters, Shannon K. McWeeney, and Brian J. Druker. Functional genomic landscape of acute myeloid leukaemia. *Nature*, 562(7728):526–531, 2018.

[42] Lindsay F. Rizzardi, Peter F. Hickey, Varenka Rodriguez DiBlasi, Rakel Tryggvadóttir, Colin M. Callahan, Adrian Idrizi, Kasper D. Hansen, and Andrew P. Feinberg. Neuronal brain-region-specific DNA methylation and chromatin accessibility are associated with neuropsychiatric trait heritability. *Nature Neuroscience*, 22(2):307–316, 2019.

[43] GTEx Consortium, Taru Tukiainen, Alexandra-Chloé Villani, Angela Yen, Manuel A. Rivas, Jamie L. Marshall, Rahul Satija, Matt Aguirre, Laura Gauthier, Mark Fleharty, Andrew Kirby, Beryl B. Cummings, Stephane E. Castel, Konrad J. Karczewski, François Aguet, Andrea Byrnes, Tuuli Lappalainen, Aviv Regev, Kristin G. Ardlie, Nir Hacohen, and Daniel G. MacArthur. Landscape of X chromosome inactivation across human tissues. *Nature*, 550(7675):244–248, 2017.

[44] Jeffrey T. Leek, W. Evan Johnson, Hilary S. Parker, Andrew E. Jaffe, and John D. Storey. The sva package for removing batch effects and other unwanted variation in high-throughput experiments. *Bioinformatics*, 28(6):882–883, 2012.

[45] Olivier Delaneau, Halit Ongen, Andrew A. Brown, Alexandre Fort, Nikolaos I. Panousis, and Emmanouil T. Dermitzakis. A complete tool set for molecular QTL discovery and analysis. *Nature Communications*, 8(1):15452, 2017.

[46] O. Delaneau, M. Zazhytska, C. Borel, G. Giannuzzi, G. Rey, C. Howald, S. Kumar, H. Ongen, K. Popadin, D. Marbach, G. Ambrosini, D. Bielser, D. Hacker, L. Romano, P. Ribaux, M. Wiederkehr, E. Falconnet, P. Bucher, S. Bergmann, S. E. Antonarakis, A. Reymond, and E. T. Dermitzakis. Chromatin three-dimensional interactions mediate genetic effects on gene expression. *Science*, 364(6439):eaat8266, 2019.

[47] Joe R. Davis, Laure Fresard, David A. Knowles, Mauro Pala, Carlos D. Bustamante, Alexis Battle, and Stephen B. Montgomery. An efficient multiple-testing adjustment for eQTL studies that accounts for linkage disequilibrium between variants. *The American Journal of Human Genetics*, 98(1):216–224, 2016.

[48] C. B. Peterson, M. Bogomolov, Y. Benjamini, and C. Sabatti. TreeQTL: Hierarchical error control for eQTL findings. *Bioinformatics*, 32(16):2556–2558, 2016.

[49] Amaro Taylor-Weiner, François Aguet, Nicholas J. Haradhvala, Sager Gosai, Shankara Anand, Jaegil Kim, Kristin Ardlie, Eliezer M. Van Allen, and Gad Getz. Scaling computational genomics to millions of individuals with GPUs. *Genome Biology*, 20(1):228, 2019.

[50] R. J. Simes. An improved Bonferroni procedure for multiple tests of significance. *Biometrika*, 73(3):751–754, 1986.

[51] Xinzhou Ge, Yiling Elaine Chen, Dongyuan Song, MeiLu McDermott, Kyla Woyshner, Antigoni Manousopoulou, Ning Wang, Wei Li, Leo D. Wang, and Jingyi Jessica

Li. Clipper: P-value-free FDR control on high-throughput data from two conditions. *Genome Biology*, 22(1):288, 2021.

[52] John D. Storey and Robert Tibshirani. Statistical significance for genomewide studies. *Proceedings of the National Academy of Sciences*, 100(16):9440–9445, 2003.

[53] Heather J. Zhou, Xinzhou Ge, and Jingyi Jessica Li. ClipperQTL: Ultrafast and powerful eGene identification method. *bioRxiv*, 2023.

[54] Stephen C. Johnson. Hierarchical clustering schemes. *Psychometrika*, 32(3):241–254, 1967.

[55] Andreas Buja and Nermin Eyuboglu. Remarks on parallel analysis. *Multivariate Behavioral Research*, 27(4):509–540, 1992.

[56] Heather J. Zhou. PCA for hidden variable inference in QTL mapping: An R package and tutorial. *GitHub*, 2022.

[57] Heather J. Zhou, Lei Li, Yumei Li, Wei Li, and Jingyi Jessica Li. PCA outperforms popular hidden variable inference methods for molecular QTL mapping. *Zenodo*, 2022.

[58] Otto Bretscher. *Linear Algebra With Applications*. Pearson Prentice Hall, Upper Saddle River, NJ, 4th edition, 2009.

[59] Qin Qin Huang, Scott C Ritchie, Marta Brozynska, and Michael Inouye. Power, false discovery rate and Winner's Curse in eQTL studies. *Nucleic Acids Research*, 46(22): e133–e133, 2018.

[60] Yoav Benjamini and Yosef Hochberg. Controlling the false discovery rate: A practical and powerful approach to multiple testing. *Journal of the Royal Statistical Society: Series B (Methodological)*, 57(1):289–300, 1995.

[61] Olivier Ledoit and Michael Wolf. A well-conditioned estimator for large-dimensional covariance matrices. *Journal of Multivariate Analysis*, 88(2):365–411, 2004.

[62] Yoav Benjamini and Daniel Yekutieli. The control of the false discovery rate in multiple testing under dependency. *The Annals of Statistics*, 29(4):1165–1188, 2001.

[63] Muhammad Ammar Malik and Tom Michoel. Restricted maximum-likelihood method for learning latent variance components in gene expression data with known and unknown confounders. *G3 Genes—Genomes—Genetics*, 12(2):jkab410, 2022.

[64] Nicolo Fusi, Oliver Stegle, and Neil D Lawrence. Joint modelling of confounding factors and prominent genetic regulators provides increased accuracy in genetical genomics studies. *PLoS Computational Biology*, 8(1):9, 2012.

[65] Caroline Du, Julong Wei, Shibo Wang, and Zhenyu Jia. Genomic selection using principal component regression. *Heredity*, 121(1):12–23, 2018.

[66] Anna S. E. Cuomo, Giordano Alvari, Christina B. Azodi, single-cell eQTLGen consortium, Davis J. McCarthy, and Marc Jan Bonder. Optimizing expression quantitative trait locus mapping workflows for single-cell studies. *Genome Biology*, 22(1):188, 2021.

[67] Pablo E. García-Nieto, Ban Wang, and Hunter B. Fraser. Transcriptome diversity is a systematic source of variation in RNA-sequencing data. *PLoS Computational Biology*, 18(3):e1009939, 2022.