Genetic prediction with ARG-powered linear algebra

Hanbin Lee*[1], Nathaniel S. Pope[2], Jerome Kelleher[3], Gregor Gorjanc[4], and Peter L. Ralph[2]

[1]Department of Statistics, University of Michigan, Ann Arbor, MI, 48109, USA
[2]Institute of Ecology & Evolution, University of Oregon, Eugene, OR, 97405, USA
[3]Big Data Institute, Li Ka Shing Centre for Health Information and Discovery, University of Oxford, Oxford OX3 7LF, UK
[4]The Roslin Institute and Royal (Dick) School of Veterinary Studies, University of Edinburgh, Edinburgh EH25 9RG, UK

November 29, 2025

Abstract

Ancestral recombination graphs (ARGs) are an attractive means for quantitative genetic analysis of complex traits because they encode the realized genetic relatedness between a sample of individuals in the presence of genetic drift, recombination, and mutation. Data structures for efficiently storing ARGs can also be used to rapidly process millions of genomes, and are thus promising for fitting linear mixed models (LMMs) to large phenotype and genome datasets. Here, we study the problems of variance component estimation and prediction of genetic values with ARGs, by describing a generative model of complex traits with additive effects on an ARG, and then developing algorithms that use the ARG to solve these problems efficiently on biobank-scale datasets. We observe nearly linear scaling of runtime with sample size, which is achieved by using the succinct tree sequence representation of the ARG for implicit matrix-vector products, along with modern randomized linear algebra algorithms. We estimate variance components using restricted maximum likelihood (REML), which we find performs substantially better than the Haseman–Elston method. In simulation tests, both variance component estimation and prediction of genetic values (using the best linear unbiased predictor, BLUP) perform nearly as well with inferred ARGs as with true ARGs. We also discuss interpretations of the variance component estimates as mutational variance and additive genetic variance. We provide an implementation of the algorithms as a Python package `tslmm`, which leverages the tree sequence library `tskit`.

1 Introduction

The ancestral recombination graph (ARG) encodes the genetic relationships between a set of recombining genomes (Hudson, 1990; Griffiths and Marjoram, 1997; Wong et al., 2024). Recent methods are capable of reconstructing ARGs from population genomic data (reviewed in Lewanski, Grundler, and Bradburd, 2024; Brandt et al., 2024), providing a window into the history of the genomes from which it is possible to identify

*E-mail: `hblee@umich.edu`

signatures of natural selection as well as demographic events that happened in the past (e.g., Stern et al., 2021; Pope et al., 2023). Some methods scale to hundreds of thousands of samples (Kelleher et al., 2019; Zhang et al., 2023), allowing applications to biobanks and large modern animal or plant breeding datasets. There has long been interest in using ARGs to analyze complex (quantitative) traits (e.g., Larribe, Lessard, and Schork, 2002; Minichiello and Durbin, 2006; see Selle et al., 2021; Link et al., 2023 for more citations). Following the recent advances in ARG inference (Rasmussen et al., 2014; Speidel et al., 2019; Kelleher et al., 2019; Zhang et al., 2023; Deng, Nielsen, and Song, 2024; Gunnarsson et al., 2024), several groups have used inferred ARGs to identify genome regions harboring genetic variation for complex traits (Zhang et al., 2023; Link et al., 2023; Gunnarsson et al., 2024; Zhu et al., 2024; Rebollo et al., 2025).

Many methods for the quantitative genetic analysis of complex traits use linear mixed models (LMMs), which use prior genetic information through a genetic relatedness matrix (GRM). A GRM can be obtained from pedigree, phylogeny, genotypes, or the underlying ARG (reviewed in Lehmann et al., 2025). Work thus far has used the ARG to define a GRM in terms of the total amount of shared ancestral material ("area of shared edges", as described later), and so is the expected value of the genotype GRM (VanRaden, 2008; Speed and Balding, 2015) given the ARG structure and the infinite-sites mutation model. This has been referred to as a "ARG-GRM", "eGRM", or "branch GRM". Since there are in principle many ways to obtain a GRM from an ARG, in this paper we use the term "branch GRM" (Lehmann et al., 2025).

Recent developments show promising results for the use of ARGs in quantitative genetics. Zhang et al. (2023) proposed a Monte-Carlo algorithm for estimating the branch GRM and used it to estimate heritability and variant associations. They showed that the branch GRM can improve heritability estimation and, by using simulated variants on the ARG, complement traditional imputation approaches to detect signals from ungenotyped variants. Link et al. (2023) summarized the aggregate signal of variants within a local window by computing the branch GRM from local trees spanning the window. When applied to a Native Hawaiian cohort, the method successfully found signals that were overlooked by conventional GWAS due to low variant density and lack of insufficient imputation panels. Rebollo et al. (2025) compared the use of pedigree, genotype, and branch GRMs for prediction of genetic values in a rice dataset with two sub-species, and found that using the branch GRM outperforms other GRMs. Rebollo et al. (2025) also backsolved the predicted genetic values to variant associations and demonstrated how these combine into haplotype values in local trees.

Zhang et al. (2023), Link et al. (2023), and Rebollo et al. (2025) used ARGs to compute GRM for downstream analysis, but did not take full advantage of the computational potential of inferred ARGs. They all used existing software (e.g., GCTA (Yang et al., 2011) or ASReml (Butler et al., 2023)) and substituted the conventional genotype GRM with their branch GRM version to fit the LMM. This approach is computationally challenging: with N individuals storing a GRM scales with N^2 while many downstream computations with a GRM scale with N^3. Another challenge in computing the GRM is the large number of distinct local trees as one moves along the genome. However, the ARG itself can be seen as a sparse representation of both the local trees and the genotypes themselves (Ané and Sanderson, 2005; Wong et al., 2024), and can be used to compute a broad range of statistics of both, orders of magnitude faster than variant-by-variant or tree-by-tree approaches (Ralph, Thornton, and Kelleher, 2020; Zhu et al., 2024; Lehmann et al., 2025).

Zhu et al. (2024) was the first paper to utilize this computational potential of the ARG to fit an LMM using the branch GRM. They devised a randomized Haseman–Elston (HE) method, called ARG-RHE, that combines randomized linear algebra with an ARG matrix-vector multiplication algorithm that recycles information shared by adjacent local trees to minimize redundant computations. Their algorithm allows estimating variance components with biobank-scale datasets, which is very difficult with traditional methods. However, room remains for improvement: their underlying algorithms compute quantities based on branch lengths of local trees using a Monte-Carlo approximation with randomly generated mutations, and thus incur a tradeoff

between accuracy and computational resources. Furthermore, the HE method, as a moment-based estimator (Haseman and Elston, 1972), has lower statistical precision compared to the restricted maximum likelihood (REML), which utilizes the full likelihood (Patterson and Thompson, 1971).

In this paper, we study the problem of *genetic prediction*: predicting the genetic values of a certain set of individuals based on the phenotypic values of other individuals and some knowledge about relatedness between them (Henderson, 1975; Lynch and Walsh, 1998; Wray et al., 2019; Mrode and Pocrnic, 2023). This shares many features with the problem of trait mapping (e.g., Strandén and Garrick, 2009; Fan, Mancuso, and Chiang, 2022; Zhang et al., 2023), and many approaches to both use the same linear mixed model:

$$\mathbf{y} = \mathbf{X}\mathbf{b} + \mathbf{g} + \boldsymbol{\varepsilon}, \tag{1}$$

where $\mathbf{y}$ is the vector of phenotypic values (observed and unobserved), $\mathbf{X}$ is a matrix of covariates and $\mathbf{b}$ are the corresponding coefficients, $\mathbf{g}$ is a vector of the genetic values, and $\boldsymbol{\varepsilon}$ is a vector of uncorrelated, Gaussian, "environmental" deviations. Prediction can be done by finding the conditional mean of the unobserved entries of $\mathbf{g}$ given the observed entries. Information about relatedness enters by assuming that the genetic values are similar for closely related individuals: with the covariance between $\mathbf{g}_i$ and $\mathbf{g}_j$ proportional to the "relatedness" between individuals i and j. A Gaussian assumption reduces the main problem – estimating $\mathbf{b}$ and $\mathbf{g}$ given the observed data – to linear algebra involving $\mathbf{X}$ and the relatedness matrix. For instance, if $\mathbf{R}$ is the relatedness matrix and $\mathbf{I}$ is the identity, then the covariance matrix of $\mathbf{y}$ is $\tau^2\mathbf{R}+\sigma^2_{\boldsymbol{\varepsilon}}\mathbf{I}$, where τ^2 and $\sigma^2_{\boldsymbol{\varepsilon}}$ are unknown values controlling the relative contributions of genetics and unmodeled noise, respectively (and thus, heritability). The most challenging step is estimation of the *variance components* τ^2 and $\sigma^2_{\boldsymbol{\varepsilon}}$. Given these, it is fairly straightforward to estimate $\mathbf{b}$ and $\mathbf{g}$.

Oversimplifying somewhat, methods differ in the computational algorithms and the choice of relatedness; here we use the ARG to both provide the measure of relatedness and to speed up computation. To this end, we first describe in some detail the generative model of complex traits on an ARG, which leads to the ARG-based linear mixed model (ARG-LMM) (Zhang et al., 2023). Next, we describe how to fit the ARG-LMM to very large datasets, obtaining estimates of variance components and predicted genetic values ("best linear unbiased predictions", or BLUPs). Computation is a major obstacle for large datasets, so here we focus on using the ARG and randomized linear algebra for efficient inference. The methods are implemented in the open-source Python package `tslmm` available at `https://github.com/hanbin973/tslmm`, which builds on the tree sequence `tskit` library, to take advantage of the project's extensive code base, documentation, unit testing, and stable API.

2 The ARG linear mixed model (ARG-LMM)

The linear mixed model framework applied to ARGs was first presented in Zhang et al. (2023) where they assume that $\mathrm{Var}\,(\mathbf{g})$, the covariance matrix of the genetic value $\mathbf{g}$ in equation (1), is the branch GRM. By explicitly expressing the genetic value in terms of ARG components, beginning from the standard genotype-based representation $\mathbf{g} = \mathbf{G}\boldsymbol{\beta}$, we demonstrate that this assumption follows from first-principles. Suppose that for the ith individual we have a phenotypic value $\mathbf{y}_i$, a set of covariates $\mathbf{X}_{i1}, \ldots, \mathbf{X}_{iK}$, and genotypes $\mathbf{G}_{i1}, \ldots \mathbf{G}_{iP}$. The equation that is commonly used to define the linear mixed model assumes that the phenotypic value of individual i, is equal to the sum of fixed effects, $\sum_j \mathbf{X}_{ij}\mathbf{b}_j$, its genetic value, $\mathbf{g}_i = \sum_p \mathbf{G}_{ip}\boldsymbol{\beta}_p$, and its environmental/non-genetic deviation $\boldsymbol{\varepsilon}_i$; in matrix form this is:

$$\mathbf{y} = \mathbf{X}\mathbf{b} + \mathbf{G}\boldsymbol{\beta} + \boldsymbol{\varepsilon}. \tag{2}$$

Here $\mathbf{b}$ and $\boldsymbol{\beta}$ are coefficients for the non-genetic covariates and genotypes, respectively. However, the features of the sample ARG are not visible in this equation. Hence, we seek an alternative expression of equation (2) that explicitly contains ARG-related terms.

We work with the "infinitesimal" model of phenotype evolution on the ARG, that is used implicitly or explicitly in a variety of papers; see Lehmann et al. (2025) for a recent review. The model can be thought of as the extension to a recombining genome of the "Brownian motion" model of phenotype evolution on phylogenies used by Felsenstein (1973) and Thompson (1975). The model can also be thought of as the specialization of the classical infinitesimal model (Fisher, 1918; Barton, Etheridge, and Véber, 2017) to the situation where we know not only the pedigree of individuals but also which segments of genome they have inherited from parental genomes within the pedigree or from ancestral genomes beyond the pedigree. This model assumes that the net effects of any mutations on a phenotype are additive, and so the genetic covariance in phenotypic values between any pair of individuals is proportional to the total amount of shared ancestral opportunity for mutation.

Here, we propose an explicit phenotype model that induces the branch GRM. To make the model concrete, an ARG $\mathcal{T}$ is defined using collections of nodes $\mathcal{N}$ and edges $\mathcal{E}$. Each node n represents a (possibly ancestral) chromosome, and so has a (birth) time t_n, measured in time ago (i.e., time before the present). A special set of nodes are the *samples*, for which we have complete genome information. Each edge represents a segment of genome spanning $[a_e, b_e)$ that was inherited by a child node c_e from a parent node p_e. (The "child" and "parent" terms refer to relationships in the graph, and so these may be separated by more than one generation.) Along the segment $[a_e, b_e)$, the subtree descending from c_e may change due to further recombinations. Suppose that we split the genomic segment for each edge into disjoint pieces, which we call *sub-edges*, so that there are no recombinations in the ARG below each sub-edge. An example is illustrated in Figure 1. The induced graph structure on sub-edges is a *multitree* (Furnas and Zacks, 1994), which has a number of useful properties (Salehi Nowbandegani et al., 2023; DeHaas, Pan, and Wei, 2024). For instance, values cached at nodes are updated once per sub-edge in the algorithms in Ralph, Thornton, and Kelleher (2020) and Zhu et al. (2024).

With the above assumptions and taking an infinite-sites mutation model (Kimura, 1969; Watterson, 1975), any mutations that occur along a sub-edge e are inherited by all nodes below the sub-edge. Subsequently, the sample nodes below a sub-edge remain constant along its whole span. Thus, we can define the "sub-edge dosage" design matrix $\mathbf{Z}$, with rows indexed by individuals and columns indexed by sub-edges, so that $\mathbf{Z}_{ie}$ is equal to the number of haplotypes the individual i inherited from sub-edge e (for diploids, $\mathbf{Z}_{ie} \in \{0, 1, 2\}$). This allows us to express the genetic value $\mathbf{g} = \mathbf{G}\boldsymbol{\beta}$ as a linear function of sub-edge dosages $\mathbf{Z} \in \mathbb{R}^{N \times E}$ and sub-edge effects $\mathbf{u} \in \mathbb{R}^E$, which turns equation (2) into

$$\mathbf{y} = \mathbf{X}\mathbf{b} + \mathbf{Z}\mathbf{u} + \boldsymbol{\varepsilon}. \tag{3}$$

Here the sub-edge effects $\mathbf{u}$ represent the net effects of any mutations that occur on each sub-edge; further details of this conversion can be found in Appendix A. We assume that $\boldsymbol{\varepsilon}$ are independent, have mean 0 and variance $\sigma^2_{\boldsymbol{\varepsilon}}$. What about $\mathbf{u}$? The total opportunity for mutation of an sub-edge is equal to the per base-pair mutation rate multiplied by its "area", $A_e = (b_e - a_e)(t_{p_e} - t_{c_e})$, which is its genomic span multiplied by its time length. So, an "infinitesimal" model would assume that each $\mathbf{u}_e$ have mean zero and variance $\tau^2 A_e$; see Appendix A for more careful discussion of this point. This implies that the covariance matrix of $\mathbf{Z}\mathbf{u}$ is $\tau^2\mathbf{B} = \tau^2\mathbf{Z}^T\boldsymbol{\Sigma}_A\mathbf{Z}$, where $\boldsymbol{\Sigma}_A$ is the diagonal matrix with areas A_e on the diagonal. $\mathbf{B}$ is the branch GRM using the definition in Lehmann et al. (2025). τ^2 is the variance of mutational effects $\mathbf{u}$ per generation and base-pair.

In summary, the generative model assumes that:

1. genetic values are additive on the ARG;
2. environmental effects are independent with variance σ_ε^2;
3. and the net effects of each sub-edge are independent,
4. have mean zero,
5. and variance proportional to sub-edge area,

An implication of this model is the "ARG linear mixed model" (ARG-LMM): let $\mathbf{Z}$ be the individual/sub-edge design matrix defined in the text, and $\mathbf{\Sigma}_A$ the diagonal matrix with the areas of each sub-edge. Then the covariance of the observed phenotypic values $\mathbf{y} = \mathbf{Xb} + \mathbf{Zu} + \varepsilon$ is:

$$\tau^2 \mathbf{Z}^T \mathbf{\Sigma}_A \mathbf{Z} + \sigma_\varepsilon^2 \mathbf{I},$$

and the covariance of the genetic values $\mathbf{g} = \mathbf{Zu}$ is $\tau^2 \mathbf{B}$, where τ^2 is the variance of mutational effects per generation, per base-pair, and $\mathbf{B}$ is the branch GRM,

$$\mathbf{B} = \mathbf{Z}^T \mathbf{\Sigma}_A \mathbf{Z}.$$

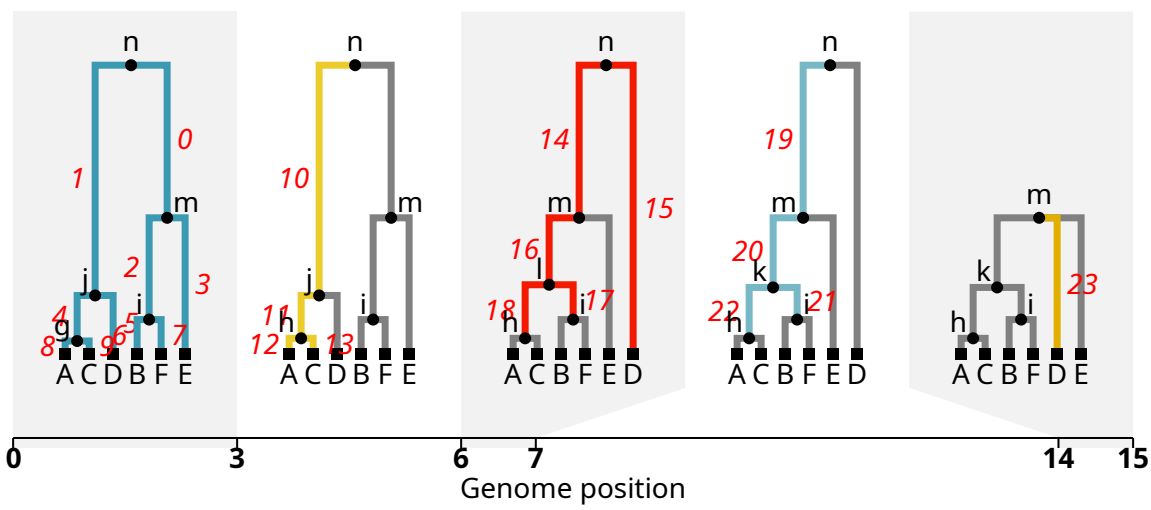

Figure 1: A short tree sequence with sub-edges labeled: moving left to right, each new sub-edge is colored and given a label; sub-edges that remain the same from the previous tree are grey. There are 23 total sub-edges but fewer edges: for instance, the edge from (parent) n to (child) m is split into three sub-edges, labeled 0, 14, and 19.

2.1 Assumptions

An advantage to explicitly formulating the generative model is that it makes the assumptions visible, and suggests ways forward if the assumptions seem unlikely to produce good results. Note, however, that this is *a* generative model that leads to the LMM that we use below (and is used elsewhere; Fan, Mancuso, and

Chiang, 2022; Zhang et al., 2023; Rebollo et al., 2025); other assumptions may arrive at the same model. Here we discuss issues conceptually; see Appendix A for a more rigorous formulation.

While equation (3) implies an additive model without epistasis or dominance, the model can also be thought of as a first-order approximation in the presence of such non-additive effects. Independence of sub-edges is violated when there are multiple causal mutations on related sub-edges, that is one sub-edge is ancestral to another, at the same site. For instance, if the same allele appears independently on different sub-edges, then both alleles are assumed to have the same effect under the additive model. As quantified in Appendix A, the correlations thus induced are small.

The assumption that effects have mean zero could easily be violated. For instance, we expect most mutations in coding regions to be deleterious, so mutations in DNA repair pathways might have a mean positive effect on cancer risk. However, this assumption turns out to be relatively unimportant. As discussed in Appendix A, mean effects are absorbed into the intercept that is typically included in $\mathbf{X}$ as long as all individuals are taken at roughly the same time.

The assumption that variance is proportional to area is perhaps the strongest. One way to justify it is as follows: suppose that mutations occur at rate μ per base-pair, and the effects of these mutations are additive, with mean zero and variance σ^2. If mutations cannot obscure the effect of each other (i.e., a continuum-of-alleles model, as in Kimura and Crow, 1964), then the variance of the total effect of the mutations on a given sub-edge e is $\tau^2 A_e$ with $\tau^2 = \mu\sigma^2$. (If furthermore μ is large and σ is small, then by the central limit theorem, the net effect of those mutations is Gaussian, as in Barton, Etheridge, and Véber, 2017.) This calculation takes the point of view that (a) the net effect of any mutations occurring along an sub-edge is unknown (and random); but also (b) the mutations themselves are unobserved. The latter point might seem odd, because we observe genotypes more directly than ARGs – shouldn't the variance of the effects of an sub-edge be proportional to the number of mutations actually on it, rather than the *potential* for mutations? Perhaps, but we find the point of view useful in a few ways. First, it is unlikely that all causal alleles are genotyped, or that all genotyped polymorphisms are causal, and so estimates of the branch GRM have the potential to better reflect a shared genetic basis for the phenotype. Indeed, Zhang et al. (2023) and Link et al. (2023) identified putative ungenotyped causal alleles following this point of view. Second, the framework makes it natural to distinguish properties of the underlying generative process (such as the mutational variance) from descriptors of the studied samples (such as the additive genetic variance). Finally, the methods we describe can be easily adapted by replacing sub-edge area with the number of mutations on the sub-edge.

A further implication of the last assumption is that causal mutations fall uniformly along the genome. In practice, we might have a prior idea of where these might fall, thus leading to a heterogeneous rate. This could be accounted for by changing the genomic coordinate system to reflect the opportunity for causal mutations. Similarly, if the phenotype is under selection, then it is commonly assumed that causal mutations are unlikely to be common. This might be accounted for by rescaling time, so that causal mutations occur uniformly along the rescaled time axis. These are discussed in more detail in Appendix A.

3 Efficient GRM-vector multiplication, and ARG-LMM simulation

The key to computational efficiency in subsequent sections is the ability to use the tree sequence as, effectively, a sparse matrix representation with which we can efficiently perform GRM-vector products. In Lehmann et al. (2025), we showed that this computation amounts to "adding up values down an ARG". To see why this is, consider the problem of multiplying the GRM by $\mathbf{1}_N$, the constant vector of 1's of length N. The (uncentered) GRM is $\mathbf{Z}^T\mathbf{\Sigma}_A\mathbf{Z}$, so we want to compute $\mathbf{v} = \mathbf{Z}^T\mathbf{\Sigma}_A\mathbf{Z}\mathbf{1}_N$. Since $\mathbf{Z}_{ei} = 1$ if sample i is below sub-edge e, and $[\mathbf{\Sigma}_A]_{ee}$ is the area of sub-edge e, $\mathbf{v}$ is obtained by assigning to each sub-edge a weight which is the

product of the number of samples below the sub-edge multiplied by its area, and then assigning to $\mathbf{v}_i$ the sum of weights of all sub-edges above sample i. However, a naïve tree-by-tree implementation of this description would not take advantage of the substantial structure shared across many trees.

"Adding up values down an ARG" is precisely how the generative model of the previous section works: the genetic value of a sample is equal to the sum of the effects of all sub-edges ancestral to the sample, So, it turns out that an efficient algorithm for simulation from the generative model has the same shape as the matrix-vector product algorithm. We describe an algorithm to do just this in Appendix C, which we find useful for clarifying the dependency structures within the ARG, and why using the full ARG structure is substantially more efficient than tree-by-tree methods.

Although sub-edges are central here, they are not explicitly formed in the algorithms we present in this paper, because creating and storing them can incur a significant computational burden. Instead, we work with the original edges stored in the tree sequences to implement what is algorithmically equivalent to operations on sub-edges.

4 Variance component estimation

Now, we turn to estimation of the variance components, which we do using a modified version the average information restricted maximum likelihood (AI-REML) algorithm. This section derives the REML algorithm as applied to the ARG-LMM model, and describes the computational methods we use to apply it to large-scale problems in `tslmm`.

As a model of phenotypes $\mathbf{y}$, ARG-LMM is a multivariate Gaussian distribution. (Although the derivation is based on a Gaussian model, the resulting REML estimator is robust to distributional error provided the GRM is correct; see **Discussion**.) Hence, we'd like to obtain estimates of τ^2 and $\sigma_{\boldsymbol{\varepsilon}}^2$ in the ARG-LMM model, which is, concisely:

$$\mathbf{y} \mid \mathbf{X}, \mathbf{b}, \tau^2, \sigma_{\boldsymbol{\varepsilon}}^2 \sim \text{Normal}(\mathbf{X}\mathbf{b}, \mathbf{V}) \quad \text{where} \quad \mathbf{V} = \tau^2\mathbf{B} + \sigma_{\boldsymbol{\varepsilon}}^2\mathbf{I}, \tag{4}$$

where $\mathbf{y}$ is the vector of phenotypic values, $\mathbf{B} = \mathbf{Z}\boldsymbol{\Sigma}_A\mathbf{Z}^T$ as before, $\mathbf{X}$ is an $N \times K$ matrix of covariates, N is the sample size (the length of $\mathbf{y}$), K is the number of covariates, and $\mathbf{b}$ is the vector of coefficients. Rather than directly finding the maximum likelihood estimates using the whole model, it turns out to be advantageous to instead find τ^2 and $\sigma_{\boldsymbol{\varepsilon}}^2$ that maximize the likelihood of the portion of $\mathbf{y}$ that is orthogonal to the columns of $\mathbf{X}$. This leads to the restricted maximum likelihood (REML) objective (Patterson and Thompson, 1971), which is, up to a constant factor:

$$\ell(\mathbf{y}, \mathbf{X}; \tau^2, \sigma_{\boldsymbol{\varepsilon}}^2) = (N - K)\log(2\pi) + \log\det(\mathbf{V}) + \log\det(\mathbf{X}^T\mathbf{V}^{-1}\mathbf{X}) + \mathbf{y}^T\mathbf{P}\mathbf{y}, \tag{5}$$

where $\mathbf{P} = \mathbf{V}^{-1} - \mathbf{V}^{-1}\mathbf{X}(\mathbf{X}^T\mathbf{V}^{-1}\mathbf{X})^{-1}\mathbf{X}^T\mathbf{V}^{-1}$. If certain assumptions are met, REML is unbiased and statistically the most efficient estimator of variance components (Patterson and Thompson, 1971; Harville, 1977a; Westfall, 1987; Zhou, 2017).

So, our goal is to find τ^2 and $\sigma_{\boldsymbol{\varepsilon}}^2$ that maximize (5). To explain how this is done, we turn to general-purpose optimization theory. Consider an unconstrained optimization problem for a twice continuously differentiable function $f : \mathbb{R}^p \to \mathbb{R}$, finding $\boldsymbol{\theta}^* = \operatorname{argmin}_{\boldsymbol{\theta}\in\mathbb{R}^p} f(\boldsymbol{\theta})$. (We will have $\theta = (\tau^2, \sigma_{\boldsymbol{\varepsilon}}^2)$, and so $f(\tau^2, \sigma_{\boldsymbol{\varepsilon}}^2) = \ell(\mathbf{y}, \mathbf{X}; \tau^2, \sigma_{\boldsymbol{\varepsilon}}^2)$.) The Newton-Raphson method seeks to find an approximate solution by iterating:

$$\boldsymbol{\theta}_{n+1} = \boldsymbol{\theta}_n - \left[\nabla_{\boldsymbol{\theta}}^2 f(\boldsymbol{\theta}_n)\right]^{-1} \cdot \nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}_n), \tag{6}$$

where $\nabla_{\boldsymbol{\theta}} f$ is the gradient of f and $\nabla^2_{\boldsymbol{\theta}} f$ is the Hessian.

Fisher scoring is a variant of the Newton-Raphson method when the optimization target includes realized random variables. Suppose $\mathbf{x}$ is a random variable that is generated from a distribution parameterized by $\boldsymbol{\theta}$. The optimization target (e.g., the negative log-likelihood) $\ell(x, \theta)$ is a function of both $\boldsymbol{\theta}$ and x where x is a realization of $\mathbf{x}$, and so we now want to find $\boldsymbol{\theta}^* = \operatorname{argmin}_{\boldsymbol{\theta} \in \mathbb{R}^p} \ell(x, \boldsymbol{\theta})$. The corresponding Newton-Raphson update is:

$$\boldsymbol{\theta}_{n+1} = \boldsymbol{\theta}_n - \left[\nabla^2_{\boldsymbol{\theta}} \ell(x, \boldsymbol{\theta}_n)\right]^{-1} \nabla_{\boldsymbol{\theta}} \ell(x, \boldsymbol{\theta}_n). \tag{7}$$

Fisher scoring replaces $\nabla^2_{\boldsymbol{\theta}} \ell(x, \boldsymbol{\theta}_n)$ by its expectation:

$$\boldsymbol{\theta}_{n+1} = \boldsymbol{\theta}_n - \mathbb{E}_{\boldsymbol{\theta}_n} \left[\nabla^2_{\boldsymbol{\theta}} \ell(\mathbf{x}, \boldsymbol{\theta}_n)\right]^{-1} \nabla_{\boldsymbol{\theta}} \ell(x, \boldsymbol{\theta}_n), \tag{8}$$

where the random variable $\mathbf{x}$ sampled at $\boldsymbol{\theta} = \boldsymbol{\theta}_n$ replaces the realized observation x. The expectation $\mathbb{E}_{\boldsymbol{\theta}} \left[\nabla^2_{\boldsymbol{\theta}} \ell(\mathbf{x}, \boldsymbol{\theta})\right]$ is called the *Fisher information*.

Average information is the average of these two quantities:

$$\mathrm{AI}(x, \theta) = \frac{1}{2} \left(\nabla^2_{\boldsymbol{\theta}} \ell(x, \boldsymbol{\theta}) + \mathbb{E}_{\boldsymbol{\theta}} \left[\nabla^2_{\boldsymbol{\theta}} \ell(\mathbf{x}, \boldsymbol{\theta})\right]\right), \tag{9}$$

and the update in iteration is:

$$\boldsymbol{\theta}_{n+1} = \boldsymbol{\theta}_n - \mathrm{AI}(x, \boldsymbol{\theta}_n)^{-1} \nabla_{\boldsymbol{\theta}} \ell(x, \boldsymbol{\theta}_n). \tag{10}$$

All three methods – Newton–Raphson, Fisher scoring, and average information – are second-order optimization methods. Roughly speaking, second-order methods locally approximate the optimization target as a quadratic function by utilizing curvature information encoded in the second derivative (the Hessian). While Newton-Raphson uses the Hessian directly, Fisher scoring uses the Hessian's average over the data distribution at the current step, and average information uses the average of the two. As quadratic approximation is more accurate than linear approximation used by first-order methods, second-order methods generally require fewer iterations and converge more tightly than first-order methods, although quadratic methods are often computationally more expensive than first-order methods because computing the Hessian or related quantities can be very demanding (Boyd and Vandenberghe, 2004). To this end, we develop an efficient average information REML (AI-REML) routine for ARG-LMM with a minimal additional overhead beyond a stochastic estimate of the gradient. First, we derive expressions for the AI-REML updates (which have useful features not found in the general case; Gilmour, Thompson, and Cullis, 1995), and then we discuss how to compute these efficiently.

4.1 Average information residual maximum likelihood (AI-REML)

To find the average information update (equation (10)), we need the first and second derivatives of the REML objective (equation (5)) with respect to τ^2 and $\sigma^2_{\boldsymbol{\varepsilon}}$. (Readers not interested in the computations may skip to the punchline at the end of this section.) Since for any invertible-matrix-valued function $A(x)$, $\partial_x \log \det A(x) =$

$\mathrm{tr}(A(x)^{-1}\partial_x A(x))$, then after a little algebra using the cyclic property of the trace, the first derivatives are:

$$\begin{aligned}
\frac{\partial \ell}{\partial \sigma_{\boldsymbol{\varepsilon}}^2} &= \mathrm{tr}(\mathbf{P}(\nabla_{\sigma_{\boldsymbol{\varepsilon}}^2}\mathbf{V})) - \mathbf{y}^T\mathbf{P}(\nabla_{\sigma_{\boldsymbol{\varepsilon}}^2}\mathbf{V})\mathbf{P}\mathbf{y} \\
&= \mathrm{tr}(\mathbf{P}) - \mathbf{y}^T\mathbf{P}\mathbf{P}\mathbf{y} \\
\frac{\partial \ell}{\partial \tau^2} &= \mathrm{tr}(\mathbf{P}(\nabla_{\tau^2}\mathbf{V})) - \mathbf{y}^T\mathbf{P}(\nabla_{\tau^2}\mathbf{V})\mathbf{P}\mathbf{y} \\
&= \mathrm{tr}(\mathbf{P}\mathbf{B}) - \mathbf{y}^T\mathbf{P}\mathbf{B}\mathbf{P}\mathbf{y},
\end{aligned} \tag{11}$$

because $\nabla_{\sigma_{\boldsymbol{\varepsilon}}^2}\mathbf{V} = \mathbf{I}$ and $\nabla_{\tau^2}\mathbf{V} = \mathbf{B}$. Similarly, the second derivatives are:

$$\begin{aligned}
\frac{\partial \ell}{\partial \sigma_{\boldsymbol{\varepsilon}}^2 \partial \sigma_{\boldsymbol{\varepsilon}}^2} &= -2\mathbf{y}^T\mathbf{P}(\nabla_{\sigma_{\boldsymbol{\varepsilon}}^2}\mathbf{V})\mathbf{P}(\nabla_{\sigma_{\boldsymbol{\varepsilon}}^2}\mathbf{V})\mathbf{P}\mathbf{y} + \mathrm{tr}(\mathbf{P}(\nabla_{\sigma_{\boldsymbol{\varepsilon}}^2}\mathbf{V})\mathbf{P}(\nabla_{\sigma_{\boldsymbol{\varepsilon}}^2}\mathbf{V})) \\
\frac{\partial \ell}{\partial \sigma_{\boldsymbol{\varepsilon}}^2 \partial \tau^2} &= -2\mathbf{y}^T\mathbf{P}(\nabla_{\sigma_{\boldsymbol{\varepsilon}}^2}\mathbf{V})\mathbf{P}(\nabla_{\tau^2}\mathbf{V})\mathbf{P}\mathbf{y} + \mathrm{tr}(\mathbf{P}(\nabla_{\sigma_{\boldsymbol{\varepsilon}}^2}\mathbf{V})\mathbf{P}(\nabla_{\tau^2}\mathbf{V})) \\
\frac{\partial \ell}{\partial \tau^2 \partial \tau^2} &= -2\mathbf{y}^T\mathbf{P}(\nabla_{\tau^2}\mathbf{V})\mathbf{P}(\nabla_{\tau^2}\mathbf{V})\mathbf{P}\mathbf{y} + \mathrm{tr}(\mathbf{P}(\nabla_{\tau^2}\mathbf{V})\mathbf{P}(\nabla_{\tau^2}\mathbf{V})).
\end{aligned} \tag{12}$$

The expectations of the second derivatives, averaging over values of $\mathbf{y}$, are:

$$\begin{aligned}
\mathbb{E}\left[\frac{\partial \ell}{\partial \sigma_{\boldsymbol{\varepsilon}}^2 \partial \sigma_{\boldsymbol{\varepsilon}}^2}\right] &= -\,\mathrm{tr}(\mathbf{P}(\nabla_{\sigma_{\boldsymbol{\varepsilon}}^2}\mathbf{V})\mathbf{P}(\nabla_{\sigma_{\boldsymbol{\varepsilon}}^2}\mathbf{V})) \\
\mathbb{E}\left[\frac{\partial \ell}{\partial \sigma_{\boldsymbol{\varepsilon}}^2 \partial \tau^2}\right] &= -\,\mathrm{tr}(\mathbf{P}(\nabla_{\sigma_{\boldsymbol{\varepsilon}}^2}\mathbf{V})\mathbf{P}(\nabla_{\tau^2}\mathbf{V})) \\
\mathbb{E}\left[\frac{\partial \ell}{\partial \tau^2 \partial \tau^2}\right] &= -\,\mathrm{tr}(\mathbf{P}(\nabla_{\tau^2}\mathbf{V})\mathbf{P}(\nabla_{\tau^2}\mathbf{V})).
\end{aligned} \tag{13}$$

Finally, averaging the Hessian (12) and the Fisher information (13) yields the average information:

$$\begin{aligned}
\mathrm{AI}(\sigma_{\boldsymbol{\varepsilon}}^2, \sigma_{\boldsymbol{\varepsilon}}^2) &= -\mathbf{y}^T\mathbf{P}(\nabla_{\sigma_{\boldsymbol{\varepsilon}}^2}\mathbf{V})\mathbf{P}(\nabla_{\sigma_{\boldsymbol{\varepsilon}}^2}\mathbf{V})\mathbf{P}\mathbf{y} \\
&= -\mathbf{y}^T\mathbf{P}\mathbf{P}\mathbf{P}\mathbf{y} \\
\mathrm{AI}(\sigma_{\boldsymbol{\varepsilon}}^2, \tau^2) &= -\mathbf{y}^T\mathbf{P}(\nabla_{\sigma_{\boldsymbol{\varepsilon}}^2}\mathbf{V})\mathbf{P}(\nabla_{\tau^2}\mathbf{V})\mathbf{P}\mathbf{y} \\
&= -\mathbf{y}^T\mathbf{P}\mathbf{P}\mathbf{B}\mathbf{P}\mathbf{y} \\
\mathrm{AI}(\tau^2, \tau^2) &= -\mathbf{y}^T\mathbf{P}(\nabla_{\tau^2}\mathbf{V})\mathbf{P}(\nabla_{\tau^2}\mathbf{V})\mathbf{P}\mathbf{y} \\
&= -\mathbf{y}^T\mathbf{P}\mathbf{B}\mathbf{P}\mathbf{B}\mathbf{P}\mathbf{y},
\end{aligned} \tag{14}$$

where, notably, the computationally-expensive traces have canceled. Hence, our terms for the AI-REML update (10) are:

$$\nabla_{\boldsymbol{\theta}}\ell = \begin{bmatrix} \mathrm{tr}(\mathbf{P}) - \mathbf{y}^T\mathbf{P}\mathbf{P}\mathbf{y} \\ \mathrm{tr}(\mathbf{P}\mathbf{B}) - \mathbf{y}^T\mathbf{P}\mathbf{B}\mathbf{P}\mathbf{y} \end{bmatrix} \quad \text{and} \quad \mathrm{AI}(\ell) = \begin{bmatrix} \mathrm{AI}(\sigma_{\boldsymbol{\varepsilon}}^2, \sigma_{\boldsymbol{\varepsilon}}^2) & \mathrm{AI}(\sigma_{\boldsymbol{\varepsilon}}^2, \tau^2) \\ \mathrm{AI}(\sigma_{\boldsymbol{\varepsilon}}^2, \tau^2) & \mathrm{AI}(\tau^2, \tau^2) \end{bmatrix}. \tag{15}$$

Our next goal is to describe how to compute these values efficiently.

4.2 Efficient computation with AI-REML

To apply AI-REML, we will need to begin with an initial guess at τ^2 and $\sigma_{\boldsymbol{\varepsilon}}^2$, then repeatedly iterate the updates (10) using the expressions in (15). Since these expressions involve products of $N \times N$ matrices, and the update itself requires inversion of such a matrix, doing this explicitly is impractical for large N. Instead, we now describe how to carry out all required computations using implicit matrix-vector multiplication with the ARG. The basic tool we have for this is the algorithm presented in Lehmann et al. (2025), which allows us to multiply a vector by $\mathbf{B}$ with nearly linear complexity in the number of individuals; this is implemented in `tskit` as `genetic_relatedness_vector`.

The most expensive single operation we need to perform is multiplication by $\mathbf{P}$. First, note that to multiply by $\mathbf{P}$ it suffices to multiply by $\mathbf{V}^{-1}$; since $\mathbf{X}$ is only $N \times K$, multiplication by $\mathbf{X}$ is done explicitly. To multiply by $\mathbf{V}^{-1}$, we use the conjugate gradient method (Hestenes and Stiefel, 1952), an iterative algorithm which multiplies $\mathbf{V}$ to a vector multiple times to obtain $\mathbf{V}^{-1}\mathbf{x}$. Since $\mathbf{V} = \tau^2\mathbf{B} + \sigma_{\boldsymbol{\varepsilon}}^2\mathbf{I}$, multiplication by $\mathbf{V}$ is no harder than multiplication by $\mathbf{B}$. Multiplication by $\mathbf{P}$ lets us compute all terms in the AI matrix, and the non-trace terms in the gradient. Some further simplifications are possible: for instance, to compute $\mathbf{y}^T\mathbf{P}\mathbf{P}\mathbf{y}$, we first estimate the fixed effects by $\widehat{\mathbf{b}} = (\mathbf{X}^T\mathbf{V}^{-1}\mathbf{X})^{-1}\mathbf{X}^T\mathbf{V}^{-1}\mathbf{y}$. Then, we compute $\mathbf{P}\mathbf{y}$ by $\mathbf{P}\mathbf{y} = \mathbf{V}^{-1}(\mathbf{y} - \mathbf{X}\widehat{\mathbf{b}})$, and $\mathbf{y}^T\mathbf{P}\mathbf{P}\mathbf{y} = \|\mathbf{P}\mathbf{y}\|^2$.

The next ingredient we need for the gradient is the ability to compute traces, which turn out to be the most computationally intensive step. For instance, $\mathrm{tr}(\mathbf{P})$ in $\partial\ell/\partial\sigma_{\boldsymbol{\varepsilon}}^2$ is:

$$\begin{aligned}\mathrm{tr}(\mathbf{P}) &= \mathrm{tr}(\mathbf{V}^{-1}) - \mathrm{tr}(\mathbf{V}^{-1}\mathbf{X}(\mathbf{X}^T\mathbf{V}^{-1}\mathbf{X})^{-1}\mathbf{X}^T\mathbf{V}^{-1}) \\ &= \mathrm{tr}(\mathbf{V}^{-1}) - \mathrm{tr}((\mathbf{X}^T\mathbf{V}^{-1}\mathbf{X})^{-1}\mathbf{X}^T\mathbf{V}^{-1}\mathbf{V}^{-1}\mathbf{X}),\end{aligned} \tag{16}$$

First note that $\mathbf{X}^T\mathbf{V}^{-1}\mathbf{X}$ is only a $K \times K$ matrix, so can be inverted explicitly, and that furthermore the second trace term is a trace of a $K \times K$ matrix, which can be computed easily. So, the difficult part of computing $\mathrm{tr}(\mathbf{P})$ is computation of $\mathrm{tr}(\mathbf{V}^{-1})$. We obtain an unbiased estimate of each trace with the XTrace algorithm, which uses a fixed number of matrix-vector multiplications against random vectors (Epperly, Tropp, and Webber, 2024). In our default setting, we use 50 $\mathbf{P}$ multiplications for the estimation of each of the traces (see Table 1), which is conservative in our experience.

Combining these tools – implicit matrix-vector multiplication by $\mathbf{B}$, conjugate gradient for inverses, and XTrace for traces – we have highly efficient ways to compute AI-REML updates. There are a few more details for the full algorithm. First, we use a randomized Nyström preconditioner (Frangella, Tropp, and Udell, 2021) to speed up the conjugate gradient algorithm to perform multiplication by $\mathbf{V}^{-1}$. This preconditioner uses a randomized singular value decomposition that depends on the number of test vectors used. Similarly, trace estimation with the XTrace algorithm (Epperly, Tropp, and Webber, 2024) depends on a chosen number of test vectors. Default values for these and other parameters are listed in Table 1. Our implementation of AI-REML method is iterative and stochastic, and so a stopping criteria is needed. When the parameter estimates change by less than 5%, `tslmm` continues for an additional 15 steps and then terminates, returning the average value of the last 15 iterations.

How efficient is it? The matrix-vector algorithm for multiplication with $\mathbf{B}$ performs an $\mathcal{O}(N)$ setup step, then one update for each node above each added or removed edge, and so has computational complexity $\mathcal{O}(N + E\log N)$, where N is the number of individuals and E is the number of edges (Lehmann et al., 2025). In practice, runtime of variance component estimation is highly dependent on the number of matrix-vector multiplications, which we minimized through the use of modern randomized algorithms. Most matrix-vector operations are performed on many vectors, and so are parallelized across the number of available threads (our benchmarking results shown below, however, used a single thread). Furthermore, note that while the gradient

Table 1: The default hyperparameter settings of `tslmm`.

Hyperparameter name	Default value
Conjugate gradient convergence threshold	10^{-5}
Number of test vectors for gradient estimation	50
Rank of the randomized Nyström preconditioner	500
Number of test vectors for randomized HE method	50
Number of threads in matrix-vector multiplication	1
REML convergence criteria	0.05
Minimum REML steps before termination	15

is stochastic, the average information matrix is not; hence this AI-REML optimization routine is more stable than "fully-stochastic" second-order methods.

4.3 Initialization by Haseman–Elston

The AI-REML algorithm is iterative, so it needs a starting estimate. We initialize the variance component parameters using a randomized Haseman–Elston (HE) method. For a derivation, see Wu and Sankararaman (2018), but briefly the initial values of τ^2 and $\sigma_{\boldsymbol{\varepsilon}}^2$ are obtained by solving:

$$\begin{bmatrix} \mathrm{tr}(\mathbf{B}^2) & \mathrm{tr}(\mathbf{B}) \\ \mathrm{tr}(\mathbf{B}) & \mathrm{tr}(\mathbf{I}) \end{bmatrix} \begin{bmatrix} \tau^2 \\ \sigma_{\boldsymbol{\varepsilon}}^2 \end{bmatrix} = \begin{bmatrix} \mathbf{y}^T\mathbf{B}\mathbf{y} \\ \mathbf{y}^T\mathbf{y} \end{bmatrix}, \tag{17}$$

in which the traces are computed by XTrace (Epperly, Tropp, and Webber, 2024) with 50 test vectors as above. Computing $\mathrm{tr}(\mathbf{B}^2)$ and $\mathrm{tr}(\mathbf{B})$ is far cheaper than a single AI-REML iteration because multiplication with $\mathbf{P}$ requires multiple rounds of $\mathbf{B}$ multiplication. Then, solving equation (17) is immediate. Therefore, initialization by the randomized HE method only occupies a small fraction of the overall AI-REML routine.

4.4 Simulations for evaluation of methods

Next, we produced a collection of simulations to benchmark the methods. Briefly, we used `msprime` (Kelleher, Etheridge, and McVean, 2016; Baumdicker et al., 2022) for a coalescent simulation of an exponentially growing panmictic population. Looking back through time (following the coalescent simulation), the population starts at 100,000 individuals today, and declines at a rate of 1% per generation. Each individual has a genome that was either 1Mbp (10^6bp) or 100Mbp (10^8bp) (depending on the situation, below) and experienced a recombination rate of 10^{-8} per base-pair per generation. From the resulting population we sampled varying numbers of diploid individuals.

For real data, we do not have access to the true ARGs. So, for some tests we used `msprime` to simulate mutations on the tree sequences with rate 10^{-8} per base-pair per generation to obtain haplotypes, which were used in `tsinfer` (Kelleher et al., 2019) and `tsdate` (Wohns et al., 2022) (with the "variational gamma" algorithm) to obtain inferred and dated tree sequences.

To simulate phenotypes, we first simulated genetic values using the algorithm described in Appendix C, then added independent environmental noise, as in the ARG-LMM model. (In all benchmarks, phenotypes were generated from the true simulated tree sequence, not inferred tree sequences.) To make the variance due to genetic and non-genetic components roughly comparable, we divided the genetic values by the square root of $4\sum_T s_T t_{\mathrm{root},T}$, where the sum is over local trees and $t_{\mathrm{root},T}$ is the time of the root of a local tree

T. This is because the genetic variance of a single haploid genome is proportional to $\sum_T s_T t_{\mathrm{root},T}$, the total area from the single genome up to the root along the entire genome, and so the genetic variance of a diploid is bounded above by four times this quantity. The final phenotypic value for each individual was obtained by multiplying their genetic value by τ and adding an independent Gaussian environmental deviation with mean 0 and variance σ_ϵ^2.

4.5 Runtime

To assess the scalability of the methods, we simulated phenotypes described above on ARGs, using $\tau^2 = 1$ and $\sigma_\epsilon^2 = 1$, and estimated the variance components with `tslmm`. We did this five times on each simulated tree sequence to obtain an average runtime.

The results show that `tslmm` conforms to the predicted scaling. Variance component estimation for 10^5 individuals and a one-chromosome genome took only a few hours on a single thread. Runtimes for `tslmm` on simulated datasets for 100Mbp diploid genomes are shown in Figure 2, as a function of increasing ARG size. Although matrix-vector multiplication using the algorithm of Lehmann et al. (2025) is $\mathcal{O}(N + E\log(N))$, the number of multiplications required for conjugate gradient could also scale with N or E, resulting in different computational complexity. However, the lines in Figure 2 suggest that this is not the case: empirical runtimes for variance component estimation are fit well by an $\mathcal{O}(N + E\log(N))$ model.

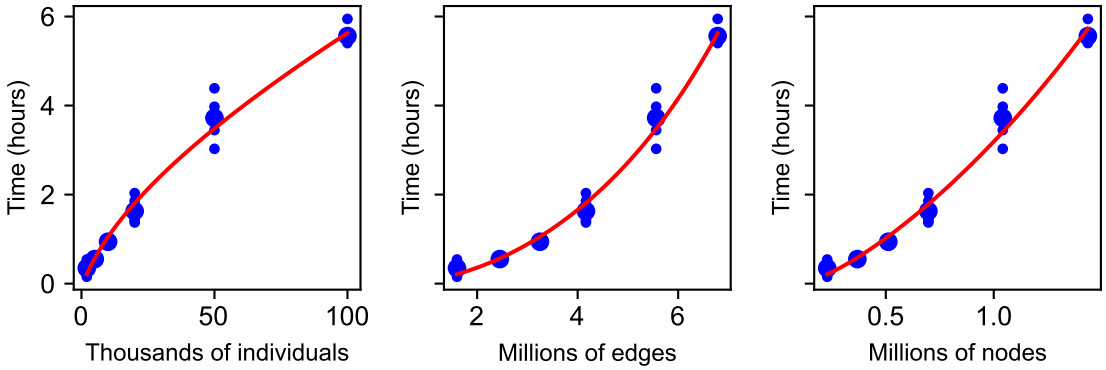

Figure 2: Measured runtimes (on a single thread of Xeon Gold 6140 CPU) for variance component estimation on 30 simulations of a 100Mbp genome with a recombination rate of 10^{-8}: five independent simulations at each of six different sample sizes of diploid individuals. Since runtime is expected to scale with the number of edges, we show the same results against number of diploid individuals (left), total number of edges (middle), and number of nodes (right). Larger dots show the mean across replicates, and lines show results from the linear model "runtime = 87 (# individuals/10^3) + 365 (# edges/10^6) $\log$(# individuals/10^3) - 185". The linear model was fit by least squares, and displayed red lines are obtained by using the average value of the unspecified variable across simulated scenarios with the given value of the specified variable (for instance, in the first plot, the line shows predicted runtime at the number of individuals shown on the x-axis and the average number of edges seen across the simulations having that many individuals).

4.6 Accuracy of variance component estimates

To evaluate accuracy of the method we used simulated tree sequences of length 1Mbp with all other parameters including the demography as described above. On each tree sequence we simulated phenotypes with $\sigma^2_{\boldsymbol{\varepsilon}} = 1$ and a range of values of $\tau^2 \in \{0.0, 0.2, 0.4, 0.6, 0.8, 1.0\}$. Then, we estimated variance components, both using the true tree sequences and the inferred tree sequences (as described above). We simulated phenotypes, and then inferred variance components, five times on each tree sequence and at each value of τ^2, each, to quantify the variability of the estimates due to different realizations of simulated phenotypes. Alongside `tslmm`'s main AI-REML routine, we present the performance of the randomized HE method that `tslmm` uses as a starting location for optimization. The relative performance of the two methods is of interest because there is growing interest in using the HE estimator for large-scale quantitative genetics due to its substantially lower computational cost relative to REML-based methods (Wu and Sankararaman, 2018; Pazokitoroudi et al., 2020; Zhu et al., 2024; Loya et al., 2025).

The results are shown in Figure 3 for $N = 2000$ and $N = 50,000$ diploids; additional values of N are shown in Supplementary Figures A1 and A2. The figure compares estimation on true and inferred tree sequences, as well as the final REML estimates to the estimates produced by the HE method. When the true ARG is available, `tslmm` accurately recovers the variance components, to usually within 10%–20% for the larger sample sizes. Estimates remain nearly as accurate when using an ARG inferred with `tsinfer` and `tsdate` rather than the true ARG. However, as seen in Supplementary Figure A1, there is an upwards bias in the inferred τ^2 for the largest sample size ($N = 100,000$ diploids) for larger values of τ^2 (Figure A1). This bias can also be seen in the mean absolute values (MAE) where `tslmm` on inferred tree sequences has a much larger MAE than on true tree sequences (Figure A2) despite a comparable variance (Figure A1). The fact that estimation on inferred ARGs works better with 50,000 than with 100,000 samples (as opposed to using true ARGs, for which the larger sample size has lower MAE), implies that the bias is due to ARG inference artifacts, which should be correctable given ongoing work on improving ARG inference methods.

Furthermore, the final AI-REML estimates are substantially more accurate than their HE counterparts for both true and inferred trees, with even AI-REML on inferred tree sequences producing better estimates than HE on true tree sequences.

4.7 Relationship to narrow-sense heritability

One major purpose of LMMs is to estimate additive genetic variance and narrow-sense heritability. A standard practice is to obtain the estimates τ^2 and $\sigma^2_{\boldsymbol{\varepsilon}}$ for the corresponding LMM. Then, one interprets τ^2 as the additive genetic variance and computes the narrow-sense heritability as $h^2 = \tau^2/(\tau^2 + \sigma^2_{\boldsymbol{\varepsilon}})$. However, τ^2 in our current model has a different interpretation, and so we describe below this fundamental distinction.

Additive genetic variance of a group of individuals (samples) is defined to be the variance of their (additive) genetic values. If we write $\mathbf{g} = \mathbf{Zu}$ for these genetic values, then this variance is:

$$V_G = \frac{1}{N}\sum_{i=1}^{N}(\mathbf{g}_i - \overline{\mathbf{g}})^2 = \frac{1}{N}(\mathbf{P}_N\mathbf{g})^T(\mathbf{P}_N\mathbf{g}), \tag{18}$$

where $\mathbf{P}_N = \mathbf{I}_N - \frac{1}{N}\mathbf{1}_N\mathbf{1}_N^T$. This quantity could be treated as an estimate of the population's additive genetic variance, but the accuracy of the estimate will depend on how the sampled individuals in $\mathbf{g}$ represent the population. It's tempting to relate τ^2 and V_G, however these quantities differ fundamentally; τ^2 is a parameter in the generative model (ARG-LMM), while V_G is an emergent property whose distribution depends on τ^2. For instance, the *expected value* of V_G in a diploid Wright-Fisher population of size N_e is $4N_eP\tau^2$ (Lynch

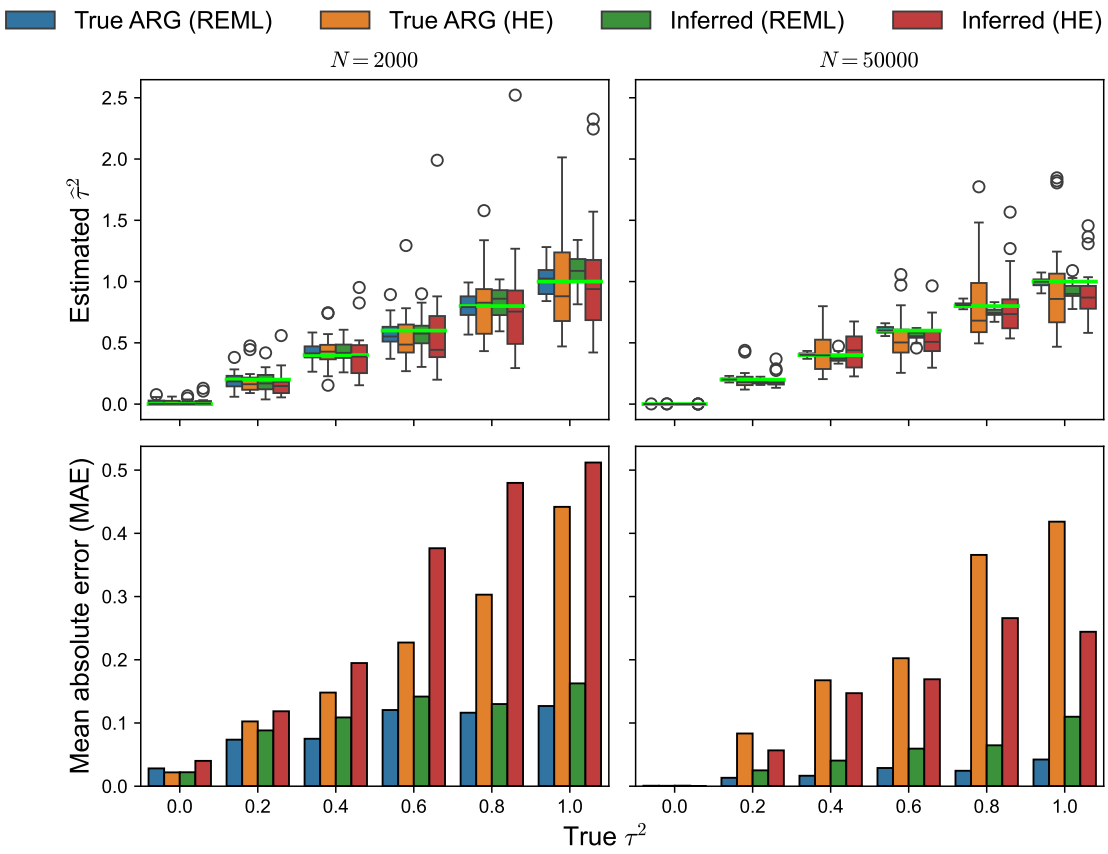

Figure 3: `tslmm` estimates of the genetic variance component τ^2 from true ("True") and inferred ("Inferred") ARGs, using the randomized Haseman–Elston method ("HE") and AI-REML ("REML"), over 200 replicates of 1Mbp genome with a recombination rate of 10^{-8} and $\sigma_\epsilon^2 = 1$, at each of six different values of $\tau^2 \in \{0.0, 0.2, 0.4, 0.6, 0.8, 1.0\}$. Left column: $N = 2,000$ diploid samples; right column: $N = 50,000$ diploid samples. (Top) Distribution of estimates from the four methods with a standard boxplot: horizontal black line at the median, box extends to the quartiles, whiskers to quartiles $\pm$ 1.5 times the interquartile range. The superimposed horizontal line bright green lines denote the true parameter values. (Bottom) The mean absolute error with respect to the true parameter value.

and Hill, 1986), where P is the number of loci, but genetic drift causes V_G of the population to fluctuate randomly over time. Furthermore, V_G is also impacted by linkage-disequilibrium, which is in turn driven by population processes that impact phenotypes, such as selection and migration (see Lara et al., 2022 and references therein).

How do we estimate V_G from `tslmm`? First note that, still taking the ARG as fixed but the sub-edge effects $\mathbf{u}$ as random, the expected value of the estimator (18) is:

$$\begin{aligned}
\mathbb{E}\left[V_G\right] &= \frac{1}{N}\mathbb{E}\left[(\mathbf{P}_N\mathbf{g})^T(\mathbf{P}_N\mathbf{g})\right] \\
&= \frac{1}{N}\mathbb{E}\left[\mathrm{tr}(\mathbf{g}^T\mathbf{P}_N^T\mathbf{P}_N\mathbf{g})\right], \quad \text{because } \mathbf{g}^T\mathbf{P}_N^T\mathbf{P}_N\mathbf{g} \text{ is a scalar} \\
&= \frac{1}{N}\,\mathrm{tr}(\mathbf{P}_N\mathbb{E}\left[\mathbf{g}\mathbf{g}^T\right]\mathbf{P}_N^T), \quad \text{because } \mathrm{tr}(AB) = \mathrm{tr}(BA).
\end{aligned}$$

Now, because $\mathbb{E}\left[\mathbf{g}\mathbf{g}^T\right] = \tau^2\mathbf{B}$, and since $\mathbf{P}_N\mathbf{B}\mathbf{P}_N^T = \widetilde{\mathbf{B}}$, the *centered* branch GRM, we are left with $\mathbb{E}\left[V_G\right] = \tau^2\,\mathrm{tr}(\widetilde{\mathbf{B}})/N$. This suggests that V_G should be close to $\tau^2\,\mathrm{tr}(\widetilde{\mathbf{B}})/N$. To get an idea of how close (if our model describes reality well), we can compute the variance of this estimator. A similar calculation, provided in Appendix B, shows that:

$$\mathrm{Var}\left(V_G\right) = \frac{\tau^4}{N^2}\sum_{n,m}(\widetilde{\mathbf{B}}_{nm})^2.$$

Since the off-diagonal elements of $\widetilde{\mathbf{B}}$ are $\mathcal{O}(1/N)$, this variance is $\mathcal{O}(1/N)$. Therefore, $\tau^2\,\mathrm{tr}(\widetilde{\mathbf{B}})/N$ is a consistent estimator of V_G, with an error of the order $1/\sqrt{N}$. We raise caution that this calculation only pertains to panmictic populations. The error can be substantial in practice: for instance, $\mathrm{Var}\left(V_G\right)$ is larger in the presence of greater linkage, because the random value of a particular $\mathbf{u}$ will have a larger effect on the realized V_G. For more discussion on the estimation of and changes in genetic variance, see Lara et al. (2022).

5 Genetic prediction: BLUPs

Once `tslmm` has estimated variance components from data, genetic prediction is relatively straightforward: the typical approach is to use the "best linear unbiased predictor" (BLUP) of the LMM. Suppose we have an ARG that describes relationships between a number of individuals, N_o of which are phenotyped, and $N - N_o$ of which are not. As before, we are working with the model (4), and $\mathbf{y}$ is the vector of all observed phenotype values, but now we wish to find the conditional distribution of the genetic values of a set of individuals whose phenotypic values may or may not have been observed.

Since the model is multivariate Gaussian, this reduces to linear algebra. It is easiest to first explain the general principles that we use, then translate the results into the specific model at hand. Suppose that $\mathbf{x}$ and $\mathbf{z}$ are jointly Gaussian, so that the vector $(\mathbf{x}, \mathbf{z})$ is multivariate Gaussian with mean $\boldsymbol{\mu}$ and covariance $\boldsymbol{\Sigma}$. Write $\boldsymbol{\mu}_\mathbf{x}$ and $\boldsymbol{\mu}_\mathbf{z}$ for the sub-vectors of $\boldsymbol{\mu}$ giving the means of $\mathbf{x}$ and $\mathbf{z}$, and write $\boldsymbol{\Sigma}_{\mathbf{xx}}$, $\boldsymbol{\Sigma}_{\mathbf{zz}}$, and $\boldsymbol{\Sigma}_{\mathbf{xz}}$ for the respective submatrices of $\boldsymbol{\Sigma}$. Then, the conditional expectation of $\mathbf{x}$ given $\mathbf{z}$ is:

$$\mathbb{E}\left[\mathbf{x} \mid \mathbf{z}\right] = \boldsymbol{\mu}_\mathbf{x} + \boldsymbol{\Sigma}_{\mathbf{xz}}\boldsymbol{\Sigma}_{\mathbf{zz}}^{-1}(\mathbf{z} - \boldsymbol{\mu}_\mathbf{z}), \tag{19}$$

and the conditional variance of $\mathbf{x}$ given $\mathbf{z}$ is:

$$\mathrm{Var}(\mathbf{x} \mid \mathbf{z}) = \boldsymbol{\Sigma}_{\mathbf{xx}} - \boldsymbol{\Sigma}_{\mathbf{xz}}\boldsymbol{\Sigma}_{\mathbf{zz}}^{-1}\boldsymbol{\Sigma}_{\mathbf{zx}}. \tag{20}$$

Note that $\mathbb{E}\left[\mathbf{x} \mid \mathbf{z}\right]$ is a linear function of $\mathbf{z}$, and it is a general fact that the conditional mean is an unbiased estimator that minimizes mean-squared error, and so $\mathbb{E}\left[\mathbf{x} \mid \mathbf{z}\right]$ is, in some sense, the "best linear unbiased predictor" of $\mathbf{x}$ given $\mathbf{z}$.

Now suppose we'd like to estimate the genetic values of a set of unphenotyped individuals. We denote the individuals whose phenotypes we have observed with a subscript o, so that $\mathbf{y}_o$ are the observed phenotypes, and n for the non-phenotyped individuals, so that $\mathbf{g}_n$ are their genetic values. (In what follows the two sets of individuals can in fact overlap, but it will be easier to discuss using this convention.) To use the general theory, set $\mathbf{x} = \mathbf{g}_n$, and $\mathbf{z} = \mathbf{y}_o$. Then, the following quantities in equation (19) are:

$$\begin{aligned}
\boldsymbol{\Sigma}_{\mathbf{xz}} &= \mathrm{Cov}\left(\mathbf{g}_n, \mathbf{y}_o\right) = \mathrm{Cov}\left(\mathbf{g}_n, \mathbf{g}_o\right) = \tau^2\mathbf{B}_{n,o} \\
\boldsymbol{\Sigma}_{\mathbf{zz}} &= \mathrm{Cov}\left(\mathbf{y}_o, \mathbf{y}_o\right) = \tau^2\mathbf{B}_{o,o} + \sigma_{\boldsymbol{\varepsilon}}^2\mathbf{I}_{o,o} \\
\boldsymbol{\mu}_{\mathbf{z}} &= \mathbf{X}_o\mathbf{b} \quad \text{and} \quad \boldsymbol{\mu}_{\mathbf{x}} = \mathbf{0}.
\end{aligned} \tag{21}$$

In practice, we replace τ^2, $\sigma_{\boldsymbol{\varepsilon}}^2$, and $\mathbf{b}$ with their estimates $\hat{\tau}^2$, $\hat{\sigma}_{\boldsymbol{\varepsilon}}^2$, and $\hat{\mathbf{b}}$ and hence obtain empirical BLUPs (Robinson, 1991; Sorensen and Gianola, 2002), which are:

$$\widehat{\mathbf{g}}_n = \hat{\tau}^2\mathbf{B}_{n,o}\left(\hat{\tau}^2\mathbf{B}_{o,o} + \hat{\sigma}_{\boldsymbol{\varepsilon}}^2\mathbf{I}_{o,o}\right)^{-1}\left(\mathbf{y}_o - \mathbf{X}_o\hat{\mathbf{b}}\right). \tag{22}$$

The conditional variances can be calculated using equation (20). The subscripted matrices are the submatrices of the original matrix corresponding to the subscripts. Note that these calculations require access to multiplication with submatrices of $\mathbf{B}$, which can be done as described in Appendix D.

In general, we may set $\mathbf{x}$ to any linear combination of sub-edge effects in the tree sequence, as long as we have access to $\boldsymbol{\Sigma}_{\mathbf{xz}}$, namely the covariance matrix between the unobserved variable and the observed phenotypic values $\mathbf{y}_o$. The latter provide information about other model variables or their combinations. Hence, $\boldsymbol{\Sigma}_{\mathbf{zz}}^{-1}(\mathbf{z} - \boldsymbol{\mu}_{\mathbf{z}})$ is always set to $(\tau^2\mathbf{B}_{o,o} + \sigma_{\boldsymbol{\varepsilon}}^2\mathbf{I}_{o,o})^{-1}(\mathbf{y}_o - \mathbf{X}_o\mathbf{b})$. This is the most compute intensive part of equation (19) but we only need to compute it once for all BLUP predictions. Since $\mathbf{y}_o - \mathbf{X}_o\mathbf{b}$ is a vector, we can obtain $(\tau^2\mathbf{B}_{o,o} + \sigma_{\boldsymbol{\varepsilon}}^2\mathbf{I}_{o,o})^{-1}(\mathbf{y}_o - \mathbf{X}_o\mathbf{b})$ through conjugate gradient as mentioned above. This has many potential applications, such as estimating mutation and edge effects, node values, and individual genetic values, but also their combinations along the genome for genome-wide association studies, over time for studying phenotype evolution, or along the genome and over time to study the origin of the associations. We leave the in-depth investigation of this feature to future work.

5.1 Accuracy of BLUPs

We now assess how well `tslmm`'s BLUPs match the true genetic values, using the same simulated phenotypes and ARGs on 1Mbp genomes described in section 4.4. The benchmarks from true tree sequences offer the theoretical performance of `tslmm` provided the ARG is correct, while those from inferred tree sequences indicates the degree to which errors in the ARG introduced by inference affect the quality of the BLUPs. To do this, we first randomly chose half of the individuals as "phenotyped" and set aside the others as "unphenotyped" (but we have true genetic values and phenotypic values for all of them). Using the ARG (either true or inferred, depending on the test), we estimated variance components using only the "phenotyped" individuals, then using these estimates and equation (22), computed BLUPs for the "unphenotyped" individuals.

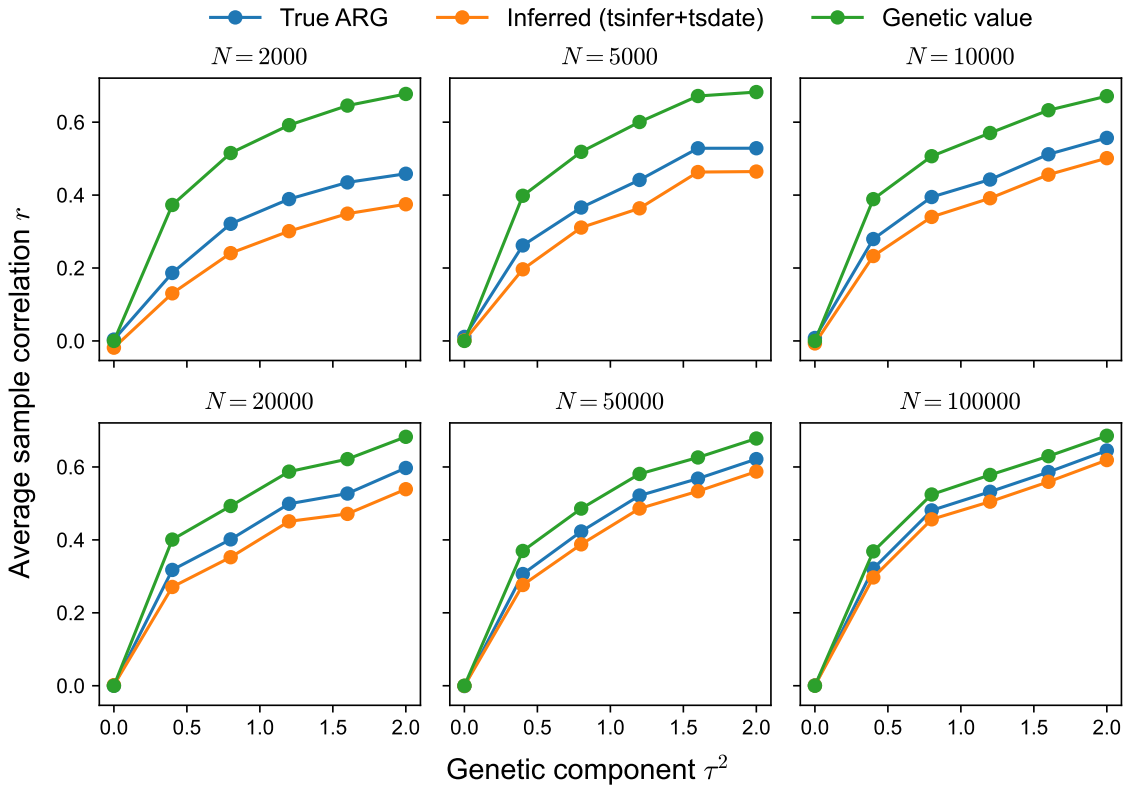

Figure 4: Average Pearson correlation between phenotypic values and `tslmm`'s BLUP of genetic values from true (blue) and inferred (orange) ARGs, using the same simulations as in Figure 3. The green dotted line is the correlation between the phenotypic value and true genetic value, thereby providing the upper bound of phenotype prediction accuracy. For each simulation, the model was fit using phenotypes from half of the individuals in the ARG, and prediction accuracy is measured using BLUPs from the remaining "held-out" individuals.

Figure 4 shows the resulting Pearson correlations between predicted genetic values and the phenotypic values, along with correlations between the true genetic values and the phenotypic values for comparison. Since τ^2 controls the proportion of the heritable component, correlations are larger at larger values of τ^2, simply because more heritable phenotypes are more predictable (recall that the phenotype is equal to genetic value plus environmental component). So, even perfect prediction of genetic values (the dotted line in Figure 4) results in correlations below 1. `tslmm`'s predictions perform only slightly worse than this optimal behavior, but approach perfect prediction at the largest sample sizes ($N = 100,000$). Furthermore, we see only a slight decrease in the phenotype prediction accuracy when using an inferred ARG compared to the true ARG.

5.2 Prediction accuracy with more realistic simulations

The results above are promising, but the simulations we used represent an ideal situation. First, the phenotypes were simulated under the ARG-LMM model (rather than discrete effects associated with mutations). Second, the Wright–Fisher model of demography generates a GRM with very little fine-scale structure, while real data often exhibits fine-scale spatial structure that could in principle affect the accuracy of `tslmm`'s randomized linear algebra methods. We take this opportunity to test the accuracy of phenotype prediction with `tslmm` on a more realistic simulation.

To do this, we simulated a population forward in time using `SLiM` version 4 (Haller and Messer, 2023), under an individual-based model with spatially-restricted mating and dispersal, following Chevy et al. (2025). The simulation takes place in a continuous square spatial domain, where the population size is stochastic but kept roughly constant by local competition between nearby individuals. Average dispersal distance and strength of competition were chosen to result in a strong signal of isolation by distance, so that nearby individuals are typically close relatives. This population was simulated for 150 time units (where a time unit corresponds to the expected lifetime of an individual) and the resulting pedigree was recorded. Subsequently, we took all individuals alive at the end of the simulation and simulated a tree sequence backward in time within the pedigree (producing an un-coalesced ARG) using `msprime` with the `fixed_pedigree` model (Anderson-Trocmé et al., 2023). This tree sequence was for a genome of length of 10^8bp and a recombination rate of 10^{-8}. Finally, we "recapitated" the tree sequence by completing the within-pedigree tree sequence backwards in time with the standard coalescent model in `msprime`, using the average population size from the spatial simulation. We repeated both within-pedigree and recapitation coalescent simulations to acquire multiple realizations of the tree sequence from the pedigree. Finally, for each simulated tree sequence, we simulated neutral mutations at a rate of 10^{-8}, again using `msprime`. This resulted in a total of 13,307 spatially-indexed individuals.

We then simulated the phenotypes using `tstrait` (Tagami, Bisschop, and Kelleher, 2024) that assigns effects to mutations, instead of **Algorithm T** that assigns effects to sub-edges. One percent of the mutations were set to be causal and their effects were drawn from a standard normal distribution. As before, the resulting genetic value was divided by the square root of the total area up to the roots, so that τ^2 and σ_ε^2 would be of comparable size. Finally, we returned phenotypic values by adding random environmental deviations from the standard normal distribution to the genetic values. This resulted in a substantial spatial autocorrelation of genetic values, shown in Figure 5A.

Finally, we computed BLUPs: first, we estimated variance components using phenotypes of 6,654 individuals, and then we obtained BLUPs of genetic values for a separate set of 6,653 "unphenotyped" individuals. Figure 5B shows the accuracy measured as the correlation between true phenotypes and predicted genetic values; boxplots show the distribution of correlations across replicate simulations. (As in Figure 4, perfect prediction is not possible because the phenotypes also include the "environmental" deviation.) In this situation, prediction using both the true ARG and an inferred ARG (leftmost boxplots) are very good – as good as the best of Figure 4, likely because the spatial structuring leads to a more even distribution of close relatives. In this situation, HE method (third boxplot) does roughly as well as AI-REML.

5.3 Comparison to pedigree and Monte-Carlo GRMs

Finally, we use the same spatial simulations in the previous section to compare `tslmm` to two other methods for phenotype prediction: using the pedigree, and using the Monte-Carlo estimate of the branch GRM generated by `arg-needle-lib`. Both use the same LMM, but differ only in what GRM is used.

Comparison to the pedigree is natural, because plant and animal breeding have long predicted genetic values using a pedigree GRM that computes the expected amount of shared genetic material between rela-

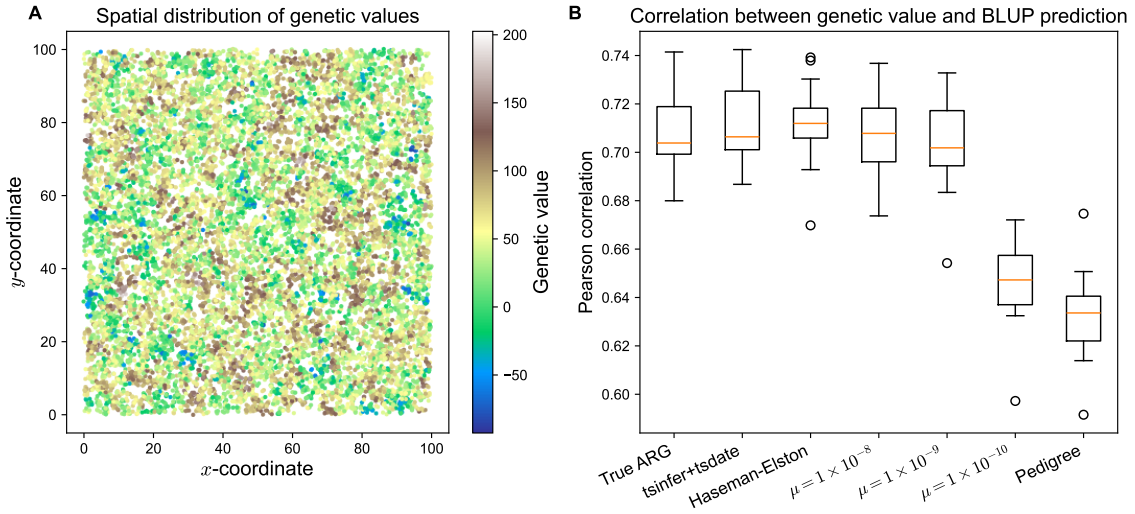

Figure 5: Accuracy of ARG, genotype, and pedigree BLUPs for non-phenotyped individuals in a spatially structured population. (A) The spatial distribution of individuals' genetic values, exhibiting spatial autocorrelation. (B) The Pearson correlation between BLUPs and true genetic values over 20 replicate simulations, summarized with a boxplot (interquartile range) and median (orange line). The x-axis labels refer to the type of GRM: "True": `tslmm` using the true ARG; "`tsinfer` + `tsdate`": `tslmm` using the ARG inferred by `tsinfer` and `tsdate`; "μ = value ": REML (`nloptr`) using the Monte-Carlo SNP genotype GRM (see text) computed from simulated mutations at rate μ by `arg-needle-lib`; "Pedigree": Custom REML optimization via `nloptr` with the pedigree GRM. All methods except `tsinfer` +`tsdate` used the true ARG.

tives (Wright, 1922; Emik and Terrill, 1949; Cruden, 1949). This is done by computing BLUPs in the same way we have described above (Henderson, 1984; Lynch and Walsh, 1998; Mrode and Pocrnic, 2023). These BLUPs are conditional on the observed family structure encoded in the pedigree, while leaving ancestry beyond the pedigree founders unknown, and are hence averaged over. With ARG-LMM we compute BLUPs conditionally on an observed (or inferred) realization of recombinations, mutations and Mendelian segregations since the most recent common ancestors of all sampled individuals, as this is what is recorded in the ARG. So, provided the "observed" ARG is accurate, the BLUPs from ARG-LMM in `tslmm` should be more accurate than from a pedigree-based LMM – but it is not clear by how much or if ARG inference will reduce the accuracy. We computed the pedigree GRM with the `getASubset` function from the `pedigreeTools` package (Vazquez et al., 2018) using the Colleau (2002)'s algorithm.

We also compared our method to the Monte-Carlo estimate of the GRM produced by `arg-needle-lib` (Zhang et al., 2023). This works by computing the usual genotype GRM after randomly generating mutations on the ARG under the infinite-sites mutation model, whose expectation is the branch GRM. Zhang et al. (2023) showed that this performs well if mutation rate is high enough, but it is of interest to see how high mutation rate must be, as Zhu et al. (2024) finds that power and bias are affected by the mutation rate. To do this, we used `arg-needle-lib` on true tree sequences generated by the simulation, setting $\alpha = 0$ and a range of Monte-Carlo mutation rates.

For both pedigree and Monte-Carlo GRMs, we computed BLUPs using `nloptr` to optimize the REML objective. The remaining boxplots in Figure 5B show the resulting accuracies. Results for Monte-Carlo estimated GRMs (labeled by μ, the mutation rate used) show that as previously reported, accuracy is roughly equivalent as for our exact method with higher mutation rate ($\mu = 10^{-8}$), but decreases at lower mutation rates. The final boxplot, labeled "Pedigree", shows that prediction using the pedigree performs substantially less well, as expected.

6 Discussion

In this paper, we have presented a collection of methods and open source software that use the ARG to efficiently fit LMMs to population phenotype data, for variance component estimation and genetic prediction. The methods are implemented in the `tslmm` package and make use of an implicit matrix-vector multiplication algorithm as well as modern randomized linear algebra algorithms to allow computation on large (biobank-scale) datasets. The methods are highly efficient, an efficiency that comes from using the ARG – specifically, the succinct tree sequence encoding – as a sparse matrix representation of the underlying relatedness structure of a population. We have also discussed the relationship of this ARG LLM to pedigree and genotype LLMs, and derived a generative model from the underlying mutational model.

Using the ARG allowed us to work with an explicitly generative model. As recently reviewed by Lehmann et al. (2025), many of the methods for estimating phenotype variance components and genetic values use essentially the same LMM, differing mostly in choice of GRM. Instead of focusing on the GRM, we focus on the underlying model for genetic values. Sparseness of the resulting model derives from the generative biological processes themselves, and this sparse structure of the model can be leveraged for efficient computations with the LMM (Rue and Held, 2005). Although such a structure has been found for pedigree data (Henderson, 1976) and has been leveraged for efficient pedigree algorithms (e.g. Meuwissen and Luo, 1992; Colleau, 2002), finding it for observed genotype matrices has so far been elusive (but see Misztal, 2015). In Lehmann et al. (2025) and this paper, we turned to ARG for this structure, because ARGs encode the generative process with past branching/coalescence, recombination, and mutation events.

In addition to computational efficiency, the generative model lets us examine assumptions and points the

way towards generalizations. So, we carefully described the model, including a first-principles derivation from a model of complex traits (Appendix A). As one possible extension, one could incorporate information about the locations of causal mutations along the genome, simply by rescaling genomic coordinates so that the length of a region is proportional to the expected proportion of mutational variance the region is responsible for. Similarly, the model assumes that mutations fall uniformly in time, even though this is known not to be the case for mutations under selection and causal mutations are likely under some form of selection. So, a readjustment of time – similar to the genotype GRM weighting by functions of frequency – may give better estimates. However, carefully exploring the many strategies is beyond the scope of the current paper.

Heritability and mutational variance The generative model we use has advantages in interpretation over the models used by pedigree and genotype GRMs. With a pedigree, we model variation of genetic values between individuals as a function of their parent average (between family) terms and Mendelian sampling (within family) terms. The variance of these terms in the pedigree model and its GRM is parameterized by the additive genetic variance of the pedigree founders (Henderson, 1984; Quaas, 1988; Kennedy, Schaeffer, and Sorensen, 1988; Lynch and Walsh, 1998; Mrode and Pocrnic, 2023). With genotypes, we model variation of genetic values between individuals as a function of their observed genotypes and unobserved allele substitution effects, hence the genotype model and its GRM are parameterized by the variance of allele substitution effects (Whittaker, Haley, and Thompson, 1997; Meuwissen, Hayes, and Goddard, 2001). There are several variations of the genotype model with a corresponding GRM, which try to connect the variance of allele substitution effects with additive genetic variance (VanRaden, 2008; Yang et al., 2010; Speed et al., 2012). However, this connection requires additional assumptions (Gianola et al., 2009; Campos, Sorensen, and Gianola, 2015; Speed and Balding, 2015). Critically, additive genetic variance is by definition a function of both genotypes and allele substitution effects (Falconer, 1995; Lynch and Walsh, 1998) and there are methods that follow this definition (Hou et al., 2019; Schreck, Piepho, and Schlather, 2019; Lara et al., 2022; Feldmann, Covarrubias-Pazaran, and Piepho, 2023). Our analysis matches this definition; we show that additive genetic variance depends on the sample at hand, because genotype distributions vary between samples. With the ARG, we model variation of genetic values between individuals as a function of edge effects, hence the ARG model and its GRM are parameterized by the variance of edge effects (τ^2). We can estimate this generative parameter (with units of variance generated per generation) thanks to the fact that the ARG lets us express relatedness in units of generations – another example of estimated ARGs providing a time axis for genomic data.

We also discussed estimation of the additive genetic variance for a given sample. In some models, mutational variance and additive genetic variance have a closer relationship. For instance, it is common to define the relatedness matrix as $\mathbf{X}\mathbf{X}^T$, where $\mathbf{X}_{ij} = (\mathbf{x}_{ij} - p_j)/(p_j(1-p_j))^\alpha$ (or $\mathbf{X}_{ij} = (\mathbf{x}_{ij} - 2p_j)/(2p_j(1-p_j))^\alpha$ in diploids), where $\mathbf{x}_{ij}$ is the number of derived alleles at locus j carried by individual i, p_j is the derived allele frequency in the sample at locus j, and $\alpha = 1/2$ (VanRaden, 2008; Yang et al., 2010). Then, for haploids or if Hardy-Weinberg equilibrium holds, $\mathrm{tr}(\mathbf{X}\mathbf{X}^T)$ is equal to M, the number of SNPs, provided the linkage-disequilibrium across loci is sufficiently weak. In this case, the two quantities (τ^2 and $\mathbb{E}\left[V_G\right]$) are essentially the same, differing only by a factor of M. However, this is not the case in our model, nor would it be if $\alpha \neq 1/2$ is a good description of reality.

On unobserved mutations Although the branch GRM naturally arises from a mutational model, the choice to use the branch GRM can be counterintuitive. This is because the branch GRM treats the ARG as fixed while treating the number and locations of the causal mutations on the ARG as random, even though mutations are more directly observed (but not necessarily the causal mutations). To clarify this relationship, we give careful consideration to these assumptions, including a more mathematical treatment in the Appendix. (In fact, we

can be slightly more general than the infinite-sites model, which allows recurrent mutations in unrelated edges.) In practice, the model should work well for complex traits whose causal mutations are unknown and perhaps ungenotyped. In theory, we do observe mutations more directly than the ARG, hence can perfectly determine their number and locations, but their effect sizes are still undetermined, and so the random effects models one uses in the two cases are very similar. If we have strong prior knowledge of which alleles are causal, and these are all genotyped, a corresponding genotype GRM just with these sites is expected to work better (although computing with it using an ARG could be very efficient). In practice, however, genotype information is not so certain. A mathematically appealing feature of treating mutations as random is that genetic fixed effects all collapse to an intercept, alleviating the burden of modeling fixed effects. Which approach works better in practice (or, perhaps a mix of the two) remains an empirical question.

Limitations and robustness More generally, there are many directions to explore how this might be best applied to real data. One such factor is the ARG inference method, including data pre-processing steps and choice of software. The ideal ARG-based estimator would be unbiased, so that effects of the many inevitable errors in ARG estimation "average out" in the downstream estimates. Furthermore, the Gaussian model for edge effects may be a better choice for some traits than others.

Some aspects of the method suggest the results will be robust. For instance, our derivation of the AI-REML algorithm is based on a Gaussian model, even though mutations appear on branches in a discrete fashion. However, Harville (1977b) and Westfall (1987) showed that REML coincides with minimum norm quadratic unbiased estimation (MINQUE), which is a moment-based estimator. Hence, normality is not a requirement for consistent point estimation, provided the covariance matrix $\mathbf{V}$ is correct. Indeed, `tslmm` returned consistent variance component estimates and BLUPs in the spatial benchmark, where the phenotype was generated from discrete mutations by `tstrait` (Figure A3).

Another way things can go wrong in practice is in convergence of estimates, but results here are again encouraging. Above, we saw that there is a fortuitous cancellation in the average information matrix for the ARG-LMM model that may contribute substantially to AI-REML's efficiency. The Hessian (equation (12)) and the Fisher information (equation (13)) each have a trace term, but these cancel in average information (equation (14)). This is not only computationally convenient, but it also means that we do not rely on stochastic estimates of the curvature, which can be problematic in practice for second-order optimization methods.

Relationship to Zhu et al. (2024) Some of the aims of this paper overlap with those of the concurrent work in Zhu et al. (2024): both papers use the ARG for fast linear algebra to fit variance components of complex traits. Zhu et al. (2024) goes much further than we do in application, including analysis of many real traits in an ARG derived from real data, while we study the generative model (ARG-LMM), focus on algorithmic and theoretical improvements, and test these in two simulations with variance component estimation (AI-REML) and genetic prediction (BLUP). One difference is that we perform exact computations with the branch GRM, while Zhu et al. (2024) use a Monte-Carlo approach to randomly drop mutations on branch (edge) area. As shown in Zhu et al. (2024) (as well as in Fan, Mancuso, and Chiang, 2022 and Zhang et al., 2023), this provides highly accurate estimates as long as the number of mutations is large enough. However, there may be a meaningful accuracy-speed tradeoff (see Figure S2 of Zhu et al., 2024). Both Zhu et al. (2024) and `tslmm` implement the HE method, but we also implement REML; the latter is more accurate, but may not scale as well to large numbers of genomic windows. To compute traces, Zhu et al. (2024) use the Hutchinson estimator that has an error with variance $1/n$ (where n is the number of random vectors used; Hutchinson, 1990); we use a newer estimator that has variance $1/n^2$ (XTrace; Epperly, Tropp, and Webber, 2024), and so in practice needs many fewer test vectors. Unfortunately, the software accompanying Zhu et al. (2024) is not available at the time of writing, so we have not been able to compare the software implementations.

Future directions Rebollo et al. (2025) and Mafra Fortuna et al. (2025) study the estimation of haplotype and mutation effects on local trees. The machinery presented in this paper generalizes their tree-by-tree approach, allowing to compute any linear combination of sub-edge effects at scale. Such application includes time-resolved association testing (Link et al., 2023; Zhang et al., 2023; Zhu et al., 2024) and estimating ancestral genetic values (Edge and Coop, 2018; Peng, Mulder, and Edge, 2025). A slight modification of the GRM-vector multiplication algorithm is required to implement these applications, so we leave them to future work.

In closing The best way to model and predict complex traits doubtless depends on many factors, including the type of data available and the evolutionary forces on the trait. The results we have shown here demonstrate two of the primary advantages to working with ARGs: (1) they provide a "time axis", which leads to more interpretable results, and (2) they are extremely compact, so allow easy computation with large datasets.

7 Data availability

`tslmm` is a Python package available at `https://github.com/hanbin973/tslmm`. The results and the figures of the paper can be reproduced using the scripts at `https://github.com/hanbin973/tslmm_paper`.

8 Acknowledgments

Many thanks to Roshni Patel for explaining to us some aspects of heritability, to Doc Edge for discussion about an early idea of the paper, and to Yan Wong and Jonathan Terhorst for helpful comments and suggestions on a draft. We also thank the participants of the Probabilistic Modeling in Genomics 2025 conference for valuable questions and discussions. This research was supported in part through computational resources and services provided by Advanced Research Computing at the University of Michigan, Ann Arbor.

9 Funding

HL acknowledges the support from the Department of Statistics, University of Michigan, Ann Arbor through the Departmental Fellowship. NP, JK, and PR acknowledge support from the NIH NHGRI (research grant HG012473). GG acknowledges support from the BBSRC ISP grant to The Roslin Institute (BBS/E/D/30002275, BBS/E/RL/230001A, and BBS/E/RL/230001C), BBSRC research grant BB/T014067/1, and the NRC research grant 346741.

10 Conflicts of interest

The author(s) declare no conflicts of interest.

References

Anderson-Trocmé, Luke et al. (2023). "On the genes, genealogies, and geographies of Quebec". In: *Science* 380.6647, pp. 849–855.

Ané, Cécile and Michael Sanderson (Feb. 2005). "Missing the forest for the trees: phylogenetic compression and its implications for inferring complex evolutionary histories". In: *Systematic biology* 54.1, pp. 146–157. ISSN: 10635157. DOI: 10.1080/10635150590905984. URL: https://pubmed.ncbi.nlm.nih.gov/15805016.

Barton, N H, A M Etheridge, and A Véber (Dec. 2017). "The infinitesimal model: Definition, derivation, and implications". In: *Theor Popul Biol* 118, pp. 50–73. DOI: 10.1016/j.tpb.2017.06.001. URL: https://www.ncbi.nlm.nih.gov/pubmed/28709925.

Baumdicker, Franz et al. (2022). "Efficient ancestry and mutation simulation with msprime 1.0". In: *Genetics* 220.3, iyab229.

Boyd, Stephen and Lieven Vandenberghe (2004). *Convex optimization*. Cambridge university press.

Brandt, Débora Y C et al. (2024). "The Promise of Inferring the Past using the Ancestral Recombination Graph (ARG)". In: *Genome Biology and Evolution*, evae005. DOI: 10.1093/gbe/evae005. URL: https://doi.org/10.1093/gbe/evae005.

Butler, D G et al. (2023). *ASReml-R Reference Manual Version 4.2*. URL: https://vsni.co.uk.

Campos, Gustavo de los, Daniel Sorensen, and Daniel Gianola (2015). "Genomic Heritability: What Is It?" In: *PLOS Genetics* 11.5, pp. 1–21. DOI: 10.1371/journal.pgen.1005048. URL: https://doi.org/10.1371/journal.pgen.1005048.

Chevy, Elizabeth T et al. (2025). "Population genetics meets ecology: A guide to individual-based simulations in continuous landscapes". In: *Ecology and Evolution* 15.4, e71098.

Colleau, Jean-Jacques (2002). "An indirect approach to the extensive calculation of relationship coefficients". In: *Genetics Selection Evolution* 34.409. DOI: 10.1186/1297-9686-34-4-409. URL: https://doi.org/10.1186/1297-9686-34-4-409.

Cruden, Dorothy (1949). "The computation of inbreeding coefficients: for closed populations". In: *Journal of heredity* 40.9, pp. 248–251. DOI: 10.1093/oxfordjournals.jhered.a106039. URL: https://doi.org/10.1093/oxfordjournals.jhered.a106039.

DeHaas, Drew, Ziqing Pan, and Xinzhu Wei (Dec. 2024). "Enabling efficient analysis of biobank-scale data with genotype representation graphs". In: *Nature Computational Science*. ISSN: 2662-8457. DOI: 10.1038/s43588-024-00739-9. URL: https://doi.org/10.1038/s43588-024-00739-9.

Deng, Yun, Rasmus Nielsen, and Yun S. Song (Mar. 2024). "Robust and Accurate Bayesian Inference of Genome-Wide Genealogies for Large Samples". In: *bioRxiv*. DOI: 10.1101/2024.03.16.585351. URL: http://dx.doi.org/10.1101/2024.03.16.585351.

Durrett, Rick (2008). *Probability Models for DNA Sequence Evolution*. 2nd. New York, NY: Springer. ISBN: 978-0387781686.

Edge, Michael D and Graham Coop (Nov. 2018). "Reconstructing the History of Polygenic Scores Using Coalescent Trees". In: *Genetics* 211.1, pp. 235–262. ISSN: 1943-2631. DOI: 10.1534/genetics.118.301687. URL: http://dx.doi.org/10.1534/genetics.118.301687.

Emik, L. Otis and Clair E. Terrill (1949). "Systematic procedures for calculating inbreeding coefficients". In: *Journal of Heredity* 40.2, pp. 51–55. DOI: 10.1093/oxfordjournals.jhered.a105986. URL: https://doi.org/10.1093/oxfordjournals.jhered.a105986.

Epperly, Ethan N., Joel A. Tropp, and Robert J. Webber (Jan. 2024). "XTrace: Making the Most of Every Sample in Stochastic Trace Estimation". In: *SIAM Journal on Matrix Analysis and Applications* 45.1, pp. 1–23. ISSN: 1095-7162. DOI: 10.1137/23m1548323. URL: http://dx.doi.org/10.1137/23M1548323.

Ewens, Warren J (Jan. 2004). *Mathematical population genetics 1*. en. 2nd ed. Interdisciplinary Applied Mathematics. New York, NY: Springer.

Falconer, Douglas S. (1995). *Introduction to Quantitative Genetics*. 4th. Longman, pp. 264–266.

Fan, Caoqi, Nicholas Mancuso, and Charleston W. K. Chiang (2022). "A genealogical estimate of genetic relationships". In: *The American Journal of Human Genetics* 109.5, pp. 812–824. DOI: 10.1016/j.ajhg.2022.03.016. URL: https://www.sciencedirect.com/science/article/pii/S0002929722001124.

Feldmann, Mitchell J, Giovanny Covarrubias-Pazaran, and Hans-Peter Piepho (2023). "Complex traits and candidate genes: estimation of genetic variance components across multiple genetic architectures". In: *G3: Genes, Genomes, Genetics* 13.9. DOI: 10.1093/g3journal/jkad148. URL: http://dx.doi.org/10.1093/g3journal/jkad148.

Felsenstein, J (Sept. 1973). "Maximum-likelihood estimation of evolutionary trees from continuous characters". In: *Am J Hum Genet* 25.5, pp. 471–492. URL: http://www.ncbi.nlm.nih.gov/pmc/articles/PMC1762641/.

Fisher, Ronald A. (1918). "The correlation between relatives on the supposition of Mendelian inheritance". In: *Earth and Environmental Science Transactions of the Royal Society of Edinburgh* 52.2, pp. 399–433. DOI: 10.1017/S0080456800012163. URL: https://doi.org/10.1017/S0080456800012163.

Frangella, Zachary, Joel A. Tropp, and Madeleine Udell (2021). *Randomized Nyström Preconditioning*. arXiv: 2110.02820 [math.NA]. URL: https://arxiv.org/abs/2110.02820.

Furnas, George W. and Jeff Zacks (1994). "Multitrees: enriching and reusing hierarchical structure". In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. CHI '94. Boston, Massachusetts, USA: Association for Computing Machinery, pp. 330–336. ISBN: 0897916506. DOI: 10.1145/191666.191778. URL: https://doi.org/10.1145/191666.191778.

Gianola, Daniel et al. (2009). "Additive Genetic Variability and the Bayesian Alphabet". In: *Genetics* 183.1, pp. 347–363. DOI: 10.1534/genetics.109.103952. URL: http://dx.doi.org/10.1534/genetics.109.103952.

Gilmour, Arthur R., Robin Thompson, and Brian R. Cullis (Dec. 1995). "Average Information REML: An Efficient Algorithm for Variance Parameter Estimation in Linear Mixed Models". In: *Biometrics* 51.4, p. 1440. ISSN: 0006-341X. DOI: 10.2307/2533274. URL: http://dx.doi.org/10.2307/2533274.

Griffiths, Robert C. and Paul Marjoram (1997). "An ancestral recombination graph". In: *Progress in population genetics and human evolution (Minneapolis, MN, 1994)*. Vol. 87. IMA Vol. Math. Appl. New York: Springer, pp. 257–270. URL: http://www.math.canterbury.ac.nz/~r.sainudiin/recomb/ima.pdf.

Gunnarsson, Árni Freyr et al. (2024). "A scalable approach for genome-wide inference of ancestral recombination graphs". In: *bioRxiv*. DOI: 10.1101/2024.08.31.610248. eprint: https://www.biorxiv.org/content/early/2024/09/02/2024.08.31.610248.full.pdf. URL: https://www.biorxiv.org/content/early/2024/09/02/2024.08.31.610248.

Haller, Benjamin C and Philipp W Messer (2023). "SLiM 4: multispecies eco-evolutionary modeling". In: *The American Naturalist* 201.5, E127–E139.

Harville, David A. (1977a). "Maximum Likelihood Approaches to Variance Component Estimation and to Related Problems". In: *Journal of the American Statistical Association* 72.358, pp. 320–338. DOI: 10.1080/01621459.1977.10480998. URL: http://dx.doi.org/10.1080/01621459.1977.10480998.

— (June 1977b). "Maximum Likelihood Approaches to Variance Component Estimation and to Related Problems". In: *Journal of the American Statistical Association* 72.358, pp. 320–338. ISSN: 1537-274X. DOI: 10.1080/01621459.1977.10480998. URL: http://dx.doi.org/10.1080/01621459.1977.10480998.

Haseman, J. K. and R. C. Elston (1972). "The investigation of linkage between a quantitative trait and a marker locus". In: *Behavior Genetics* 2.1, pp. 3–19. DOI: 10.1007/bf01066731. URL: http://dx.doi.org/10.1007/BF01066731.

Heaton, Matthew J. et al. (2018). "A Case Study Competition Among Methods for Analyzing Large Spatial Data". In: *Journal of Agricultural, Biological and Environmental Statistics* 24.3, pp. 398–425. DOI: 10.1007/s13253-018-00348-w. URL: http://dx.doi.org/10.1007/s13253-018-00348-w.

Henderson, C. R. (1984). *Applications of Linear Models in Animal Breeding*. Guelph, Ontario: University of Guelph.

— (Nov. 1975). "Use of Relationships Among Sires to Increase Accuracy of Sire Evaluation". In: *Journal of Dairy Science* 58.11, pp. 1731–1738. ISSN: 0022-0302. DOI: 10.3168/jds.s0022-0302(75)84777-1. URL: http://dx.doi.org/10.3168/jds.S0022-0302(75)84777-1.

Henderson, Charles R. (1976). "A Simple Method for Computing the Inverse of a Numerator Relationship Matrix Used in Prediction of Breeding Values". In: *Biometrics* 32.1, pp. 69–83. DOI: 10.2307/2529339. URL: https://www.jstor.org/stable/2529339.

Hestenes, Magnus R and Eduard Stiefel (1952). "Methods of conjugate gradients for solving linear systems". In: *Journal of research of the National Bureau of Standards* 49.6, pp. 409–436.

Hou, Kangcheng et al. (2019). "Accurate estimation of SNP-heritability from biobank-scale data irrespective of genetic architecture". In: *Nature Genetics* 51.8, pp. 1244–1251. DOI: 10.1038/s41588-019-0465-0. URL: http://dx.doi.org/10.1038/s41588-019-0465-0.

Hudson, R.R. (1990). "Gene genealogies and the coalescent process". In: *Oxford surveys in evolutionary biology* 7.1, p. 44. URL: http://web.eve.ucdavis.edu/pbg298/pdfs/Hudson_OxfordSurveysEvolBiol_1991.pdf.

Hutchinson, M.F. (1990). "A stochastic estimator of the trace of the influence matrix for Laplacian smoothing splines". In: *Communications in Statistics - Simulation and Computation* 19.2, pp. 433–450. DOI: 10.1080/03610919008812866. URL: http://dx.doi.org/10.1080/03610919008812866.

Isserlis, L. (Nov. 1918). "On a Formula for the Product-Moment Coefficient of any Order of a Normal Frequency Distribution in any Number of Variables". In: *Biometrika* 12.1/2, p. 134. ISSN: 0006-3444. DOI: 10.2307/2331932. URL: http://dx.doi.org/10.2307/2331932.

Kelleher, Jerome, Alison M Etheridge, and Gilean McVean (May 2016). "Efficient Coalescent Simulation and Genealogical Analysis for Large Sample Sizes". In: *PLOS Computational Biology* 12.5. Ed. by Yun S. Song, e1004842. ISSN: 1553-7358. DOI: 10.1371/journal.pcbi.1004842. URL: http://dx.doi.org/10.1371/journal.pcbi.1004842.

Kelleher, Jerome et al. (Sept. 2019). "Inferring whole-genome histories in large population datasets". In: *Nature Genetics* 51.9, pp. 1330–1338. ISSN: 1546-1718. DOI: 10.1038/s41588-019-0483-y. URL: http://dx.doi.org/10.1038/s41588-019-0483-y.

Kennedy, BW, LR Schaeffer, and DA Sorensen (1988). "Genetic properties of animal models". In: *Journal of Dairy science* 71.Supplement 2, pp. 17–26. DOI: 10.1016/S0022-0302(88)79975-0. URL: https://www.sciencedirect.com/science/article/pii/S0022030288799750.

Kimura, Motoo (1969). "The number of heterozygous nucleotide sites maintained in a finite population due to steady flux of mutations". In: *Genetics* 61.4, pp. 893–903. DOI: 10.1093/genetics/61.4.893. URL: http://dx.doi.org/10.1093/genetics/61.4.893.

Kimura, Motoo and James F Crow (Apr. 1964). "The number of alleles that can be maintained a finite population." In: *Genetics* 49.4, pp. 725–738. ISSN: 1943-2631. DOI: 10.1093/genetics/49.4.725. URL: http://dx.doi.org/10.1093/genetics/49.4.725.

Lara, Letícia A. de C. et al. (Jan. 2022). "Temporal and genomic analysis of additive genetic variance in breeding programmes". In: *Heredity* 128.1, pp. 21–32. ISSN: 1365-2540. DOI: 10.1038/s41437-021-00485-y. URL: https://doi.org/10.1038/s41437-021-00485-y.

Larribe, Fabrice, Sabin Lessard, and Nicholas J Schork (Sept. 2002). "Gene mapping via the ancestral recombination graph". In: *Theoretical population biology* 62.2, pp. 215–229. ISSN: 00405809. DOI: 10.1006/tpbi.2002.1601. URL: https://pubmed.ncbi.nlm.nih.gov/12167358.

Lehmann, Brieuc et al. (Mar. 2025). "On ARGs, pedigrees, and genetic relatedness matrices". In: *bioRxiv*. DOI: 10.1101/2025.03.03.641310. URL: http://dx.doi.org/10.1101/2025.03.03.641310.

Lewanski, Alexander L., Michael C. Grundler, and Gideon S. Bradburd (2024). "The era of the ARG: An introduction to ancestral recombination graphs and their significance in empirical evolutionary genomics". In: *PLOS Genetics* 20.1, pp. 1–24. DOI: 10.1371/journal.pgen.1011110. URL: https://doi.org/10.1371/journal.pgen.1011110.

Link, Vivian et al. (2023). "Tree-based QTL mapping with expected local genetic relatedness matrices". In: *The American Journal of Human Genetics* 110.12, pp. 2077–2091. DOI: 10.1016/j.ajhg.2023.10.017. URL: https://www.sciencedirect.com/science/article/pii/S0002929723003956.

Loya, Hrushikesh et al. (Jan. 2025). "A scalable variational inference approach for increased mixed-model association power". In: *Nature Genetics* 57.2, pp. 461–468. ISSN: 1546-1718. DOI: 10.1038/s41588-024-02044-7. URL: http://dx.doi.org/10.1038/s41588-024-02044-7.

Lynch, Michael and William G. Hill (Sept. 1986). "Phenotypic evolution by neutral mutation". In: *Evolution* 40.5, pp. 915–935. ISSN: 1558-5646. DOI: 10.1111/j.1558-5646.1986.tb00561.x. URL: http://dx.doi.org/10.1111/j.1558-5646.1986.tb00561.x.

Lynch, Michael and Bruce Walsh (Jan. 1998). *Genetics and analysis of quantitative traits*. en. New York, NY: Oxford University Press.

Mafra Fortuna, Gabriela et al. (2025). "Estimating haplotype values and mutation effects in the context of a local DNA tree". In: *bioRxiv*. DOI: 10.1101/2025.11.08.687335. URL: https://www.biorxiv.org/content/early/2025/11/08/2025.11.08.687335.

Meuwissen, T. H. E., B. J. Hayes, and M. E. Goddard (2001). "Prediction of Total Genetic Value Using Genome-Wide Dense Marker Maps". In: *Genetics* 157.4, pp. 1819–1829. DOI: 10.1093/genetics/157.4.1819. URL: https://dx.doi.org/10.1093/genetics/157.4.1819.

Meuwissen, Theodorus H. E. and Z. Luo (1992). "Computing inbreeding coefficients in large populations". In: *Genetics Selection Evolution* 24.4, pp. 305–313. DOI: 10.1186/1297-9686-24-4-305. URL: https://doi.org/10.1186/1297-9686-24-4-305.

Minichiello, Mark J. and Richard Durbin (2006). "Mapping Trait Loci by Use of Inferred Ancestral Recombination Graphs". In: *The American Journal of Human Genetics* 79.5, pp. 910–922. ISSN: 0002-9297. DOI: https://doi.org/10.1086/508901. URL: http://www.sciencedirect.com/science/article/pii/S0002929707608349.

Misztal, Ignacy (2015). "Inexpensive Computation of the Inverse of the Genomic Relationship Matrix in Populations with Small Effective Population Size". In: *Genetics* 202.2, pp. 401–409. DOI: 10.1534/genetics.115.182089. URL: http://dx.doi.org/10.1534/genetics.115.182089.

Mrode, Raphael and Ivan Pocrnic (2023). *Linear Models for the Prediction of the Genetic Merit of Animals*. Wallingford, UK: CABI, p. 412.

Patterson, H. D. and R. Thompson (1971). "Recovery of inter-block information when block sizes are unequal". In: *Biometrika* 58.3, pp. 545–554. DOI: 10.1093/biomet/58.3.545. URL: http://dx.doi.org/10.1093/biomet/58.3.545.

Pazokitoroudi, Ali et al. (Aug. 2020). "Efficient variance components analysis across millions of genomes". In: *Nature Communications* 11.1. ISSN: 2041-1723. DOI: 10.1038/s41467-020-17576-9. URL: http://dx.doi.org/10.1038/s41467-020-17576-9.

Peng, Dandan, Obadiah J Mulder, and Michael D Edge (Mar. 2025). "Evaluating ARG-estimation methods in the context of estimating population-mean polygenic score histories". In: *GENETICS* 229.4. Ed. by S Gravel. ISSN: 1943-2631. DOI: 10.1093/genetics/iyaf033. URL: http://dx.doi.org/10.1093/genetics/iyaf033.

Pope, Nathaniel S. et al. (2023). "The expansion of agriculture has shaped the recent evolutionary history of a specialized squash pollinator". In: *Proceedings of the National Academy of Sciences* 120.15, e2208116120. DOI: 10.1073/pnas.2208116120. eprint: https://www.pnas.org/doi/pdf/10.1073/pnas.2208116120. URL: https://www.pnas.org/doi/abs/10.1073/pnas.2208116120.

Quaas, Richard L. (1988). "Additive genetic model with groups and relationships". In: *Journal of Dairy Science* 71.5, pp. 1338–1345. DOI: 10.3168/jds.S0022-0302(88)79691-5. URL: https://www.sciencedirect.com/science/article/pii/S0022030288796915.

Ralph, Peter (June 2019). "An empirical approach to demographic inference with genomic data". In: *Theoretical Population Biology* 127, pp. 91–101. ISSN: 0040-5809. DOI: 10.1016/j.tpb.2019.03.005. URL: http://dx.doi.org/10.1016/j.tpb.2019.03.005.

Ralph, Peter, Kevin Thornton, and Jerome Kelleher (2020). "Efficiently Summarizing Relationships in Large Samples: A General Duality Between Statistics of Genealogies and Genomes". In: *Genetics* 215.3, pp. 779–797. DOI: 10.1534/genetics.120.303253. URL: https://doi.org/10.1534/genetics.120.303253.

Rasmussen, Matthew D. et al. (May 2014). "Genome-Wide Inference of Ancestral Recombination Graphs". In: *PLoS Genetics* 10.5. Ed. by Graham Coop, e1004342. ISSN: 1553-7404. DOI: 10.1371/journal.pgen.1004342. URL: http://dx.doi.org/10.1371/journal.pgen.1004342.

Rebollo, I et al. (2025). "Leveraging ancestral recombination graphs for quantitative genetic analysis of rice yield in indica and japonica subspecies". In: *bioRxiv*. DOI: 10.1101/2025.01.14.633033. URL: https://doi.org/10.1101/2025.01.14.633033.

Robinson, G. K. (1991). "That BLUP is a Good Thing: The Estimation of Random Effects". In: *Statistical Science* 6.1, pp. 15–32. DOI: 10.1214/ss/1177011926. URL: http://dx.doi.org/10.1214/ss/1177011926.

Rue, H. and L. Held (2005). *Gaussian Markov Random Fields: Theory and Applications*. Vol. 104. Monographs on Statistics and Applied Probability. London: Chapman & Hall. URL: http://dx.doi.org/10.1201/9780203492024.

Salehi Nowbandegani, Pouria et al. (2023). "Extremely sparse models of linkage disequilibrium in ancestrally diverse association studies". In: *Nature Genetics* 55.9, pp. 1494–1502.

Schreck, Nicholas, Hans-Peter Piepho, and Martin Schlather (2019). "Best Prediction of the Additive Genomic Variance in Random-Effects Models". In: *Genetics* 213.2, pp. 379–394. DOI: 10.1534/genetics.119.302324. URL: http://dx.doi.org/10.1534/genetics.119.302324.

Selle, M L et al. (2021). "Hierarchical modeling of haplotype effects on a phylogeny". In: *Frontiers in Genetics* 11, p. 1774. DOI: 10.3389/fgene.2020.531218.

Sorensen, Daniel and Daniel Gianola (2002). *Likelihood, Bayesian, and MCMC Methods in Quantitative Genetics*. Statistics for Biology and Health. New York: Springer.

Speed, Doug and David J. Balding (2015). "Relatedness in the post-genomic era: is it still useful?" In: *Nature Reviews Genetics* 16.1, pp. 33–44. DOI: 10.1038/nrg3821. URL: https://www.nature.com/articles/nrg3821.

Speed, Doug et al. (Dec. 2012). "Improved Heritability Estimation from Genome-wide SNPs". In: *The American Journal of Human Genetics* 91.6, pp. 1011–1021. ISSN: 0002-9297. DOI: 10.1016/j.ajhg.2012.10.010. URL: http://dx.doi.org/10.1016/j.ajhg.2012.10.010.

Speidel, Leo et al. (2019). "A method for genome-wide genealogy estimation for thousands of samples". In: *Nature Genetics* 51.9, pp. 1321–1329. DOI: 10.1038/s41588-019-0484-x. URL: http://dx.doi.org/10.1038/s41588-019-0484-x.

Stern, Aaron J et al. (Feb. 2021). "Disentangling selection on genetically correlated polygenic traits via whole-genome genealogies". In: *American Journal of Human Genetics* 108.2, pp. 219–239. ISSN: 00029297. DOI: 10.1016/j.ajhg.2020.12.005. URL: https://pubmed.ncbi.nlm.nih.gov/33440170.

Strandén, I. and D. J. Garrick (2009). "Derivation of equivalent computing algorithms for genomic predictions and reliabilities of animal merit". In: *Journal of Dairy Science* 92.6, pp. 2971–2975. DOI: 10.3168/jds.2008-1929. URL: https://dx.doi.org/10.3168/jds.2008-1929.

Tagami, Daiki, Gertjan Bisschop, and Jerome Kelleher (May 2024). "tstrait: a quantitative trait simulator for ancestral recombination graphs". In: *Bioinformatics* 40.6. Ed. by Russell Schwartz. ISSN: 1367-4811. DOI: 10.1093/bioinformatics/btae334. URL: http://dx.doi.org/10.1093/bioinformatics/btae334.

Thompson, E.A. (1975). *Human Evolutionary Trees*. Cambridge University Press. ISBN: 9780521099455.

VanRaden, P M (2008). "Efficient Methods to Compute Genomic Predictions". In: *Journal of Dairy Science* 91.11, pp. 4414–4423. DOI: 10.3168/jds.2007-0980. URL: http://linkinghub.elsevier.com/retrieve/pii/S0022030208709901.

Vazquez, Ana Ines et al. (Dec. 2018). *pedigreeTools: Versatile Functions for Working with Pedigrees*. DOI: 10.32614/cran.package.pedigreetools. URL: http://dx.doi.org/10.32614/CRAN.package.pedigreeTools.

Wakeley, John (Aug. 2008). *Coalescent Theory*. en. Greenwood Village, CO: Roberts & Company.

Watterson, G.A. (1975). "On the number of segregating sites in genetical models without recombination". In: *Theoretical Population Biology* 7.2, pp. 256–276. DOI: 10.1016/0040-5809(75)90020-9. URL: http://dx.doi.org/10.1016/0040-5809(75)90020-9.

Westfall, Peter H. (Sept. 1987). "A Comparison of Variance Component Estimates for Arbitrary Underlying Distributions". In: *Journal of the American Statistical Association* 82.399, pp. 866–874. ISSN: 1537-274X. DOI: 10.1080/01621459.1987.10478510. URL: http://dx.doi.org/10.1080/01621459.1987.10478510.

Whittaker, J. C., C. S. Haley, and R. Thompson (1997). "Optimal weighting of information in marker-assisted selection". In: *Genetical Research* 69.2, pp. 137–144. DOI: 10.1017/S0016672397002711.

Wohns, Anthony Wilder et al. (Feb. 2022). "A unified genealogy of modern and ancient genomes". In: *Science* 375.6583. ISSN: 1095-9203. DOI: 10.1126/science.abi8264. URL: http://dx.doi.org/10.1126/science.abi8264.

Wong, Yan et al. (July 2024). "A general and efficient representation of ancestral recombination graphs". In: *GENETICS* 228.1. Ed. by G Coop. ISSN: 1943-2631. DOI: 10.1093/genetics/iyae100. URL: http://dx.doi.org/10.1093/genetics/iyae100.

Wray, Naomi R et al. (Feb. 2019). "Complex Trait Prediction from Genome Data: Contrasting EBV in Livestock to PRS in Humans". In: *Genetics* 211.4, pp. 1131–1141. ISSN: 1943-2631. DOI: 10.1534/genetics.119.301859. URL: http://dx.doi.org/10.1534/genetics.119.301859.

Wright, Sewall (1922). "Coefficients of Inbreeding and Relationship". In: *The American Naturalist* 56.645, pp. 330–338. DOI: 10.1086/279872. URL: https://doi.org/10.1086/279872.

Wu, Yue and Sriram Sankararaman (June 2018). "A scalable estimator of SNP heritability for biobank-scale data". In: *Bioinformatics* 34.13, pp. i187–i194. ISSN: 1367-4811. DOI: 10.1093/bioinformatics/bty253. URL: http://dx.doi.org/10.1093/bioinformatics/bty253.

Yang, Jian et al. (June 2010). "Common SNPs explain a large proportion of the heritability for human height". In: *Nature Genetics* 42.7, pp. 565–569. ISSN: 1546-1718. DOI: 10.1038/ng.608. URL: http://dx.doi.org/10.1038/ng.608.

Yang, Jian et al. (Jan. 2011). "GCTA: A Tool for Genome-wide Complex Trait Analysis". In: *The American Journal of Human Genetics* 88.1, pp. 76–82. ISSN: 0002-9297. DOI: 10.1016/j.ajhg.2010.11.011. URL: http://dx.doi.org/10.1016/j.ajhg.2010.11.011.

Zhang, Brian C. et al. (2023). "Biobank-scale inference of ancestral recombination graphs enables genealogical analysis of complex traits". In: *Nature Genetics* 55.5, pp. 768–776. DOI: 10.1038/s41588-023-01379-x. URL: https://doi.org/10.1038/s41588-023-01379-x.

Zhou, Xiang (Dec. 2017). "A unified framework for variance component estimation with summary statistics in genome-wide association studies". In: *The Annals of Applied Statistics* 11.4. ISSN: 1932-6157. DOI: 10.1214/17-aoas1052. URL: http://dx.doi.org/10.1214/17-AOAS1052.

Zhu, Jiazheng et al. (Aug. 2024). "Fast variance component analysis using large-scale ancestral recombination graphs". In: *BioRxiv*. DOI: 10.1101/2024.08.31.610262. URL: http://dx.doi.org/10.1101/2024.08.31.610262.

A Model formulation and assumptions

A.1 Expressing a complex trait on an ARG

We show the correspondence between site and branch (edge) based trait expressions. We define an ARG $\mathcal{T} = (\mathcal{N}, \mathcal{E})$ as a collection of nodes $\mathcal{N}$ and edges $\mathcal{E}$. Each node n has a (birth) time ago t_n, and each edge e indicates inheritance of a genomic interval $[\ell_e, r_e)$ by a child $c(e)$ from a parent $p(e)$ (Wong et al., 2024). We call these "parent" and "child" in graph terms, so the inheritance might be over more than one generation. To describe a valid ARG, the spans of all edges with the same child must be disjoint, and the time of the parent must be larger than the time of the child for each edge. We will also call the *span* of an edge $s_e = r_e - \ell_e$ (in number of base-pairs), and the *length* of an edge $l_e = t_{p(e)} - t_{c(e)}$ (in time units, such as generations). All expectations and variances are conditioned on $\mathcal{T}$, which accounts for the variation in observed genome data due to recombination and drift, and hence also the implicit population structure and pedigree of individuals in $\mathcal{T}$. Only the mutational process is left random in this setting. This setting was explored in Ralph (2019) and expanded in Ralph, Thornton, and Kelleher (2020). In those papers, the utility of treating mutations as random on a fixed ARG was to understand how well a given SNP genotype (site) statistic could be expected to estimate the corresponding summary of tree shapes.

For conceptual clarity, we assume in the following that the edges are in fact *sub-edges*: each has a unique sub-topology along its span. (Although this is not true in the case of tree sequences in general, we can enforce this by appropriate splitting of edges as in Salehi Nowbandegani et al., 2023.) Nevertheless, this splitting is merely conceptual and never takes place in the algorithms we'll demonstrate shortly, as it significantly degrades the storage efficiency of the tree sequence data structure.

We consider complex traits that follow a linear model. Write $\mathbf{G} \in \mathbb{R}^{N \times P}$ for the $N \times P$ genotype matrix for N genomes and P sites in the genome, whose $(i, j)^{\text{th}}$ entry is the number of derived alleles carried by the ith genome at the jth site. Then the trait model is:

$$\mathbf{y} = \mathbf{G}\boldsymbol{\beta} + \boldsymbol{\varepsilon} = \sum_{p=1}^{P} \mathbf{G}_p \boldsymbol{\beta}_p + \boldsymbol{\varepsilon}, \tag{A1}$$

where $\boldsymbol{\beta} \in \mathbb{R}^P$ is a vector defining a effect size for the derived allele(s) at each site. For now, we condition on $\boldsymbol{\beta}$ so that we treat it as a fixed quantity. With infinite-sites mutation model, each site has at most two alleles, so this effect is the allele substitution effect (Falconer, 1995), also called the site effect, variant effect, or mutation effect in this work. One cannot estimate $\boldsymbol{\beta}$'s entries separately because some variants appear on the same edge (or on the two edges pendant to the root). Columns of $\mathbf{G}$ corresponding to these variants will be identical, leading to perfect collinearity (linkage-disequilibrium), making them statistically indistinguishable. We will group these indiscernible variants according to their edge membership. After conditioning on $\mathcal{T}$, the remaining variation in $\mathbf{G}$ is solely due to the mutational process. We assume that mutations occur on edge e following a Poisson process at a rate proportional to the total number of base-pairs (span) multiplied by the amount of time the edge represents (length), $l_e s_e$. We let the per unit time per base-pair mutation rate u_{ep} vary across edges and positions If we let $\mathbf{1}_{ep}$ be an indicator variable that is 1 if a mutation has arrived on edge e at position p and 0 otherwise, $\mathbb{E}\left[\mathbf{1}_{ep}\right] = l_e u_{ep}$.

Recurrent mutations are allowed; however, mutations cannot occur below an existing one. This is a slight relaxation of the infinite-sites model to allow multiple recurrent mutations in unrelated lineages but no back mutation. Define $\mathbf{Z} \in \mathbb{R}^{N \times E}$ so that $\mathbf{Z}_{ie}$ is the number of chromosomes of individual i that is a descendant

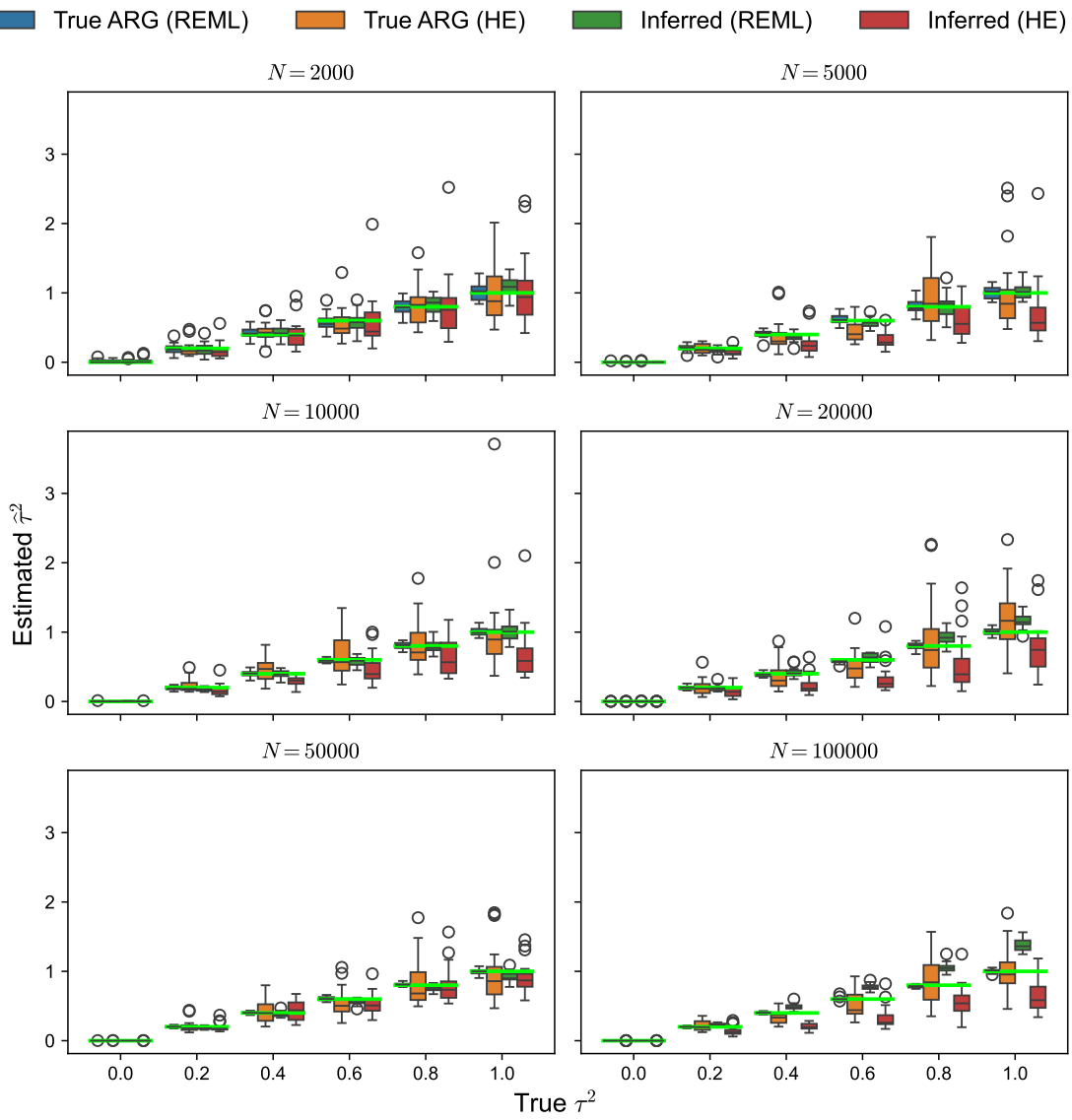

Figure A1: Statistical precision of the HE method and AI-REML. By default, the branch GRM is scaled by 1 over $4\sum_T s_T t_{\mathrm{root},T}$. The values in the labels "HE (value)" refer to the additional scaling we apply to the GRM. (Top) The distribution of estimators over 100 replicates. (Bottom) The mean squared error (MSE) of the estimators with respect to the true parameters $\sigma_\epsilon^2 = 1$ and $\tau^2 = 1$.

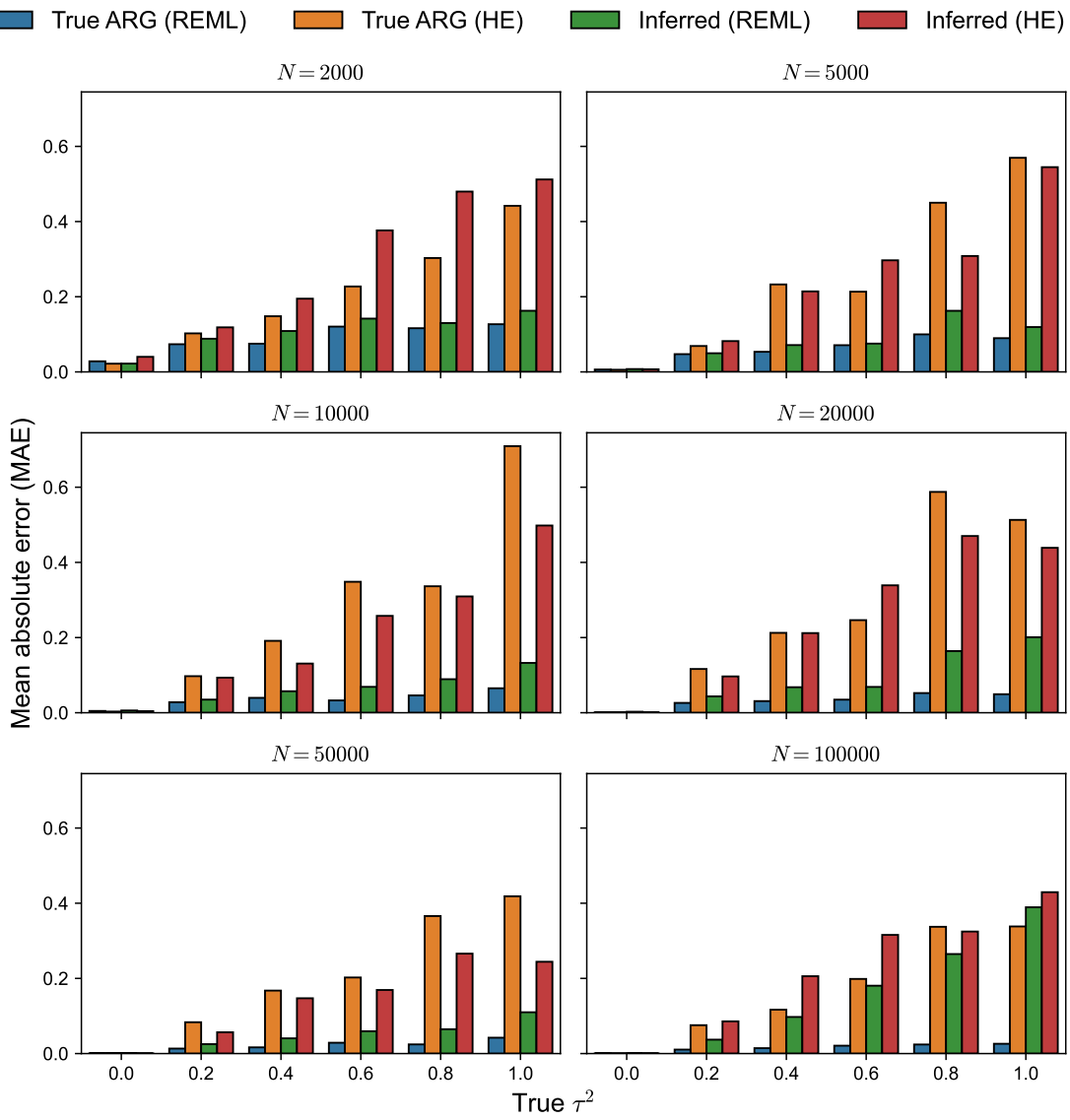

Figure A2: Statistical precision of the HE method and AI-REML. By default, the branch GRM is scaled by 1 over $4\sum_T s_T t_{\mathrm{root},T}$. The values in the labels "HE (value)" refer to the additional scaling we apply to the GRM. (Top) The distribution of estimators over 100 replicates. (Bottom) The mean absolute error (MAE) of the estimators with respect to the true parameters $\sigma_\varepsilon^2 = 1$ and $\tau^2 = 1$.

of edge e. $\mathbf{Z}_{ie}$ ranges from 0 to 2 in the case of diploids. We can express $\mathbf{G}$ with $\mathbf{Z}$ by:

$$\mathbf{G}_{ip} = \sum_{e:p\in e} \mathbf{Z}_{ie}\mathbf{1}_{ep}, \tag{A2}$$

where $p \in e$ means edge e contains the position p, and define $\mathbf{G}_p$ as the p-th column of $\mathbf{G}$. In words, one can determine individual i's genotype at position p by first examining if edge e is ancestral to i ($\mathbf{Z}_{ie}$) and then checking if the edge has a mutation at p ($\mathbf{1}_{ep}$). Finally, we sum up such contributions across all edges ($\sum_{e:p\in e}$). This also illustrates how `tstrait` computes the trait values: for a position p in a local tree, it travels down the edges (or equivalently branches for now) of the tree and checks if there's a mutation along the path from the root to the tips. Here, $\mathbf{Z}_{ie}$ tells if an edge in $\{e : p \in e\}$ is on the path from the root to a tip, which is individual i.

Using equation (A2), we can collapse variants corresponding to perfectly correlated columns of $\mathbf{G}$ into the edge on which they reside. This is a natural choice because mutations on the same edge are inherited by the same set of individuals (recall that the edge's subtree is constant by assumption). Substituting (A2) in the linear equation (A1) gives:

$$\begin{aligned}
\mathbf{y} &= \sum_{p=1}^{P} \mathbf{G}_p\boldsymbol{\beta}_p + \boldsymbol{\varepsilon} \\
&= \sum_{p=1}^{P} \left(\sum_{e:p\in e} \mathbf{Z}_e\boldsymbol{\beta}_p\mathbf{1}_{ep} \right) + \boldsymbol{\varepsilon} \quad \because \text{substitute (A2)} \\
&= \sum_{e=1}^{E} \sum_{p:p\in e} \mathbf{Z}_e\boldsymbol{\beta}_p\mathbf{1}_{ep} + \boldsymbol{\varepsilon} \quad \because \text{swap the sums over } e \text{ and } p \\
&= \sum_{e=1}^{E} \mathbf{Z}_e \left(\sum_{p:p\in e} \boldsymbol{\beta}_p\mathbf{1}_{ep} \right) + \boldsymbol{\varepsilon} \quad \because \mathbf{Z}_e \text{ does not depend on } p \\
&= \sum_{e=1}^{E} \mathbf{Z}_e\mathbf{v}_e + \boldsymbol{\varepsilon} \quad \because \mathbf{v}_e = \sum_{p:p\in e} \boldsymbol{\beta}_p\mathbf{1}_{ep} \\
&= \mathbf{Z}\mathbf{v} + \boldsymbol{\varepsilon}.
\end{aligned} \tag{A3}$$

The key step is at the third line which swaps the summation over edges e and sites p. Then, we pulled out $\mathbf{Z}_e$ out of the inner sum over p and merged the inner sum into $\mathbf{v}_e$. This derivation presents a way to convert a genotype-based expression of a trait $\mathbf{y} = \sum_p \mathbf{G}_p\boldsymbol{\beta}_p + \boldsymbol{\varepsilon}$ into a branch-based expression $\mathbf{y} = \sum_e \mathbf{Z}_e\mathbf{v}_e + \boldsymbol{\varepsilon}$.

A.2 Linear algebra formulation

We now connect this linear model to classical formulations in quantitative genetics and selective breeding literature on the pedigree model (Henderson, 1976; Quaas, 1988). While we work with a linear model on an ARG, not pedigree, our aim here is to highlight the inherent sparse structure of phenotype generative models and the critical connection with sparse linear algebra and efficient computational algorithms (Rue and Held, 2005). We start by noting that sub-edges are also edges – each has a parent and child node and a genomic interval – but with the property (as above) that the subtree below them in the ARG does not change. These can be obtained as follows: iterating along the genome, each time the local tree changes, it changes through

addition or removal of an edge. If the parent of the newly added or removed edge is n, then we move up through the local tree, and split each edge we encounter. For instance, if n is not a root then there is an edge with child n and parent p, and interval $[\ell, r)$ that spans the current position x. By "splitting" this edge we mean to replace it with two edges: one with interval $[\ell, x)$ and the other with interval $[x, r)$. The resulting shorter edges will be all sub-edges.

We say that sub-edge e_1 *directly inherits* from sub-edge e_2 if $p(e_1) = c(e_2)$ and their genomic intervals overlap. In fact, it then follows that the interval of e_2 is contained within that of e_1: in other words, $[\ell(e_2), r(e_2)) \subseteq [\ell(e_1), r(e_1))$. Now, define the square matrix $\mathbf{A}$ to have rows and columns indexed by sub-edges, with $\mathbf{A}_{ef} = 1$ if sub-edge e directly inherits from f, and 0 otherwise. This $\mathbf{A}$ is $\mathbf{P}$ in Quaas (1988). The *sub-edge inheritance matrix* is then $\mathbf{I} + \mathbf{A} + \mathbf{A}^2 + \cdots = (\mathbf{I} - \mathbf{A})^{-1}$: this has $(e, f)^{\text{th}}$ entry equal to 1 if $e = f$ or if e inherits from f (i.e., if the edge represented by e carries genetic material inherited also along f), and 0 otherwise. Note that the seemingly infinite series above is in fact a finite sum because the depth of an ARG is finite. Now, write $\mathbf{M}$ for the *individual-sub-edge incidence matrix*: for an individual i and a sub-edge e, let $\mathbf{M}_{ie} = 1$ if $c(e) \in i$ and $\mathbf{M}_{ie} = 0$ otherwise. $c(e) \in i$ means individual i contains node $c(e)$. The matrix $\mathbf{Z}$ from the main text is thus:

$$\mathbf{Z} = \mathbf{M}(\mathbf{I} - \mathbf{A})^{-1}.$$

The directed graph whose adjacency matrix is $\mathbf{A}$, the *sub-edge graph*, is a multitree, and has many convenient properties. For instance, many standard ARG computations that require left-to-right iteration can be implemented as bottom-up or top-down on the sub-edge graph. This property is shared by the genome representation graph of DeHaas, Pan, and Wei (2024). Typically, there will be a multiplicative factor of order $\log N$ more sub-edges than edges, where N is the sample size. However, this difference can be large in practice as the constant depends on demographic history. An advantage to working with the sub-edge graph is that it allows use of more general graph-based tools, as for instance by Salehi Nowbandegani et al. (2023). However, we are not aware of a reason that such algorithms are necessarily more efficient.

This formulation is useful conceptually, but we did not find it to be useful computationally, because typical left-to-right algorithms on the ARG naturally work with sub-edges without explicitly constructing them (for instance, those in Ralph, Thornton, and Kelleher, 2020 or Zhu et al., 2024). We are not aware of any argument that algorithms using a structure like the sub-edge graph are inherently more efficient than algorithms that efficiently use the tree sequence.

Sub-edges, and the "splitting" procedure are very similar to the "bricks" and the "bricking" operation used by Salehi Nowbandegani et al. (2023). However, "bricks" are defined so that they have a unique set of descendant nodes, rather than a unique subtree. Since subtrees can change but still have the same set of descendants, bricks are thus larger than sub-edges, while retaining the useful properties above. However, the brick graph is no longer a multitree. We chose the more strict definition mainly because it does not require retaining and checking the descendant nodes at each step.

The linear algebra formulation shows the sparse structure of our trait generative model on an ARG, enabling scalable algorithms as shown in Lehmann et al. (2025) and this study. For completeness, we point to Misztal (2015), who followed a similar goal – enabling inexpensive computations with the inverse of genotype GRM by avoiding inversion of a large GRM. Instead of focusing on the trait generative model, Misztal (2015) split the genotyped sample into core and non-core sets of individuals and expressed genetic values of non-core individuals conditional on core individuals. This conditioning also admits a sparse inverse of genotype GRM and is technically the same as conditioning on "inducing point" or "knots" in spatial modeling (e.g., Heaton et al., 2018). While this core conditioning approach is computationally very efficient for populations with limited genetic diversity, such as in selective breeding, it does not scale as well to more

diverse settings as the work presented in this study.

A.3 The branch-based phenotype model is a linear mixed model

The last expression we arrived at expresses a phenotype in terms of edges:

$$\mathbf{y} = \sum_e \mathbf{Z}_e \mathbf{v}_e + \boldsymbol{\varepsilon}. \tag{A4}$$

We refer to $\mathbf{v}_e$ as the edge effect and we shall characterize its probabilistic features such as its distribution. We go back to the definition of the edge effect and observe that it is a collection of indicator random variables driven by mutation:

$$\mathbf{v}_e = \sum_{p:p\in e} \boldsymbol{\beta}_p \mathbf{1}_{ep}. \tag{A5}$$

In turn, $\mathbf{v}_e$ as a whole is a random variable. Since mutations occur at a low rate, $\mathbf{v}_e$ is likely to be zero most of the time, but conditioning on the occurrence, a mutation hits position p at a probability that is proportional to the position's mutation rate u_{ep}. The effect size of this mutation is $\boldsymbol{\beta}_p$. Altogether, $\mathbf{v}_e$ is zero or a number drawn from the pool of effect sizes $\{\boldsymbol{\beta}_p\}_{p\in e}$ with probability proportional to the mutation rate u_{ep}. Hence, the distribution is determined by the collection $\{(\boldsymbol{\beta}_p, u_{ep})\}_{p:p\in e}$.

This formulation closely mirrors the continuum-of-alleles (CoA) model proposed by Kimura and Crow (1964). Similar to an edge that contains many positions, a locus also stretches over a genome segment that is subject to mutation. When a mutation arrives, it occurs at a random position yielding a different allele every time, i.e., the infinite-alleles model. The CoA model describes the distribution of the effect size of these newly emerging alleles.

Our branch based expression in equation (A4) represents a phenotype by a design matrix with effect sizes that are random with respect to mutations. Hence, it can be considered as a linear random effects model. Furthermore, some features of the model can be deduced from the properties of the mutational process and the topology of the ARG. The variance of an edge can be computed as follows:

$$\begin{aligned}
\mathrm{Var}\left(\mathbf{v}_e\right) &= \mathrm{Var}\left(\sum_{p:p\in e} \boldsymbol{\beta}_p \mathbf{1}_{ep}\right) \\
&= \sum_{p:p\in e} \boldsymbol{\beta}_p^2 \mathrm{Var}\left(\mathbf{1}_{ep}\right) \quad \because \text{mutations at different } p \text{ are independent} \\
&= \sum_{p:p\in e} \boldsymbol{\beta}_p^2 l_e u_{ep}(1 - l_e u_{ep}) \quad \because \mathrm{Var}\left(\mathbf{1}_{ep}\right) = \mathbb{E}\left[\mathbf{1}_{ep}\right]\left(1 - \mathbb{E}\left[\mathbf{1}_{ep}\right]\right) \\
&\approx \sum_{p:p\in e} \boldsymbol{\beta}_p^2 l_e u_{ep} \quad \because l_e u_{ep} \ll 1 \text{ i.e., mutation rate is small} \\
&= \overbrace{l_e s_e}^{\text{edge area:}=A_e} \cdot \overbrace{\frac{1}{s_e} \sum_{p:p\in e} \boldsymbol{\beta}_p^2 u_{ep}}^{\substack{\text{average of squared effect sizes} \\ \text{times mutation rate:}=\tau_e^2}} \quad \because l_e \text{ does not depend on } p \\
&= A_e \tau_e^2
\end{aligned} \tag{A6}$$

The variance of an edge effect is proportional to its area A_e. τ_e^2 is the average squared effect size weighted by mutation rates. In words, regions with higher functional importance (large $\boldsymbol{\beta}_p^2$) and mutation rate (large u_{ep}) contribute more to branch/edge effect variation. Assuming that mutation rates are constant and the effect sizes are evenly dispersed across the genome, similar to the infinitesimal model, this recovers the ARG-LMM model we have used in the main portion of this paper.

In the main portion of the paper, the branch effects were assumed to be mean zero and mutually independent. A derivation of these (approximate) properties helps shed light on how and when this is a good assumption. To do this, we decompose $\mathbf{v}_e$ into two parts $\mathbf{u}_e = \mathbf{v}_e - \mathbb{E}[\mathbf{v}_e]$ and $\mathbf{f}_e = \mathbb{E}[\mathbf{v}_e]$. The vector of these quantities are defined accordingly: $\mathbf{u} = \mathbf{v} - \mathbb{E}[\mathbf{v}]$ and $\mathbf{f} = \mathbb{E}[\mathbf{v}]$. Then, we can rewrite equation (A4) as:

$$\mathbf{y} = \mathbf{Zf} + \mathbf{Zu} + \boldsymbol{\varepsilon}. \tag{A7}$$

Henceforth, we refer to $\mathbf{u}_e$ instead of $\mathbf{v}_e$ as the edge effect. From the definition, the branch effect $\mathbf{u}_e$ has mean zero. At first glance, enforcing zero mean by subtracting the mean looks like an unfounded assumption that sweeps the mean contributions $\mathbf{Zf}$ under the rug. Surprisingly, if all individuals exist at the same time, this is not an issue. Under neutral evolutionary scenarios of various kinds, we may assume that the mutation rate is constant for each position (Ewens, 2004; Wakeley, 2008; Durrett, 2008), i.e., $u_{ep} = u_p$ for all e such that $p \in e$ (edge e contains position p). In this case, $\mathbf{Zf}$ reduces to a constant:

$$\begin{aligned}
[\mathbf{Zf}]_i &= \sum_{e=1}^{E} \mathbf{Z}_{ie} \mathbb{E}\left[\sum_{p:p\in e} \boldsymbol{\beta}_p \mathbf{1}_{ep}\right] \quad \because \mathbf{f}_e = \mathbb{E}[\mathbf{v}_e] \\
&= \sum_{e=1}^{E} \sum_{p:p\in e} \mathbf{Z}_{ie} \boldsymbol{\beta}_p l_e u_p \quad \because \mathbb{E}[\mathbf{1}_{ep}] = l_e u_{ep} \text{ and } u_{ep} = u_p \\
&= \sum_{p=1}^{P} \sum_{e:p\in e} \mathbf{Z}_{ie} \boldsymbol{\beta}_p l_e u_p \quad \because \text{swap the sums over } e \text{ and } p \\
&= \sum_{p=1}^{P} \boldsymbol{\beta}_p u_p \left(\sum_{e:p\in e} \mathbf{Z}_{ie} l_e\right) \quad \because \boldsymbol{\beta}_p \text{ and } u_p \text{ do not depend on } e \\
&= \sum_{p=1}^{P} \boldsymbol{\beta}_p u_p \cdot 2t_{\mathrm{root},p} \quad \because \text{ploidy} = 2 \\
&= \text{constant with respect to } i.
\end{aligned} \tag{A8}$$

Note that $\sum_{e:p\in e} \mathbf{Z}_{ie} l_e$ is the sum of lengths of all edges ancestral to individual n that adds up to the root time at position p multiplied by ploidy. This means that the fixed component of the genetic effect is fully accounted for by a model intercept.

We can further generalize equation (A8) to incorporate time varying mutation rates. In this setting, we stipulate that a mutation rate at a position only depends on time, i.e., the mutation rate is a function of time

$u_p(t)$:

$$
\begin{aligned}
[\mathbf{Zf}]_i &= \sum_{e=1}^{E} \mathbf{Z}_{ie} \mathbb{E}\left[\sum_{p:p\in e} \boldsymbol{\beta}_p \mathbf{1}_{ep}\right] \\
&= \sum_{e=1}^{E} \sum_{p:p\in e} \mathbf{Z}_{ie} \boldsymbol{\beta}_p \mathbb{E}\left[\mathbf{1}_{ep}\right] \quad \because \mathbf{Z}_{ie} \text{ does not depend on } p \\
&= \sum_{e=1}^{E} \sum_{p:p\in e} \mathbf{Z}_{ie} \boldsymbol{\beta}_p \int_{t(e(c))}^{t(e(p))} u_p(t)dt \quad \because \mathbb{E}\left[\mathbf{1}_{ep}\right] = \int_{t(e(c))}^{t(e(p))} u_p(t)dt \\
&= \sum_{p=1}^{P} \sum_{e:p\in e} \mathbf{Z}_{ie} \boldsymbol{\beta}_p \int_{t(e(c))}^{t(e(p))} u_p(t)dt \quad \because \text{swap the sums over } e \text{ and } p \\
&= \sum_{p=1}^{P} \boldsymbol{\beta}_p \sum_{e:p\in e} \mathbf{Z}_{ie} \int_{t(e(c))}^{t(e(p))} u_p(t)dt \quad \because \boldsymbol{\beta}_p \text{ does not depend on } e \\
&= \sum_{p=1}^{P} \boldsymbol{\beta}_p \left(2\int_{0}^{t_{\mathrm{root},p}} u_p(t)dt\right) \quad \because \text{ploidy} = 2 \\
&= \text{constant. w.r.t. } i.
\end{aligned}
\tag{A9}
$$

This confirms that the fixed portion $\mathbf{Zf}$ of equation (A7) is constant in a fairly general setting with time varying mutation rates.

In order to demonstrate independence, we show that $\mathrm{Cov}\left(\mathbf{u}_e, \mathbf{u}_{e'}\right) = 0$ for all pairs $e \neq e'$. This implies independence because indicator variables are independent if and only if they are uncorrelated, similar to Gaussian random variables. We first expand $\mathrm{Cov}\left(\mathbf{u}_e, \mathbf{u}_{e'}\right)$ and reduce it to a sum of covariances of indicators:

$$
\begin{aligned}
\mathrm{Cov}\left(\mathbf{u}_e, \mathbf{u}_{e'}\right) &= \mathrm{Cov}\left(\sum_{p:p\in e} \boldsymbol{\beta}_p \mathbf{1}_{ep}, \sum_{p':p'\in e'} \boldsymbol{\beta}_{p'} \mathbf{1}_{e'p'}\right) \\
&= \sum_{p\in e, p'\in e'} \boldsymbol{\beta}_p \boldsymbol{\beta}_{p'} \mathrm{Cov}\left(\mathbf{1}_{ep}, \mathbf{1}_{e'p'}\right) \quad \because \mathrm{Cov} \text{ is bilinear} \\
&= \sum_{p\in e, e'} \boldsymbol{\beta}_p^2 \mathrm{Cov}\left(\mathbf{1}_{ep}, \mathbf{1}_{e'p}\right) \quad \because \text{mutations at different } p \text{ are independent.}
\end{aligned}
\tag{A10}
$$

The last line's simplification follows from the independence of indicator variables $\mathbf{1}_{ep}$ across different positions. When e and e' have no overlapping positions, the summation is trivially zero as $\{p : p \in e, e'\}$ will be an empty set. It's also zero if the two edges are not ancestral to each other, as mutations running on separate edges are also independent. Otherwise:

$$
\begin{aligned}
\mathrm{Cov}\left(\mathbf{1}_{ep}, \mathbf{1}_{e'p}\right) &= \mathbb{E}\left[\mathbf{1}_{ep}\mathbf{1}_{e'p}\right] - \mathbb{E}\left[\mathbf{1}_{ep}\right]\mathbb{E}\left[\mathbf{1}_{e'p}\right] \quad \because \mathrm{Cov}\left(X, Y\right) = \mathbb{E}\left[XY\right] - \mathbb{E}\left[X\right]\mathbb{E}\left[Y\right] \\
&= \mathbb{E}\left[\mathbb{E}\left[\mathbf{1}_{ep}\mathbf{1}_{e'p} \mid \mathbf{1}_{ep}\right]\right] - l_e u_{ep} l_{e'} u_{e'p} \quad \because \mathbb{E}\left[X\right] = \mathbb{E}\left[\mathbb{E}\left[X \mid Y\right]\right] \\
&= \mathbb{E}\left[\mathbf{1}_{ep}\mathbb{E}\left[\mathbf{1}_{e'p} \mid \mathbf{1}_{ep}\right]\right] - l_e u_{ep} l_{e'} u_{e'p} \quad \because \mathbb{E}\left[XY \mid X\right] = X\mathbb{E}\left[Y \mid X\right] \\
&= \mathbb{P}\left(\mathbf{1}_{ep} = 0\right) \cdot 0 \cdot l_{e'} u_{e'p} + \mathbb{P}\left(\mathbf{1}_{ep} = 1\right) \cdot 1 \cdot 0 - l_e u_{ep} l_{e'} u_{e'p} \\
&= -l_e u_{ep} l_{e'} u_{e'p}.
\end{aligned}
\tag{A11}
$$

This latter quantity is small because it is proportional to the per base-pair mutation rate squared. Not allowing child mutations gives $\mathbb{E}\left[\mathbf{1}_{e'p} \mid \mathbf{1}_{ep} = 1\right] = 0$, provided e is an ancestor of e' without the loss of generality. Therefore, we can justify the diagonal covariance matrices of random effects $\mathbf{u}$ by setting all terms involving u_{ep}^2 to zero.

In summary, we have the resulting generalization, that allows varying mutational variance along the genome and through time: The distribution of a complex trait follows a linear mixed model conditioned on the sample's ARG $\mathcal{T} = (\mathcal{N}, \mathcal{E})$:

$$\mathbf{y} = \mathbf{u}_0 + \mathbf{Z}\mathbf{u} + \boldsymbol{\varepsilon}, \tag{A12}$$

with a constant vector $\mathbf{u}_0$ (all entries have the same value). The random effects $\mathbf{u}$'s covariance matrix is:

$$\boldsymbol{\Sigma}_{\mathbf{u}} = \operatorname{diag}\left[l_e s_e \tau_e^2\right]_{e=1,\ldots,E}, \tag{A13}$$

where $\tau_e^2 = \frac{1}{s_e}\sum_{p:p\in e} \boldsymbol{\beta}_p^2 u_{ep}$. The residual $\boldsymbol{\varepsilon}$ contains the contribution of non-genetic variables. We call the covariance matrix $\operatorname{Var}(\mathbf{Z}\mathbf{u}) = \mathbf{Z}\boldsymbol{\Sigma}_{\mathbf{u}}\mathbf{Z}^T$ as the generalized branch GRM. In the model we use in the main paper, we further assume that $\boldsymbol{\beta}_p \sim \text{Normal}(\boldsymbol{\mu}_p, \tau^2)$. Then, $\operatorname{Var}(\mathbf{Z}\mathbf{u})$ further reduces to

$$\operatorname{Var}(\mathbf{Z}\mathbf{u}) = \tau^2 \mathbf{B} \tag{A14}$$

in which $\mathbf{B}$ is the branch GRM from Lehmann et al. (2025). $\boldsymbol{\mu}_p$ does not affect the GRM and is absorbed to the intercept as in equation (A8) and (A9).

B Variance of V_G

First, note that if X and Y are Gaussian, then Isserlis' Theorem (Isserlis, 1918) says that:

$$\mathbb{E}\left[X^2Y^2\right] = \mathbb{E}\left[X^2\right]\mathbb{E}\left[Y^2\right] + 2\mathbb{E}\left[XY\right]^2.$$

Expanding out $\operatorname{Var}(V_G)$, we will use this fact with $X = (\mathbf{P}_N\mathbf{g})_n$ and $Y = (\mathbf{P}_N\mathbf{g})_m$. Continuing with the notation from equation (18):

$$\begin{aligned}
\operatorname{Var}(V_G) &= \operatorname{Var}\left(\frac{1}{N}\sum_{n=1}^{N}(\mathbf{g}_n - \overline{\mathbf{g}})^2\right) \\
&= \frac{1}{N^2}\left(\mathbb{E}\left[(\mathbf{P}_N\mathbf{g})^T(\mathbf{P}_N\mathbf{g})(\mathbf{P}_N\mathbf{g})^T(\mathbf{P}_N\mathbf{g})\right] - \mathbb{E}\left[(\mathbf{P}_N\mathbf{g})^T(\mathbf{P}_N\mathbf{g})\right]^2\right) \\
&= \frac{1}{N^2}\sum_{n=1}^{N}\sum_{m=1}^{N}\left(\mathbb{E}\left[(\mathbf{P}_N\mathbf{g})_n^2(\mathbf{P}_N\mathbf{g})_m^2\right] - \mathbb{E}\left[(\mathbf{P}_N\mathbf{g})_n^2\right]\mathbb{E}\left[(\mathbf{P}_N\mathbf{g})_m^2\right]\right) \\
&= \frac{1}{N^2}\sum_{n=1}^{N}\sum_{m=1}^{N}\mathbb{E}\left[(\mathbf{P}_N\mathbf{g})_n(\mathbf{P}_N\mathbf{g})_m\right]^2 \\
&= \tau^4\frac{1}{N^2}\sum_{n=1}^{N}\sum_{m=1}^{N}\left(\widetilde{\mathbf{B}}_{n,m}\right)^2.
\end{aligned} \tag{A15}$$

Since the offdiagonals of $\widetilde{\mathbf{B}}$ are $\mathcal{O}(1/N)$ (where recall that N is the sample size), the last line tells us that $\mathrm{Var}\,(V_G) = \mathcal{O}(1/N)$, and hence, $\tau^2\,\mathrm{tr}(\widetilde{\mathbf{B}})/N$ is a consistent estimator of V_G, with an error that is of order $1/\sqrt{N}$.

C Correctness of Algorithm T

Here we present an algorithm for efficient simulation from the ARG-LMM model. This serves several purposes: first, the structure of the algorithm helps to highlight the dependency structures within the ARG. Second, the simplest tree-by-tree trait simulation algorithm for a large ARG could be computationally very costly, and this algorithm is much more efficient. Third, to illustrate how a key part of the matrix-vector product algorithm in Lehmann et al. (2025) works.

We begin with a precise explanation of what we'd like to simulate. Consider a given tree T that extends over some span s_T of the genome, and for an edge or sub-edge e and a node n write $e \geqslant_T n$ if e is ancestral to n in T. Each edge $e \in T$ connects a child $c(e)$ to a parent $p(e)$, and so has area $s_T(t_{p(e)} - t_{c(e)})$ in this tree. For each edge in each tree, let the edge effect $\mathbf{u}_{e,T}$ be a Gaussian random variable with mean zero and variance $s_T(t_{p(e)} - t_{c(e)})$. Then, the genetic value of a node n – the quantity we seek to simulate – is equal to $\sum_T \sum_{e \geqslant_T n} \mathbf{u}_{e,T}$. However, if the subtree below an edge e does not change across trees $T_i, T_{i+1}, \ldots, T_j$, then these effects can be combined: $\mathbf{u}_{e,T_i} + \cdots + \mathbf{u}_{e,T_j}$ is inherited together by the same set of nodes. By additivity of Gaussian distributions, this combined effect is itself Gaussian with variance equal to the combined areas, and so we aim to simulate $\mathbf{Zu}$, where $\mathbf{Z}$ is the individual–sub-edge design matrix defined above and $\mathbf{u}$ is a vector of sub-edge effects from a Gaussian distribution with mean zero variance equal to sub-edge areas. (We emphasize that that in practice a single edge in the ARG can subtend many different subtrees, so that each e denotes a sub-edge as defined above.)

The algorithm works, roughly, as follows. The "value" $v(n)$ of each node caches a total contribution to genetic value that is inherited by every node in the current subtree below n. The "current value" of node n is obtained by summing up the values of all nodes above n in the current tree – so, if the tree changes, we must make adjustments to maintain a correct current value. When an edge with parent p and child c is removed from the tree, two things happen. First, the effect of that edge is simulated, and added to $v(c)$. Then, the value of p and every node above p must be added to $v(c)$, because c is no longer below those nodes. Adding an edge creates a different problem: after adding the edge, c is below p in the tree, but $v(p)$ contains effects from previous parts of the genome. So, we subtract the values of p and all nodes above p from $v(c)$, so that these effects cancel out when summing up the tree from c. At the end of the algorithm all edges are removed, so we end up with the empty tree, and so $v(n)$ will be equal to the genetic value of node n.

Algorithm T. (*Trait genetic value simulation*). Given a sequence of positions $0 = b_0 < b_1 < \cdots < b_K = L$ that are recombination breakpoints along the genome of length L, and corresponding sequences of edges to remove (R_k) and add (A_k) at each position, sample the genetic values g_s for $1 \leqslant s \leqslant n_S$, as long as all samples are leaves in all trees. Let T be the current tree, $\ell_T(n) = t_{p(n)} - t_n$ be the length of the edge above node n in T (or zero, if n has no parent), initialize $k = 1$, $x(n) = 0$, and $v(n) = 0$ for all $n \in \mathcal{N}$. At all times, define $z(n) = \sqrt{\ell_{T_k}(n)(b_k - x(n))}$. The k-th tree T_k has span $s_{T_k} = b_k - b_{k-1}$.

T1. [Remove edges] For each edge $(c, p) \in R_k$, and for each node $n \geqslant_T p$, sample $\mathbf{w}_n \sim \text{Gaussian}(0, 1)$; set $v(n) \mathrel{+}= z(n)\mathbf{w}_n$; then set $v(c) \mathrel{+}= v(n)$ and $x(n) = b_k$. Then, set $x(c) = b_k$ and remove the edge.

T2. [Add edges] For each edge $(c, p) \in R_k$, and for each node $n \geqslant_T p$, sample $\mathbf{w}_n \sim \text{Gaussian}(0, 1)$; set $v(n) \mathrel{+}= z(n)\mathbf{w}_n$; then set $v(c) \mathrel{-}= v(n)$ and $x(n) = b_k$. Then, set $x(c) = b_k$ and add the edge.

T3. [Iteration] If $k < K$, set $k = k + 1$ and return to **T1**. Otherwise, set $g_s = v(s)$ for $1 \leqslant s \leqslant n_S$ and finish.

The algorithm proceeds by storing and updating intermediate quantities v, z, and x indexed by nodes. The genetic value is recovered at the end of the algorithm by aggregating these intermediate values efficiently. Updating the intermediate values is also efficient because it only alters the entries that are ancestral to the added or deleted edge in the local tree while leaving all other entries constant.

To prove that the algorithm works, we first reformulate the algorithm so that we have the edge effects beforehand, making the algorithm deterministic. To this end, for each node n in the kth tree, let $\mathbf{z}_{n,k}$ have a Gaussian distribution with mean zero and variance equal to the span of the tree, $b_k - b_{k-1}$. Recall that $x(n)$ is the last position at which node n was visited, and so we can define $k(n)$ so that $b_{k(n)} = x(n)$. Now, define $Z(n) = \sqrt{\ell_{T_k}(n)}(\mathbf{z}_{n,k(n)} + \cdots + \mathbf{z}_{n,k})$. Comparing to Algorithm C, $Z(n)$ and $z(n)\mathbf{w}_n$ have the same distribution; the only difference is that we sample from the Gaussian here in several steps. So, where steps **T1** and **T2** say "set $v(n) \mathrel{+}= z(n)\mathbf{w}_n$", instead replace these with the equivalent instruction "set $v(n) \mathrel{+}= Z(n)$".

To show that the algorithm calculates the genetic value, we first show how to obtain the genetic value for each node based on the genome up until the current position. Then, we show that steps **T1** and **T2** in Algorithm C do not change the genetic value.

First, define the "working genetic value", which is the genetic value of node n calculated only the genome up until position b_k, as

$$g_k(n) = \sum_{h=1}^{k} \sum_{u \geqslant_T n} \sqrt{\ell_{T_h}(u)}\mathbf{z}_{u,h}.$$

This has the correct distribution because $\sqrt{\ell_{T_h}(u)}\mathbf{z}_{u,h}$ has a Gaussian distribution with mean zero and variance equal to $(b_h - b_{h-1})\ell_{T_h}(u)$, which is the area of the edge above u in tree h. By additivity of Gaussians, the contribution of a given edge that spans multiple trees is again Gaussian has mean zero and variance equal to its area.

Also, define

$$\hat{g}(n) = \sum_{r : r \geqslant_T n} (v(r) + Z(r)). \tag{A16}$$

We claim that at any point in the algorithm,

$$g_k(n) = \hat{g}(n), \qquad \text{for all } n. \tag{A17}$$

At the end of the algorithm, the value returned is $\hat{g}(s)$, because tree is empty at termination, so the right-hand side reduces to $v(s)$. $z(s)$ is zero because $x(s) = b_K$ (recall that $z(n) = \sqrt{\ell_{T_k}(n)(b_k - x(n))}$). Therefore, if equality (A17) holds throughout the algorithm, then $v(s)$ is the desired genetic value, $g_K(s)$.

We prove equation (A17) by induction. The equality trivially holds at the beginning of the algorithm, because both sides are zero. Next we prove that each of the three steps in the algorithm preserves the equality (A17).

Step T3 is the easiest to analyze. By the induction step, we assume that we began this step with $\hat{g}(n) = g_k(n)$. The current tree T and the intermediate quantities v and x do not change. Advancing the position

from k to $k' = k+1$ changes $Z(n)$ by adding $\sqrt{\ell_{T_k}(n)}\mathbf{z}_{n,k+1}$, and so changes $\hat{g}(n)$ by

$$\sum_{r:r \geqslant_T n} \sqrt{\ell_{T_k}(r)}\mathbf{z}_{r,k+1}.$$

Comparing to equation (A16), this is equal to $g_{k+1}(n) - g_k(n)$, and so at the end of this step, $\hat{g}(n) = g_{k'}(n)$, as desired.

This is the only step where the working genetic value changes. In the other two steps, v, z, and x change, but $\hat{g}(n)$ remains constant, thus preserving equation (A17). We prove this only for **T2**; the argument is basically the same for **T1**.

Suppose that we added edge (c, p) where c and p are child and parent nodes of the edge, respectively. Call the tree at the start of the step T, and the tree after the step T', and similarly for x', v', Z', and $\hat{g}'$. Since adding an edge only increases the set of nodes above n, the difference in $\hat{g}(n)$ caused by **T2** is

$$\begin{aligned}\hat{g}'(n) - \hat{g}(n) &= \sum_{r:n \leqslant_{T'} r} (v'(r) + Z'(r)) - \sum_{r:n \leqslant_T r} (v(r) + Z(r)) \\ &= \sum_{r:n \leqslant_{T'} r, n \nleqslant_T r} (v'(r) + Z'(r)) + \sum_{r:n \leqslant_T r} (v'(r) + Z'(r) - v(r) - Z(r)).\end{aligned} \tag{A18}$$

We can divide the nodes of $\mathcal{N}$ into four categories as (1) the child node c, (2) nodes below c, (3) the parent node p and nodes above p, and (4) all other nodes. Since $v(r)$ and $Z(r)$ only change for nodes that are above c in T', evidently $\hat{g}'(n) = \hat{g}(n)$ for nodes in the last category. Next, note that $v(n) + Z(n) = v'(n) + Z'(n)$ for nodes n that are above p: this is because **T2** first sets $v'(n) = v(n) + Z(n)$, then sets $x(n) = b_k$, which has the effect that $Z'(n) = 0$. Therefore, $\hat{g}'(n) = \hat{g}(n)$ for nodes $n \geqslant_T p$.

This leaves only nodes $n \leqslant_{T'} c$. The first term in equation (A18) sums over $\{r : n \leqslant_{T'} r, n \nleqslant_T r\}$: nodes that are above n after adding the edge, but were not above n before the addition. So, for nodes below c, this set is simply all nodes above p, and the first summation is

$$\sum_{r:p \leqslant_{T'} r} (v'(r) + Z'(r)). \tag{A19}$$

The only nodes that change x are those above p, so for nodes $r \leqslant_T c$, $Z'(r) = Z(r)$. Furthermore, $v'(r) = v(r)$ for nodes strictly below c. So, the summand of the second summation is $v'(r) + Z'(r) - v(r) - Z(r)$ is equal to 0, except for $r = c$, in which case it is $v'(c) - v(c)$. Therefore, the second summation is just $v'(c) - v(c)$. Now, $v(c)$ changes because this step, for each $r \geqslant_T p$, first sets $v'(r) = v(r) + Z(r)$, and then subtracts $v'(r)$ from $v(c)$. So, still for nodes n below c, the second summation is

$$v'(c) - v(c) = -\sum_{r:p \leqslant_T r} (v(r) + Z(r)). \tag{A20}$$

Putting these together, the two summations in equation (A18) cancel, so $\hat{g}'(n) = \hat{g}(n)$ for all remaining nodes.

In summary, **T1** and **T2** modify the intermediate values v, x, and Z while leaving the working genetic value constant so that **T3** correctly updates the genetic value. See **Efficient ARG-LMM simulation** for more details of fully simulating the trait based on this algorithm.

D Submatrix computation

`tskit` provides multiplication by $\mathbf{B}$ (and $\mathbf{B}^{-1}$ via conjugate gradient), and for its submatrices, one can multiply an incidence matrix before and after multiplying $\mathbf{B}$ to perform matrix-vector multiplication. For instance, $\mathbf{B}_{n,o}\mathbf{v}$ can be done by:

$$\mathbf{B}_{n,o}\mathbf{v} = \mathbf{A}_n \cdot \mathbf{B} \cdot \mathbf{A}_o^T \mathbf{v}, \tag{A21}$$

where $\mathbf{A}_n$ and $\mathbf{A}_o$ are incidence matrices with each row corresponding to the non-phenotyped and observed individuals. For each row, only one column that corresponds to the index of the row in the full list of individuals are set to 1 and 0 otherwise. Multiplication by incidence matrices $\mathbf{A}_\bullet$ can be done very quickly as they are very sparse.

The conditional variances of the predicted genetic values may be useful as a measure of precision for individual BLUPs, but in large-scale applications assembling the full covariance matrix (equation 20) is infeasible. However, we can obtain unbiased estimates of the diagonal using the XDiag algorithm in Epperly, Tropp, and Webber (2024), which only requires matrix-vector products of equation (20) against a fixed number of random test vectors.

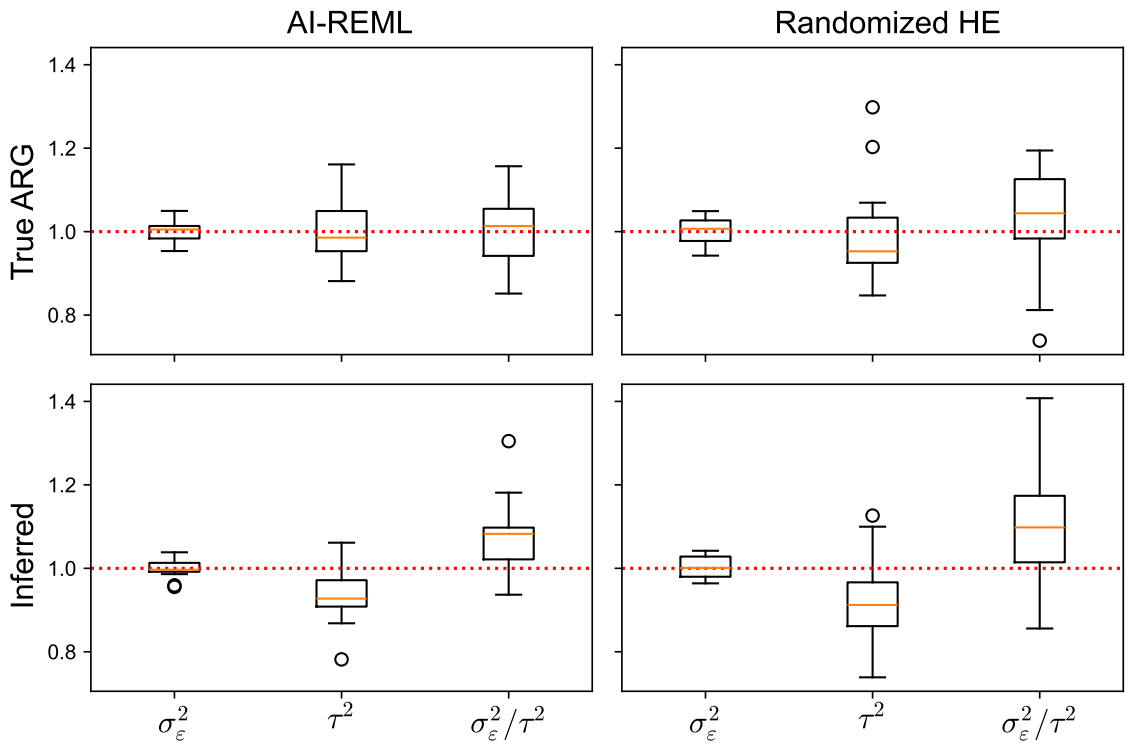

Figure A3: Statistical precision of the HE method and AI-REML in the spatial simulation benchmark. 20 replicate experiments were conducted.