GPC: An expressive and tractable deep generative model for genetic variation data

Prateek Anand[1], Anji Liu[2], Meihua Dang[3], Boyang Fu[4], Xinzhu Wei[5], Guy Van den Broeck[1,*], and Sriram Sankararaman[1,6,7,*]

[1]Department of Computer Science, UCLA, Los Angeles, CA, USA
[2]Department of Computer Science, National University of Singapore, Singapore
[3]Department of Computer Science, Stanford University, Stanford, CA, USA
[4]Department of Biomedical Informatics, Harvard Medical School, Boston, MA, USA
[5]Department of Computational Biology, Cornell University, Ithaca, NY, USA
[6]Department of Human Genetics, UCLA, Los Angeles, CA, USA
[7]Department of Computational Medicine, UCLA, Los Angeles, CA, USA
[*]Joint supervision

Abstract

Generative models play an important role in population genetics, being used to generate artificial genomes (AGs) that help benchmark methods, test evolutionary hypotheses, and construct reference panels for imputation while working around data-sharing restrictions. Existing generative models of genetic variation, however, struggle to faithfully express dependencies in the data while retaining tractability and preserving privacy.

We introduce Genetic Probabilistic Circuits (GPC), a deep generative model for genetic variation data based on hidden Chow-Liu trees represented as probabilistic circuits. GPC generalizes traditional hidden Markov models by allowing arbitrary tree structures over latent variables, enabling it to capture long-range dependencies among SNPs. GPC is tractable, supporting exact computation of marginal and conditional probabilities that enables both AG generation and direct genotype imputation (avoiding the need for simulating AGs).

We show that GPC generates AGs that accurately reproduce population structure and linkage disequilibrium patterns across a range of length scales. Compared to other deep generative approaches, GPC consistently improves imputation accuracy, with particularly strong gains for low-frequency variants and in populations that are not well-represented in public reference panels. Finally, we show that GPC better preserves privacy of the training data, thereby providing a practical framework for AG generation in settings with limited data access.

1 Introduction

Generative models of genetic variation play a central role in population genomics [1, 2]. By modeling dependencies across individuals and variants, these models support core analyses such as genotype imputation [3],

haplotype phasing [4], and ancestry inference [5]. They also form the foundation of simulators that generate artificial genomes (AGs) [6, 7, 8, 9, 10], which are widely used to test evolutionary hypotheses, estimate population genetic parameters, validate empirical results, and benchmark new methods. As the sharing of primary genetic data becomes increasingly restricted due to privacy and consent constraints, accurate simulation tools have become essential for reproducibility and equity. Trained generative models, or data simulated from them, can in principle be shared without requiring access to identifiable genomic data.

Classical population genetic simulators rely on the coalescent model [11], which generates genomes based on demographic history together with mutation and recombination rates. The coalescent with recombination [12] connects observed genetic variation to latent genealogies that change along the genome [13] and is highly expressive, but exact inference under this model is computationally challenging, motivating tractable approximations. One class approximates the coalescent as a Markovian process along the genome [14, 15], forming the basis of widely used simulators such as `msprime` [9] and its extensions [10]. A second class dispenses with explicit genealogical modeling and instead directly approximates the data distribution. A prominent example is the product-of-approximate-conditionals (PAC) model [16], which naturally yields a hidden Markov model (HMM) [17]. HMM-based models have proven highly successful in applications including haplotype phasing [18, 19], genotype imputation [20, 3], and ancestry inference [21, 22].

Motivated by the success of deep learning, recent work has explored deep generative models as an alternative approach to represent genetic variation [23, 24, 25]. Models based on generative adversarial networks (GANs) [25, 26, 27, 28], variational autoencoders (VAEs) [23, 29, 30], restricted Boltzmann machines (RBMs) [25], and more recently diffusion models [31] can be more expressive than HMMs and often produce AGs that visually resemble real genomes in principal component analysis (PCA) or allele frequency summaries. However, these models have critical limitations for genomic applications. First, GANs do not define a probability distribution over the data, making likelihood-based inference impossible. RBMs define a distribution but require computing an intractable partition function over exponentially many configurations, preventing exact likelihood evaluation. VAEs define a probability distribution but the marginal likelihood requires integrating over a continuous latent space with a nonlinear decoder, which is intractable; only a lower bound (the ELBO) is accessible in practice. Diffusion models can in principle compute exact likelihoods via probability flow ordinary differential equations, but scale poorly to SNP data: existing approaches [31] require lossy dimensionality reduction (e.g., per-gene PCA) as a preprocessing step before the model operates [31], and have not been evaluated on imputation tasks. Second, none of these approaches support efficient conditional probability estimation. This limitation makes model comparison difficult and limits their use in downstream tasks such as genotype imputation, which require computing the probability of unobserved variants conditioned on observed variants. A common workaround involves generating AGs from these models, which then serve as a reference panel for imputation methods such as Impute5, Beagle, or Minimac [32, 33, 34], though this approach can introduce additional noise in the imputation process. Third, training these models is challenging due to complex optimization objectives and large hyperparameter spaces, and convergence must be assessed through subjective visual inspection rather than principled likelihood monitoring. These difficulties collectively highlight the need for deep generative models that are both expressive and tractable.

We address this need by introducing GPC (Genetic Probabilistic Circuits), a class of deep generative models that fits genetic variation data with high accuracy while supporting efficient and exact inference. To model the distribution over single nucleotide polymorphisms (SNPs) in an individual, we propose a latent variable model in which each SNP is associated with a hidden variable and the hidden variables form a

tree-structured graphical model. This model, termed the hidden Chow-Liu tree (HCLT) [35], generalizes the HMM. Classical HMMs impose a chain structure where hidden variables corresponding to consecutive SNPs are adjacent in the tree. HCLTs relax this constraint by allowing arbitrary tree structures, which enables the model to capture long-range dependencies, or linkage disequilibrium (LD), across the genome [36]. This flexible structure allows HCLTs to more faithfully represent the distribution of genetic variation. While HCLTs are more expressive than HMMs, naively training and performing inference in HCLTs at the scale required for modern genomic datasets would be computationally prohibitive. Our approach represents HCLTs as probabilistic circuits (PCs) [37, 38], a circuit-based framework that permits tractable inference tasks such as exact likelihood and conditional probability computation. This representation allows us to leverage stochastic gradient-based learning and GPU acceleration to train HCLTs with millions of parameters on large genomic datasets.

GPC, like GANs and RBMs, can generate AGs for use as reference panels with standard imputation tools. Unlike these methods, GPC can also perform imputation directly through efficient computation of conditional probabilities, without the need for generating AGs as an intermediate step. We show that this ability further improves imputation accuracy.

We first compare GPC to baselines on genomic regions chosen from the 1000 Genomes Project (1KG) and UK Biobank (UKBB), demonstrating that GPC achieves higher held-out log-likelihoods than simpler probabilistic models and generates AGs that accurately capture population structure and linkage disequilibrium patterns across all distance ranges. We then evaluate GPC extensively for genotype imputation. Across both datasets, GPC yields significant improvements in imputation accuracy compared to other deep generative models, with substantial improvements in settings where reference genomes from the target population are not publicly available. In such population-specific settings, GPC consistently outperforms other deep generative methods, with particularly strong gains for low-frequency variants. Finally, we find that AGs produced by GPC offer improved privacy properties compared to previous deep generative models. Overall, our results indicate that the increased expressivity and tractability of GPC lead to more accurate and privacy-preserving models of human genetic variation.

2 Results

2.1 Methods Overview

We present GPC, a deep generative model for genetic variation data. GPC is based on hidden Chow-Liu trees (HCLTs) [35], a class of latent variable models that generalize hidden Markov models (HMMs), represented as probabilistic circuits (PCs) [37, 38] for efficient inference and learning.

Hidden Chow-Liu Trees HCLTs are tree-structured graphical models that represent the probability distribution over observed random variables, $\mathbf{X} = (X_1, \ldots, X_N)$. In the context of genetic variation across N SNPs, each observed X_n represents the genotype at SNP n (e.g., $X_n \in \{0, 1\}$ for haploid genomes). HCLTs rely on the following key designs:

- Hidden Variable Mapping: For every observed variable X_n, the model introduces a corresponding hidden variable Z_n (which takes one of L discrete values). Together, these form hidden variables $\mathbf{Z} = (Z_1, \ldots, Z_N)$.

- Chow-Liu Tree Architecture: Each observed variable X_n connects solely to its hidden counterpart Z_n. The hidden variables, however, connect to each other to form a tree structure. This tree is learned using the Chow-Liu algorithm [39] to capture the strongest pairwise correlations between variables (Figure 1).

- Parameterization: The entire HCLT is defined by just two sets of conditional probability distributions: the emissions $P(X_n|Z_n)$ and the tree transitions $P(Z_n|Z_{\text{Pa}(n)})$, where $\text{Pa}(n)$ denotes the parent of node n in the hidden variable tree.

The critical distinction from HMMs is structural. Classical HMMs impose a chain structure where hidden variables corresponding to consecutive SNPs are adjacent in the tree ($Z_1 \to Z_2 \to \ldots \to Z_N$). This chain topology can only propagate information between distant SNPs through all intermediate variables. HCLTs relax this constraint by allowing arbitrary tree structures, so that SNPs with strong long-range correlations (such as X_1 and X_5 being highly correlated without being correlated with intermediate SNPs) can be placed close together in the tree (Figure 1). This flexible structure allows HCLTs to more faithfully capture linkage disequilibrium patterns at all length scales.

Probabilistic Circuits and Tractable Inference While HCLTs are more expressive than HMMs, naively training and performing inference in HCLTs at the scale required for modern genomic datasets would be computationally prohibitive. Our key insight is to represent HCLTs as probabilistic circuits (PCs) [37, 38]. A PC represents a joint probability distribution through a directed acyclic graph (DAG) consisting of input, sum, and product nodes. Product nodes define factorized distributions over their inputs, while sum nodes represent mixtures weighted by learnable parameters. Under structural constraints known as smoothness and decomposability, arbitrary marginal queries can be computed in time linear in the size of the circuit. This tractability is what enables GPC to perform three key operations efficiently: (1) computing likelihoods on held-out data for principled model comparison, (2) generating artificial genomes via ancestral sampling, and (3) performing direct genotype imputation through exact conditional probability computation. By representing HCLTs as equivalent PCs and training them using the PC package PyJuice [40], we can extensively parallelize Expectation-Maximization (EM) updates on GPUs, enabling efficient training even for models with over 88 million parameters (Section 4.2).

Generating AGs from GPC is performed via ancestral sampling through the circuit in time linear in the circuit size. For genotype imputation, GPC computes conditional probabilities $p(X_{\text{missing}}|X_{\text{observed}})$ directly as ratios of marginal queries, both of which can be evaluated via a single feedforward pass through the circuit. This direct imputation capability, unique to GPC among the deep generative approaches we consider, eliminates the need for intermediate AG generation and, as we show, yields consistently higher accuracy (see Section 4.2.2 for full details on sampling and conditional inference).

2.2 Data

Our experiments evaluate different aspects of GPC in terms of its ability to generate realistic AGs, its ability to perform genotype imputation, and its privacy properties. We perform the evaluations on three datasets with distinct properties: 1000 Genomes Project Phase 3 (1KG) [41], UK Biobank (UKBB) [42], and high-coverage 1KG (high-coverage 1KG) [43], with the high-coverage 1KG dataset only being used for the array-based imputation experiment (see Section 4.1 for details).

2.3 Evaluation

Baselines We compare GPC against two categories of baselines. The first category consists of deep generative models that have been previously proposed for AG generation: (1) generative adversarial networks (WGAN) and (2) restricted Boltzmann machines (RBM) as implemented in [25]. We focus our comparison on these two methods as they are the most established deep generative approaches specifically designed and benchmarked for the local genetic structure and imputation tasks we evaluate here. More recent approaches including VAEs [29, 30] and diffusion models [31] have been proposed for related but distinct tasks, and do not support efficient, tractable probabilistic inference; we leave systematic comparison with these methods to future work. For both WGAN and RBM baselines, we use the samples generated by the corresponding authors for comparison[1] in our 1KG experiments, but we also retrain their models on our own training split to ensure a fair comparison. In imputation experiments, we additionally compare against Impute5 [32], a state-of-the-art imputation tool widely used in genetics. Impute5 serves a dual role: it is used with real reference genomes to establish performance benchmarks, and it is used with AG reference panels generated by each method to evaluate their utility for imputation.

The second category consists of simpler probabilistic graphical models (PGMs) that support tractable likelihood computation: fully-factorized distributions (Indep), Markov chain models of order 1 (Markov), and non-homogeneous hidden Markov models (HMM). These serve as baselines for density estimation and AG quality comparisons.

Evaluation criteria We evaluate these models using the following metrics: (1) log-likelihood on test data to assess the capability of each model as a density estimator; (2) summaries of AGs sampled from each model, including the top principal components and linkage disequilibrium at pairs of SNPs; (3) genotype imputation, which evaluates the models' utility on an important downstream task; and (4) privacy analysis of the AGs sampled from each model.

2.4 GPC trains efficiently with objective convergence criteria

Training GPC requires minimal hyperparameter tuning. We set the number of latent states to 128 (the maximum feasible given memory constraints) and use a small pseudocount (0.005) for smoothing. Models were trained for 2,000–5,000 epochs depending on dataset size. We use the PC package PyJuice [40] to train GPC efficiently on GPUs, achieving EM updates in under 2, 10, and 3 seconds per epoch on the 1KG, UKBB, and high-coverage 1KG datasets, respectively. On a single NVIDIA RTX A5000 GPU with 24GB available memory, GPC trains in roughly 2, 6, and 4 hours on each dataset, respectively. Once the model is loaded, generating a single sample can be done in under 2, 5, and 3 seconds for each dataset, respectively. Imputing all missing SNPs within a single sample can be done in 20 milliseconds. A key advantage over other deep generative approaches is that convergence can be monitored via held-out log-likelihood, providing an objective stopping criterion rather than subjective visual inspection. Full training details for GPC and all baseline methods are provided in Section 4.4.

2.5 GPC accurately reconstructs local genetic structure

We first compared the ability of different generative models to represent genetic variation in the 1KG and UKBB datasets. We simulate AGs with GPC and all five baselines (WGAN, RBM, HMM, Indep, and

[1] Note that [25] did not perform train/test splits so WGAN and RBM are actually trained on train+test.

Markov) for comparison. For both datasets, we use an 80% train / 20% test split performed at the individual level before haplotype separation; we generate 5,008 AGs from each model for 1KG and 10,000 AGs for UKBB.

GPC learns more accurate probabilistic models than HMM, Markov, and Indep as measured by their log likelihood on the test dataset that was not used for model fitting (Table 1). Note that we do not compare with WGAN and RBM since they do not support tractable exact likelihood computation. We additionally evaluated the quality of AGs based on whether they preserve distances across pairs of haploid genomes. To do this, we compute the pairwise differences of haploid genomes within a single dataset or between the test dataset and an AG dataset and compute the Wasserstein distance between these pairs of distributions; a lower Wasserstein distance indicates that the AGs tend to be more similar to real genomes in the test dataset. WGAN, RBM, and GPC all capture the distribution well with GPC having second-lowest distance behind WGAN on 1KG and having the lowest distance on UKBB (Figure S1 and rows 1-2 of Table S1).

We then analyze the quality of AGs generated by all models in terms of capturing commonly-used summaries of genetic variation data. To visualize all methods in the same latent space, we merge eight datasets (training set, test set, and AGs from six methods) and apply a single PCA to the combined data. Inspecting the top six principal components (PCs) of genomes in the test set and the AGs, we find that AGs generated from GPC qualitatively capture the dominant structure in this dataset as do the deep learning methods (WGAN and RBM), unlike Indep, Markov, and HMM (Figure 2 for 1KG and Figure S2 for UKBB). To quantify the accuracy of these summaries, we compute the Wasserstein distance between the 2D PCA representations of the test genomes versus the AGs (Table S1) and find that distances for the deep learning methods (WGAN, RBM, and GPC) tend to be lower than simpler models.

We also examined patterns of linkage disequilibrium (LD) to assess how the pairwise short and long-range correlations of SNPs can be captured by AGs. While WGAN and RBM tend to be accurate at longer length scales and HMM and Markov are accurate at shorter length scales, GPC is accurate across all length scales (Figure 3). Finally, we examined the topology of the CLT learned on the 1KG dataset to gain intuition into why GPC captures LD at all scales. In contrast to the chain-structured HMM in which every node has degree at most 2 and every edge connects adjacent SNPs along the genome, the learned CLT has 4,599 leaf nodes, 397 nodes with degree ≥ 5, and a maximum degree of 161, with 18.3% of edges connecting SNPs more than 1,000 positions apart in sequential index (Figure 4; Supplementary Figures S3 and S4). The non-chain tree topology allows the model to place strongly correlated SNPs close together in the tree regardless of their genomic positions, enabling direct propagation of information across long ranges without passing through all intermediate variables.

2.6 GPC improves imputation accuracy across diverse scenarios

A key downstream application of generative models and AGs is genotype imputation. To assess the quality of artificial genomes (AGs) produced by GPC and competing generative models, we evaluate their utility in several imputation settings across the datasets described in Section 2.2. One approach is to generate AGs from these models, which then serve as reference panels for Impute5 [32] and other standard imputation tools. Alternatively, as described in Section 2.1, GPC can perform *direct imputation* by leveraging efficient conditional probability computation over the learned model, without the need to generate AGs.

Following the same training protocol as in Section 2.5, we use 80/20 train/test splits and generate the same number of AGs per dataset. For population-specific experiments, the same 80/20 split is applied within the target population. Following the experimental protocol established in prior work [32, 33, 44], we

evaluate imputation accuracy at each SNP by removing the selected SNP from the test haplotypes while keeping all other SNPs observed. Imputation is then performed conditional on the remaining observed SNPs in the test haplotypes. We calculate the squared Pearson correlation (r^2) between the imputed posterior probabilities of carrying allele 1 and the true allelic state at the target SNP (excluding monomorphic SNPs), aggregating results into bins by minor allele frequency (MAF) with 95% confidence intervals computed from 10 bootstrapped replicates. This imputation protocol follows the approach of Yu et al. [44] (Sections 2.6.1 and 2.6.2). We additionally evaluate imputation from genotyping arrays following Rubinacci et al. [32] and Browning et al. [33] for which we use the high-coverage 1KG dataset (Section 2.6.3) (additional details in Section 4.3).

We consider two broad scenarios for imputation. In the first (general) scenario, imputation is applied to genotypes from a cosmopolitan sample (test set) consisting of individuals from multiple ancestries with AGs generated by different models that were also trained on a cosmopolitan panel of matched ancestries consisting of a distinct set of individuals (training set). This setting reflects common practice in large-scale genetic studies where diverse reference panels are used to impute missing genotypes across heterogeneous cohorts. The second (population-specific) scenario considers a setting in which imputation is to be performed in a population for which reference genomes are limited or restricted. Because the target population might be under-represented in public reference panels, imputation accuracy in these populations might be adversely impacted. This scenario is particularly relevant for underrepresented populations in genomic research, where privacy constraints or limited sequencing resources may restrict the availability of ancestry-matched reference data. In a realistic setting, a researcher may only have access to publicly available European reference genomes while population-specific data remains private. GPC and other generative approaches can learn from such private data and either generate AGs or perform direct imputation without exposing the underlying genomes. For the population-specific analyses, we designate a target population which is limited or restricted in access. We consider two sets of target populations: all individuals of non-European ancestry and all individuals of African ancestry. In each of these settings, we also explore how the accuracy of imputation is impacted when AGs are combined with real genomes (possibly with different genetic ancestry characteristics than the target population) that might be available.

2.6.1 General Imputation

We first examine the general setting, in which models are trained on genomes from diverse ancestries and are then used to impute into target genomes drawn from similar ancestries. To mimic this scenario, we train and test models on random splits of the 1KG and UKBB datasets. For 1KG, we also include a comparison to 5,008 AGs based on fitting WGAN and RBM to the same genomic region that were made available by the authors of [25].

We see that reference panels composed entirely of real genomes used with Impute5 achieve the highest imputation accuracy across MAF bins, providing an approximate upper bound on achievable performance (Figure 5a for 1KG and Figure 5b for UKBB). Among the generative models, GPC obtains the highest imputation accuracy across MAF bins. Averaged across both datasets, GPC (direct) achieves a 0.168 (27.5%) improvement in r^2 over the next best method, RBM (0.230 (174%) for low-frequency variants with MAF $< 1\%$). GPC (direct) imputation, via conditional probability computation rather than simulated AGs, tends to be the more accurate version, achieving a 0.105 (15.5%) improvement in r^2 (0.195 (61.5%) for low-frequency variants) over GPC with AG-based imputation, likely due to directly targeting the imputation objective rather than relying on simulated AGs. To assess the robustness of these findings, we repeated the UKBB

general imputation experiment on five additional LD blocks across distinct chromosomes (see Section 4.1.2 for details) to find the same qualitative pattern holds across all regions: GPC (direct) consistently improves over RBM (the next best method) across the range of MAF (Figure 5c; additional per-chromosome results in Supplementary Figure S5).

2.6.2 Population-Specific Imputation

We next consider imputation in populations for which reference genomes are limited or restricted. Because large public datasets are predominantly of European ancestry, we consider scenarios where the target population consists of individuals of either non-European or African ancestry. In this setting, imputation accuracy in the target population can be degraded when public European reference genomes are the only data available to the imputation method. AGs specific to the target population generated by GPC can mitigate this challenge by learning from an ancestry-matched set of private reference genomes. We also evaluate combined reference panels formed by augmenting real European data with population-specific AGs.

Analogous to the general imputation experiments, GPC (direct) remains the most accurate across MAF bins (Figures 6 and S6 for 1KG and UKBB respectively). Averaged across datasets and ancestries, GPC (direct) achieves a 0.154 (33%) improvement in r^2 over the next best method, RBM (0.202 (279%) for low-frequency variants). Unlike in the general imputation setting, GPC, when used for direct imputation, tends to be more accurate than using Impute5 with a reference panel of European genomes, likely due to the distributional mismatch between the reference and target populations. This holds consistently across the 1KG dataset. In the UKBB dataset, GPC (direct) outperforms Impute5 for common variants but is slightly less accurate for rare variants. We attribute this to our balanced sampling strategy (Section 4.1.2), in which the European reference panel available to Impute5 and the population-specific training data for the generative models are matched in sample size. Under this design, a large European reference panel (even one mismatched in ancestry) can partially compensate for distributional mismatch at rare variants, where population-specific sample sizes are most limiting. Importantly, when the real European genomes are incorporated into the training of the generative models (the "combined" setting), all methods improve, and GPC (direct combined) remains competitive with or above Impute5 across MAF bins, indicating that the gains from ancestry matching and from larger sample size are complementary. Averaged across datasets and ancestries, GPC (direct) achieves a 0.056 (12.3%) improvement in r^2 over Impute5 using European genomes (0.012 (42.1%) for low-frequency variants). Finally, we find that combining population-specific AGs with European genomes consistently improves accuracy across all MAF bins. Averaged across datasets and ancestries, GPC (direct) achieves a 0.011 (1.8%) improvement in r^2 over itself when including European genomes in the reference panel (0.023 (12.3%) for low-frequency variants).

Taken together, these results demonstrate that population-specific AGs from GPC can substantially enhance imputation accuracy in underrepresented groups, either by augmenting existing reference panels or through direct conditional imputation. The magnitude of improvement depends on several interacting factors: the degree of genetic distance between the target population and the available public reference panel, the sample size of the private population-specific data, and the frequency of the variants being imputed. When public reference data is both ancestrally mismatched and matched in size to the private data, as in our balanced UKBB design, gains over a large European-only panel are most pronounced for common variants, while rare variant performance remains sensitive to total sample size regardless of ancestry. These findings suggest that in practice, combining ancestry-matched private data with available public genome (even those from a different ancestry) will generally yield the best imputation accuracy, and that the relative

value of each component will depend on the specific target population and variant frequency spectrum.

2.6.3 Array-based Imputation

We also evaluate GPC in a more realistic scenario of imputation from variants genotyped on a commonly-used SNP array. In this experiment, we attempted to impute SNPs genotyped in the high-coverage 1KG dataset from variants that were typed on the HumanOmni5Exome array (see Section 4.1.3 for details).

GPC (direct) remains the most accurate in both the general (Figure 7) and population-specific settings (Figure 8). In the general setting, GPC (direct) achieves a 0.020 (3.5%) improvement in r^2 over the next best method, RBM (0.043 (14.3%) for low-frequency variants). In the population-specific setting, averaged across both ancestries, GPC (direct) achieves a 0.030 (5.3%) improvement in r^2 over RBM (0.040 (12.7%) for low-frequency variants). Similar to the previous population-specific setting, GPC (direct) outperforms Impute5 with the European-only reference panel, achieving a 0.285 (96.5%) improvement in r^2 averaged across both ancestries (0.324 (1,123%) for low-frequency variants). In contrast to our previous imputation experiments, here the best performance is achieved by GPC (direct) trained solely on the population-specific data, rather than a combined population-specific plus European dataset. Thus, the optimal imputation strategy is likely determined by a combination of genetic ancestries, sample size, and SNP sets, and needs further investigation. These results suggest that in realistic imputation scenarios, population-specific modeling may provide accuracy gains over approaches that aggregate across populations.

2.7 GPC balances considerations of privacy and utility

To evaluate the privacy of the AGs, we compute the nearest neighbor adversarial accuracy ($AATS$) metric introduced by Yale et al. [45]. The metric decomposes into two components: AA_{TRUTH}, which measures whether real samples are closer to other real samples than to synthetic ones, and AA_{SYN}, which measures the analogous property for synthetic samples. Values near 0.5 for both components indicate an ideal balance between utility and privacy. We report AA_{TRUTH} and AA_{SYN} separately rather than their average, as the overall $AATS$ can mask pathological cases where poor values in one component cancel those in the other. We evaluate on both training and held-out test sets and standardize genotypes prior to distance computation as this upweights the impact of rare variants which tend to be more challenging to model (see Section 4.5.1 for details).

Under this evaluation, GPC achieves AA_{SYN} closest to 0.5 across all splits and both datasets, and AA_{TRUTH} closest to 0.5 in all but one case (1KG test, where WGAN is marginally closer: 0.818 vs. 0.820), indicating the best overall balance between utility and privacy (Table 2). The near-zero AA_{SYN} values for RBM paired with near-unity AA_{TRUTH} reveal a distinct failure mode: real samples still find other real samples as their nearest neighbors (real-to-real distances are smaller than real-to-synthetic), but each synthetic sample is closer to some individual real genome than to any other synthetic sample. Rather than forming a coherent distribution, RBM produces samples that float near individual training haplotypes without capturing broader structure, a privacy risk because each synthetic genome can potentially be traced back to a specific training individual. Conversely, WGAN exhibits uniformly high values for both components, indicating that its synthetic distribution occupies a region largely disjoint from the real data, sacrificing utility. We note that even for GPC, AA_{TRUTH} remains noticeably above 0.5, suggesting that the AGs are still somewhat distinguishable from real data.

3 Discussion

In this work, we introduced GPC, a deep generative model for genetic variation based on hidden Chow-Liu trees (HCLTs) and probabilistic circuits (PCs). Unlike prior deep generative models (e.g., GANs, VAEs, and RBMs), GPC combines tractable probabilistic inference with an expressive latent variable structure, enabling principled convergence monitoring via held-out log-likelihood, accurate reconstruction of population structure and linkage disequilibrium patterns across all distance scales, and direct genotype imputation through exact conditional probability computation. In population-specific settings, GPC (direct) consistently outperforms other deep generative methods as well as Impute5 using publicly available European reference genomes, with particularly strong gains for low-frequency variants that are disproportionately population-specific. We also found that GPC AGs provide improved privacy properties compared to RBMs and WGANs, though we emphasize that demonstrating improved performance on a single metric does not guarantee complete privacy preservation. Other attack models, such as membership inference attacks [46, 47] or attribute inference attacks [48], may still pose risks. Future work should explore whether GPC can be extended to provide formal privacy guarantees through differentially private EM algorithms [49].

Despite these advantages, several limitations remain. First, scaling GPC to full genome length remains challenging and will likely require hierarchical or distributed approaches. A straightforward approach to whole genome data would involve applying GPC to LD blocks which are treated as independentof each other though such an approach is likely to ignore LD across blocks. Second, our current implementation of GPC is restricted to haploid genomes to enable direct comparisons with prior approaches. Extending GPC to generate diploid genotype data would increase the applicability of the model. Third, like all generative approaches, GPC inherits biases from the training data, and ensuring fairness across diverse cohorts remains an open challenge. Fourth, additional downstream benchmarks such as rare variant association studies, fine-mapping, or polygenic risk scores would provide a more complete picture of utility. Finally, our experiments focused on single genomic regions; genome-wide evaluation would better assess robustness across varying LD structures and recombination rates.

Additionally, recent work has extended the space of deep generative approaches for genetic data to include VAEs [29, 30], GANs operating in reduced-dimensional spaces [26, 28], and diffusion models [31], as well as genome-wide WGAN and CRBM variants [25]. Systematically comparing GPC against these approaches across imputation and other downstream tasks, and extending all methods to genome-wide scales, remains an important direction for future work.

4 Materials and Methods

4.1 Datasets

4.1.1 1000 Genomes Project Phase 3 (1KG)

We first use data from the 1000 Genomes Project (Phase 3), consisting of 2,504 unrelated diploid genomes spanning diverse ancestries. Because the genomes are phased, we split each into its two constituent haplotypes, yielding 5,008 haplotypes in total. To avoid any possibility of data leakage, all train/test splits are performed at the diploid-individual level, before haplotype separation.

Following prior work [24, 25], we analyze a contiguous 10K-SNP region on chromosome 15 (15:27,379,578 – 15:29,625,035). This dataset is used primarily to compare GPC to baselines and to evaluate how well it

captures population and LD structure. It is also used in our single-SNP imputation experiments.

4.1.2 UK Biobank (UKBB)

To evaluate performance in a larger, secondary cohort, we analyze genotypes from the UK Biobank (UKBB). We analyzed high-quality imputed SNPs (with a hard call threshold of 0.2 and an INFO score ≥ 0.8) with MAF $\geq 0.1\%$. We apply standard quality control by retaining SNPs that are under Hardy-Weinberg equilibrium ($p < 10^{-6}$) and are confidently imputed in more than 99% of the individuals. We retain individuals with no kinship to other individuals (UKBB field 22021) and exclude those with missing ancestry information. Following these procedures, we select an LD block on chromosome 22 (22:29,456,546 – 22:32,665,772). This yields 337,862 individuals and 9,820 SNPs. To assess robustness across genomic regions, we additionally select five LD blocks on separate chromosomes: chromosome 2 (2:118,367,466 – 2:121,303,783; 9,227 SNPs), chromosome 11 (11:101,331,121 – 11:103,959,636; 9,507 SNPs), chromosome 12 (12:85,990,426 – 12:89,682,122; 9,823 SNPs), chromosome 14 (14:72,889,615 – 14:76,444,767; 9,248 SNPs), and chromosome 18 (18:51,554,175 – 18:55,213,838; 9,415 SNPs). These additional regions are used for the general imputation replication experiment. We phased all data using Beagle 5.5.

To ensure computational feasibility for all methods while maintaining statistical power, we randomly select 5,000 individuals each from the European (EUR), non-European (Non-EUR), and African (AFR) ancestry groups, yielding 10,000 haplotypes per group after phasing. Due to partial overlap between the Non-EUR and AFR subsets, the combined imputation dataset includes 26,924 total haplotypes. Similar to 1KG, we use this dataset to evaluate population/LD structure and single-SNP imputation. The balanced sampling strategy also allows us to assess whether equal amounts of European and non-European data can compensate for distributional mismatch when imputing into non-European test populations.

4.1.3 High-coverage 1KG

To evaluate GPC in a realistic imputation scenario, we consider the high-coverage whole-genome sequencing release of 1KG, using the same genomic region as in the earlier 1KG analysis, mapped to build 38 coordinates (15:27,134,431 – 15:29,332,831). We restrict the analysis to the same set of 2,504 unrelated individuals. This region contains 56,766 variants. After selecting biallelic SNPs and removing those with minor allele count (MAC) ≤ 20, 14,670 SNPs remain.

Among these, 2,119 SNPs are present on the HumanOmni5Exome-4v1-2_A genotyping array. The remaining 12,551 SNPs (86%) are imputed simultaneously, providing a setting that closely mirrors real imputation pipelines. This dataset is used for our array-based imputation experiments, both overall and stratified by ancestry.

4.2 Model and Inference Methods

4.2.1 Hidden Chow-Liu Trees

Hidden Chow-Liu trees (HCLTs) [35] represent a distribution over a collection of random variables (RVs) ($\mathbf{Z} = (Z_1, \ldots, Z_N), \mathbf{X} = (X_1, \ldots, X_N)$). $\mathbf{Z}$ denotes hidden or latent RVs while $\mathbf{X}$ denotes observed RVs. The joint distribution is described by a graphical model ($\mathcal{G}$) in which the nodes in the graph represent the RVs and the lack of edges among the nodes represents conditional independence assumptions. In HCLTs, we have edges from each hidden variable to its corresponding observed random variable ($Z_n \to X_n$) while the edges among the hidden variables form a tree. When the graph over the hidden variables is a chain ($Z_1 \to$

$Z_2 \rightarrow \ldots Z_N$), we obtain a hidden Markov model (HMM). By permitting tree-structured graphs, HCLTs generalize HMMs and can provide a better representation of the data to capture long-range dependence, *e.g.*, RV X_2 and X_6 are highly correlated without being correlated with X_3, X_4, X_5 (Figure 1).

For genetic variation data over N single nucleotide polymorphisms (SNPs), each X_n denotes the genotype value at SNP $n \in \{1, \ldots, N\}$ ($X_n \in \{0, 1\}$ when we model haploid genomes). Each Z_n is a discrete RV that can take one of L values ($Z_n \in \{0, \ldots, L-1\}$). Figure 1 demonstrates how to construct an HCLT using genetic data. Given a dataset $\mathcal{D}$ that contains 6 SNPs (Figure 1(a)), we first invoke the Chow-Liu algorithm to generate a tree over the latent variables associated with each SNP (Figure 1(b)). The Chow-Liu algorithm computes pairwise mutual information between all variable pairs and finds the maximum-weight spanning tree, which can be done efficiently even for large SNP panels since it only requires computing pairwise statistics and solving a minimum spanning tree problem. The tree encodes strong variable dependencies by placing highly correlated SNPs (e.g., X_1 and X_3) closer in the generated tree. Finally, the HCLT is constructed by adding an edge from every latent variable Z_i to its corresponding observed variable X_i (Figure 1(c)). The parameters of the HCLT are those associated with the discrete conditional probability distributions $P(X_n|Z_n)$ and $P(Z_n|Z_{\text{Pa}(n)})$, where $\text{Pa}(n)$ denotes the parent of node n in the tree.

4.2.2 Probabilistic Circuits

Probabilistic Circuits (PCs) [37, 38] are a class of probabilistic models that support tractable probabilistic inference. These capabilities have allowed PCs to perform various probabilistic reasoning tasks that are out of reach for most deep generative models [50, 51], including problems in explainable AI [52, 53, 54], algorithmic fairness [55, 56], and missing data robustness [57, 54, 58]. PCs are furthermore appealing for their expressive power and suitability for density estimation, with recent advances in structure learning [59] and parameter estimation [56, 35] allowing PCs to accurately capture useful correlations in the data.

Representation PCs are an umbrella term for a wide family of tractable probabilistic models [60], including arithmetic circuits [61], sum-product networks [62], and cutset networks [63]. A PC $(\mathcal{G}, \boldsymbol{\theta})$ represents a joint probability distribution $\Pr(\mathbf{X})$ over random variables $\mathbf{X}$ through a directed acyclic graph (DAG) $\mathcal{G}$ parametrized by $\boldsymbol{\theta}$. The DAG $\mathcal{G}$ consists of three types of nodes: *input*, *sum*, and *product*. Each leaf node is an input node; each inner node n (i.e., sum or product) receives inputs from its children $\mathsf{ch}(n)$. Each node $n \in \mathcal{G}$ encodes a probability distribution $\Pr_n$, which is defined recursively as follows:

$$\Pr_n(\mathbf{x}) = \begin{cases} f_n(\mathbf{x}) & \text{if } n \text{ is an input node,} \\ \prod_{c \in \mathsf{ch}(n)} \Pr_c(\mathbf{x}) & \text{if } n \text{ is a product node,} \\ \sum_{c \in \mathsf{ch}(n)} \theta_{n,c} \Pr_c(\mathbf{x}) & \text{if } n \text{ is a sum node,} \end{cases} \tag{1}$$

where $f_n(\mathbf{x})$ is a univariate input distribution (e.g., Binomial, Gaussian), and $\theta_{n,c}$ denotes the parameter that corresponds to edge (n, c). Intuitively, a product node defines a factorized distribution over its inputs, and a sum node represents a mixture over its input distributions weighted by θ. Finally, the probability distribution of a PC is defined as the distribution represented by its root node. The size of a PC $(\mathcal{G}, \boldsymbol{\theta})$ is defined as the number of parameterized edges in its DAG $\mathcal{G}$.

Inference In contrast to many other generative models, PCs support efficient reasoning over their encoded distribution. One can compute likelihoods by evaluating the PC feed-forward as in Equation 1. Many

common reasoning tasks such as marginal probabilities and maximum a posteriori probability (MAP) are also supported by PCs. To guarantee the efficiency for computing these queries, the DAG of the PC should satisfy certain structural constraints. Please refer to [64] for a more detailed summary of various inference scenarios for PCs.

To support linear-time computation (with respect to the size of the PC) of arbitrary marginal queries, PCs need to satisfy two structural properties: smoothness and decomposability. Both are properties of the scope $\phi(n)$ of PC units n, that is, the collection of variables defined by all its input nodes.

Definition 1 (Smoothness) *A PC $(\mathcal{G}, \boldsymbol{\theta})$ is smooth if for any sum node $n \in \mathcal{G}$, its children have identical scope: $\forall c_1, c_2 \in \mathsf{ch}(n) : \phi(c_1) = \phi(c_2)$.*

Definition 2 (Decomposability) *A PC $(\mathcal{G}, \boldsymbol{\theta})$ is decomposable if for any product node $n \in \mathcal{G}$, its children have disjoint scopes: $\forall c_1, c_2 \in \mathsf{ch}(n), c_1 \neq c_2 : \phi(c_1) \cap \phi(c_2) = \emptyset$.*

Given a smooth and decomposable PC, querying an arbitrary marginal probability boils down to a feedforward evaluation of its DAG, thus the computation time is linear with respect to the size of the PC.

Sampling and Conditional Inference Sampling from a PC is performed via ancestral sampling. Starting from the root node, we traverse sum nodes by sampling a child proportionally to the edge weights $\theta_{n,c}$, and traverse product nodes by recursively sampling from all children (since they define independent factorizations over disjoint variable sets). At input nodes, we sample from the corresponding univariate distributions. This procedure generates a complete assignment to all variables in time linear in the circuit size.

For genotype imputation, we compute conditional probabilities $p(X_{\text{missing}}|X_{\text{observed}})$. Smooth and decomposable PCs support exact marginalization over any subset of variables: to marginalize out a variable X_i, we simply replace its input distribution with the constant 1 (summing over all possible values). Conditional queries are then computed as ratios of two marginal queries [37]:

$$p(X_{\text{missing}} = x|X_{\text{observed}} = e) = \frac{p(X_{\text{missing}} = x, X_{\text{observed}} = e)}{p(X_{\text{observed}} = e)}. \tag{2}$$

Both the numerator and denominator can be computed via a single feedforward pass through the circuit, making conditional inference efficient.

4.2.3 HCLTs as Probabilistic Circuits

HCLTs can be represented as smooth and decomposable PCs, meaning they support efficient (i.e., linear in the size of the PC) computation of marginal queries and likelihoods. When considering both the observed and hidden variables, HCLTs are also deterministic (i.e., for any sum node, its children have disjoint support), meaning that the MAP instance $\arg\max_{\mathbf{x}\in\mathbf{X},\mathbf{z}\in\mathbf{Z}} \Pr_n(\mathbf{x}, \mathbf{z})$ can be evaluated in linear time. However, HCLTs are not deterministic with respect to the observed variables $\mathbf{X}$ alone, which has implications for parameter estimation as described below.

4.2.4 Parameter Estimation

Any probabilistic graphical model (PGM) can be transformed into a PC that encodes the same probability distribution. We demonstrate the high-level idea of this transformation, and refer interested readers to [38] for more details. To transform an HCLT into an equivalent PC, we iteratively encode every conditional

probability $\Pr(\mathbf{X}_n|\mathbf{X}_{\mathrm{Pa}(n)})$ by representing each possible value of $\mathbf{X}_n$ as a sum node. Thus, the probability of $\Pr(\mathbf{X}_n = \mathbf{x}_n|\mathbf{X}_{\mathrm{Pa}(n)} = \mathbf{x}_{\mathrm{Pa}(n)})$ can be represented by the weight of an edge connecting $\mathbf{x}_n$ and $\mathbf{x}_{\mathrm{Pa}(n)}$. Take the HCLT in Figure 1 as an example. The conditional probabilities are encoded into a single PC in a bottom-up manner: we first encode $\Pr(X_5|Z_5)$ and then followed by $\Pr(X_4|Z_4)$ and $\Pr(Z_5|Z_4)$, and so on.

If a PC is smooth, decomposable, and deterministic, its maximum-likelihood estimation (MLE) parameters can be efficiently learned in closed form [65]. To formalize the MLE parameters, we define the context γ_n of any node n as follows. The context of the root node n_r is its support $\texttt{supp}(n_r)$. The context of any other node is the intersection of its support and the union of its parents' contexts:

$$\gamma_n := \bigcup_{c \in \mathsf{pa}(n)} \gamma_c \cap \texttt{supp}(n).$$

For any sum node n and its child c, the associated MLE parameter $\theta^*_{n,c}$ on a dataset $\mathcal{D} = \{\mathbf{x}_i\}_{i=1}^N$ is

$$\theta_{n,c} = F_{\mathcal{D}}(n,c) / \sum_{c \in \mathsf{ch}(n)} F_{\mathcal{D}}(n,c), \qquad \text{where} \quad F_{\mathcal{D}}(n,c) := \sum_{i=1}^{N} \mathbb{1}[\mathbf{x}_i \in \gamma_n \cap \gamma_c]. \tag{3}$$

The quantity $F_{\mathcal{D}}(n,c)$ is called the *circuit flow* of edge (n,c). Intuitively, circuit flows count the number of samples in $\mathcal{D}$ that "activate" an edge.

However, since HCLTs are not deterministic with respect to the observed variables $\mathbf{X}$, MLE does not have a closed-form expression, and we instead resort to Expectation-Maximization (EM), where in the E step we compute the *expected circuit flow* given incomplete data, and in the M step we estimate the closed-form MLE parameters given those expected flows [66, 56].

Concretely, given a deterministic PC $(\mathcal{G}, \boldsymbol{\theta})$ with root node r and an incomplete dataset $\mathcal{D} = \{\mathbf{x}_i\}_{i=1}^N$, the parameters for the next EM iteration are given by [56]:

$$\theta^{(\text{new})}_{n,c} = \mathrm{EF}_{\mathcal{D},\boldsymbol{\theta}}(n,c) / \sum_{c \in \mathsf{ch}(n)} \mathrm{EF}_{\mathcal{D},\boldsymbol{\theta}}(n,c), \qquad \text{where} \quad \mathrm{EF}_{\mathcal{D},\boldsymbol{\theta}}(n,c) := \sum_{i=1}^{N} \mathbb{E}_{\mathbf{z} \sim \Pr_r(\cdot|\mathbf{x}_i)} \big[\mathbb{1}[\mathbf{z}\mathbf{x}_i \in \gamma_n \cap \gamma_c]\big] \tag{4}$$

defines an expected version of circuit flow for edge (n,c) given samples with missing values in $\mathcal{D}$.

Using their equivalent PC representations, HCLTs can be trained efficiently using the PC package PyJuice [40]. By representing graphical models (e.g., HCLTs) as PCs, we take advantage of the structure of the model to extensively parallelize the computation required by EM updates (Equation 4). Further, we develop specialized GPU kernels to significantly speed up the EM algorithm. As a result, despite the model having over 88 million parameters, a single EM epoch on the 1KG dataset can be done in less than 2 seconds.

4.3 Impute5 Specifications

Haploid imputation is specified in Impute5, with all other SNPs in the region serving as buffer (observed context). Further, we provide the corresponding fine-scale human recombination map for the chromosome being imputed. These are linked in the Impute5 documentation: https://github.com/odelaneau/shapeit4/tree/master/maps.

4.4 Training Specifications

Training GPC is straightforward, requiring minimal hyperparameter tuning. We set the number of latent states (for each hidden variable in the HCLT) to 128 for both datasets – the maximum feasible given memory constraints. We observe that changing the pseudocount parameter (used to smooth probability estimates) does not significantly impact the training procedure and can be left at a small default value (0.005).

Based on initial experiments with small validation sets, we find that training for 2,000–5,000 epochs works well for these datasets, though longer training is possible without overfitting (with minimal performance gains). For the 1KG data, we trained GPC for 5,000 epochs. Due to the larger sample size (and longer training time per epoch) in the UKBB data, we stopped at 2,000 epochs. Although we do not tune the pseudocount, this would be the only parameter to consider adjusting for further optimization. Additionally, since we use full-batch EM to learn the circuit parameters, we do not need to tune other standard hyperparameters such as learning rate or batch size.

A key advantage of GPC over other deep generative approaches is that we can probabilistically determine convergence by monitoring held-out log-likelihood, which provides an objective, quantitative stopping criterion. This stands in contrast to methods that lack tractable likelihoods, where convergence must be assessed through slower, less consistent visual inspection methods that are more prone to human error.

The baseline RBM and WGAN methods require substantially more time and experimentation to tune, primarily due to their larger hyperparameter spaces and inability to calculate likelihoods or determine convergence probabilistically. For RBMs, the partition function is intractable due to an exponentially large number of configurations over the visible and hidden nodes. Therefore, we must rely on indirect metrics such as Nearest Neighbor Adversarial Accuracy (AATS) [45] and visual overlap in principal component space. Similarly, WGANs do not define a probability distribution over the data, preventing direct evaluation of sample likelihoods. Here too, we must rely on visual overlap in the principal component space to assess convergence, a subjective and time-consuming process.

We utilize the training code and notebooks provided by the authors on GitHub to tune both methods. Although we initially struggled to achieve optimal performance on the 1KG African data subset, we resolved this issue after consulting with the authors. For the RBM, we modified several parameters (e.g., higher learning rate, fewer epochs, fewer Gibbs sampling steps). For the WGAN, we increased the batch size. Otherwise, all other parameters were kept at their default values. When training on the UKBB data, we retained the default parameters; similar to the GPC case, we used fewer epochs due to longer training time but still achieved good overlap in the principal component space based on visual inspection.

The simpler PGM baselines, Indep and Markov, model the observed SNPs directly without latent variables. Indep treats each SNP as an independent Bernoulli random variable, estimating $P(X_n = 1)$ separately for each site via its sample mean (calculated by a single pass over the training data). Markov is a first-order Markov chain: it models each SNP conditioned only on the immediately preceding one in genomic order, $P(X_n \mid X_{n-1})$, and is likewise fit in closed form via maximum likelihood. Both models therefore require no iterative training. For HMM, we use the EM algorithm with random initialization. Similar to GPC, we set the number of latent states for HMM at 128 and train for 2,000–5,000 epochs depending on the dataset. Using these settings, the held-out log-likelihood converged effectively. We tried higher values of latent states for HMM but these all resulted in worse log-likelihoods and slower training overall.

4.5 Evaluation Metrics

4.5.1 Nearest Neighbor Adversarial Accuracy

To evaluate the utility and privacy of the AGs, we compute the AATS metric introduced by Yale et al. [45]:

$$\mathcal{AATS} = \frac{1}{2}\left(\frac{1}{n}\sum_{i=1}^{n}\mathbf{1}\left(d_{TS}(i) > d_{TT}(i)\right) + \frac{1}{n}\sum_{i=1}^{n}\mathbf{1}\left(d_{ST}(i) > d_{SS}(i)\right)\right) \tag{5}$$

T represents the true (real) data and S represents the synthetic data. $d_{TS}(i)$ is the distance between sample i in the true data and its nearest neighbor in the synthetic data, while $d_{TT}(i)$ is the distance between sample i in the true data and its nearest neighbor in the true data (not including itself). $d_{ST}(i)$ and $d_{SS}(i)$ are defined analogously for the synthetic data. Following Yale et al. [45], we use Euclidean distance. The $\mathcal{AATS}$ is an average of two components, which we denote AA_{TRUTH} and AA_{SYN}:

$$\underbrace{\frac{1}{n}\sum_{i=1}^{n}\mathbf{1}\left(d_{TS}(i) > d_{TT}(i)\right)}_{AA_{\text{TRUTH}}} \qquad \underbrace{\frac{1}{n}\sum_{i=1}^{n}\mathbf{1}\left(d_{ST}(i) > d_{SS}(i)\right)}_{AA_{\text{SYN}}} \tag{6}$$

Each component returns a value between 0 and 1, with values near 0.5 indicating the ideal balance between utility and privacy. We do not report the overall $\mathcal{AATS}$ because poor values in each component can cancel each other out: for instance, an AA_{TRUTH} of 0 paired with an AA_{SYN} of 1 would average to 0.5, masking both extreme memorization and a collapsed synthetic distribution.

Experimental setup. Using a 50/50 train/test split, we construct four datasets of equal size n: a real training set R_{tr}, a real test set R_{te}, and two disjoint halves of the generated synthetic data, A_1 and A_2. We compute the AATS components twice: pairing (R_{tr}, A_1) for train metrics and (R_{te}, A_2) for test metrics. For 1KG, $n = 2{,}504$, matching each real split exactly. For UKBB, the real dataset has 13,462 samples per split but we generated only 10,000 synthetic samples total, so we set $n = 5{,}000$: A_1 and A_2 are the first and second halves of the 10,000 generated samples, and 5,000 haplotypes are drawn without replacement from each real split using a fixed random seed.

Standardization. For each SNP, we compute the mean μ_j and standard deviation σ_j from R_{tr} only, and apply the transformation $x \mapsto (x - \mu_j)/\sigma_j$ to all four datasets (R_{tr}, R_{te}, A_1, A_2) before computing any distances. Monomorphic sites ($\sigma_j = 0$) are left unchanged. Fitting the scaler on R_{tr} and applying it uniformly ensures that no information from the test set or the synthetic data influences the scaling. This choice is motivated by the fact that standardization reweights each SNP's contribution to Euclidean distance by $1/\sigma_j \propto 1/\sqrt{p_j(1-p_j)}$, amplifying the contribution of rare variants (low MAF p_j) and making the metric sensitive to failures in rare-variant modeling that would otherwise go undetected.

4.5.2 Wasserstein Distance Calculation

To quantify differences between real and generated data, we used Wasserstein distances in two complementary settings. For comparisons in PCA space, we reported a two-dimensional Wasserstein distance computed between the empirical distributions of real (test set) and generated individuals. For each generative method, PCA was fit jointly on the real test samples and the corresponding generated samples ("coupled PCA"), and Wasserstein distances were evaluated in selected two-dimensional PCA subspaces (PC1–PC2, PC3–PC4,

and PC5–PC6). Distances were computed using the entropically regularized optimal transport (Sinkhorn) algorithm as implemented in the `ot` Python package, with a regularization parameter set to $\varepsilon = 2 \times 10^{-3}$. Lower values indicate closer agreement between the distributions of real and generated samples in PCA space.

To assess similarity at the haplotype level, we computed one-dimensional Wasserstein distances between distributions of pairwise haplotypic distances. For each pair of haplotypes, we computed the Manhattan (cityblock) distance, which for haploid genomes equals the number of differing SNP positions, *i.e.*, Hamming distance. We define two comparison metrics, both using the distribution of pairwise distances within the real test set (real–real) as the reference. The *within* metric compares internal diversity: it measures the Wasserstein distance between the distribution of all pairwise distances among AGs (generated–generated; $N_G(N_G - 1)/2$ values for N_G AGs) and the distribution among real test haplotypes (real–real; $N_R(N_R - 1)/2$ values for N_R real genomes). This quantifies whether generated samples exhibit similar diversity to real samples. The *between* metric compares the distribution of cross-distances between generated and real test haplotypes (generated–real; $N_G \times N_R$ values) to the same real–real reference distribution. This quantifies whether generated haplotypes are as close to real haplotypes as real haplotypes are to each other. One-dimensional Wasserstein distances were computed using `scipy.stats.wasserstein_distance`, which handles distributions with different numbers of samples by treating them as empirical distributions.

Acknowledgements

This work was supported, in part, by NIH grant GM153406, by National University of Singapore under its Start-up Grant SUG-251RES250, by the DARPA ANSR, CODORD, and SAFRON programs under awards FA8750-23-2-0004, HR00112590089, and HR00112530141, NSF grant IIS1943641, and gifts from Adobe Research, Cisco Research, Qualcomm, and Amazon. Approved for public release; distribution is unlimited. This research was conducted using the UK Biobank Resource under application 331277. We thank the participants of UK Biobank for making this work possible. We also thank the authors of [25], Burak Yelmen and Aurélien Decelle, for their assistance with training and parameter selection for the WGAN and RBM models.

Code availability

Code and experiments are available at https://github.com/sriramlab/GPC.

Data Availability

The 1000 Genomes datasets can be freely downloaded at https://www.internationalgenome.org/data. The UK Biobank dataset is available upon application at https://www.ukbiobank.ac.uk/.

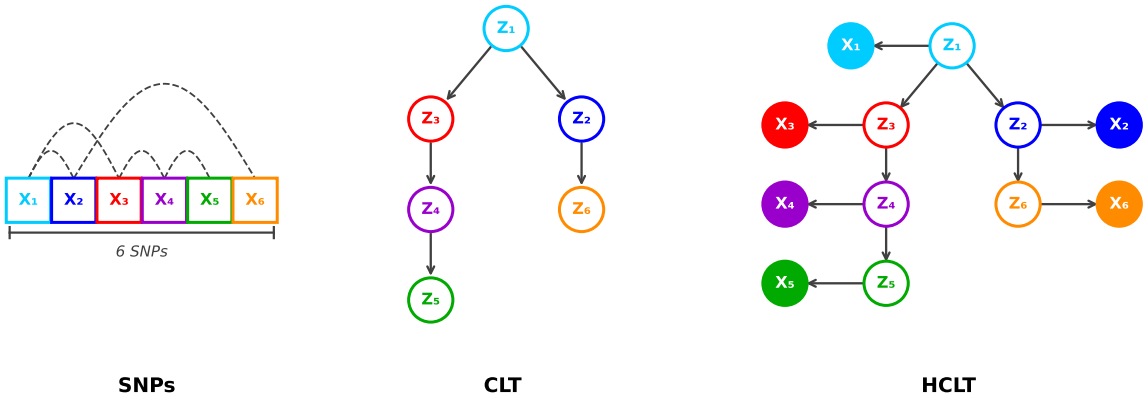


Figure 1: **Generating HCLT structures given genetic data. SNPs:** An example dataset of 6 SNPs $(X_1, \ldots, X_6)$, where dashed lines indicate SNPs with strong correlations. **CLT:** A Chow–Liu Tree learned from the SNP data, represented as a Bayesian network over latent variables $Z_1, \ldots, Z_6$. The tree structure places latent variables with stronger dependencies closer together, capturing the dominant correlation structure among the SNPs. **HCLT:** The Hidden CLT constructed from the CLT, in which each latent variable Z_i is additionally connected to its corresponding observed SNP X_i as a child node. The resulting structure respects the dependencies among SNPs while providing a principled generative model over the observed data.

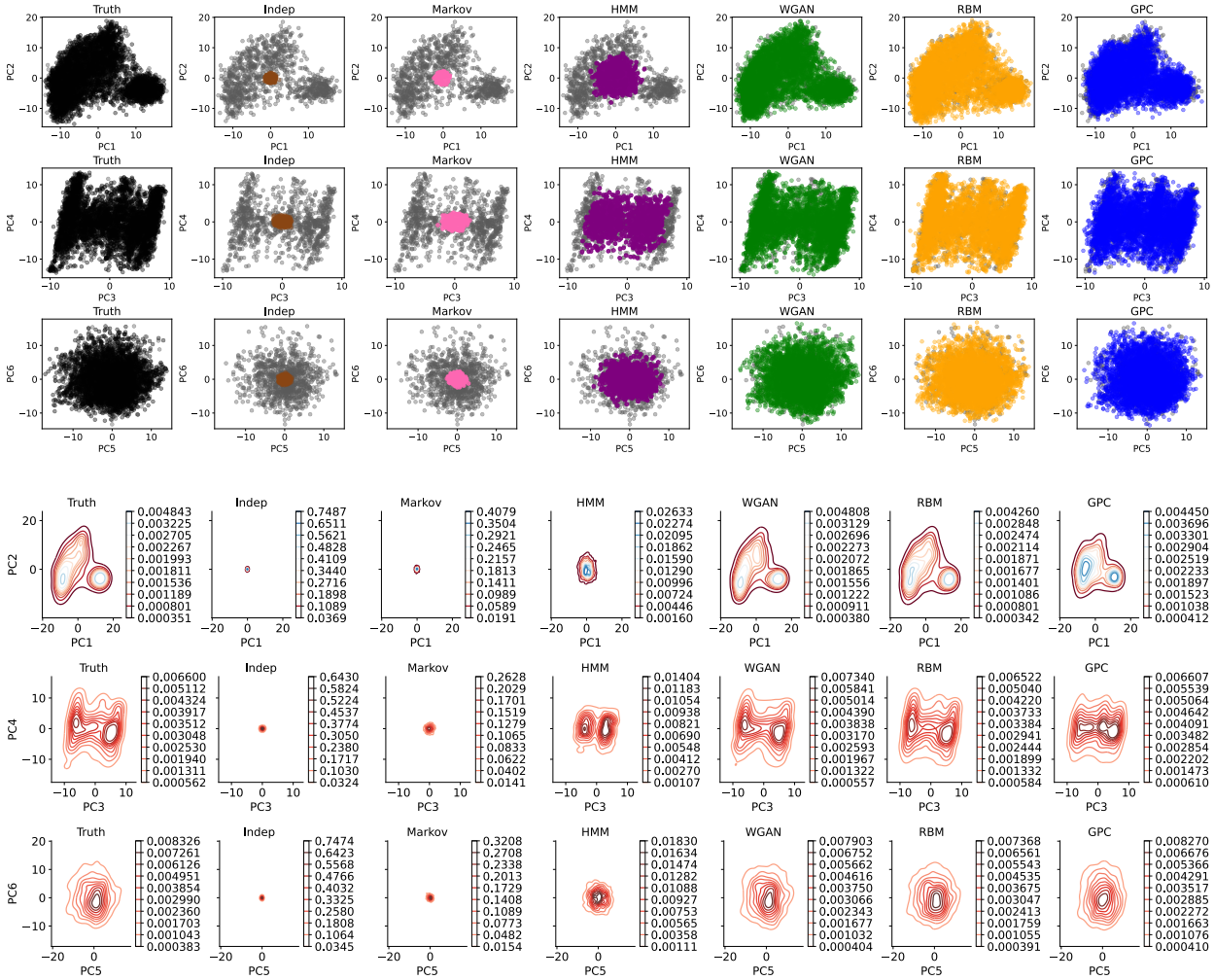
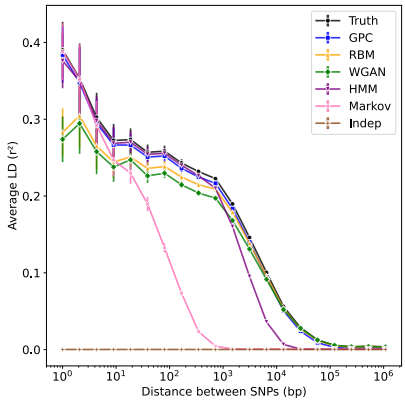
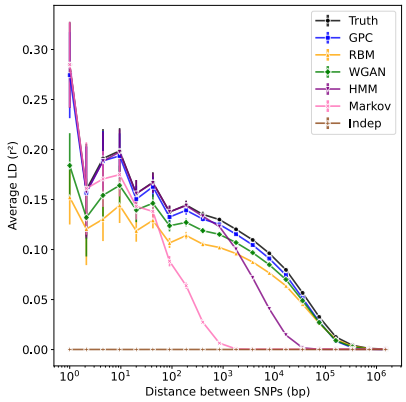


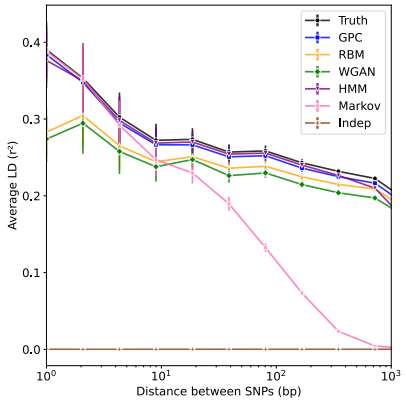
Figure 2: **Principal components analysis for AGs generated by models trained on the 1KG dataset**. (Top half) The top six principal components (PCs) of the held-out test set of genomes (gray) vs. AGs generated by each of Indep (brown), Markov (pink), HMM (purple), WGAN (green), RBM (orange), and GPC (blue). The leftmost plot shows the PCs of genomes from the training set as "perfectly" generated data (black). (Bottom half) A density map of the PCs for each dataset.



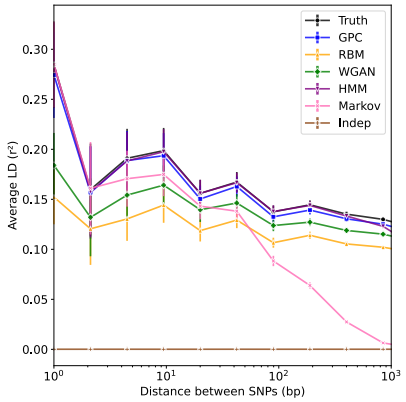
(a) 1KG: full range



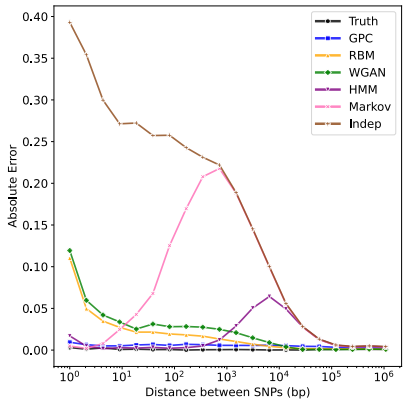
(b) UKBB: full range



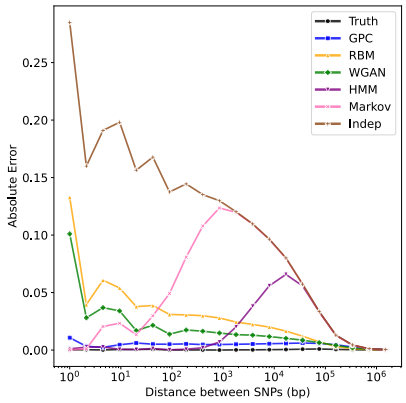
(c) 1KG: 1–1000 bp zoom



(d) UKBB: 1–1000 bp zoom



(e) 1KG: absolute error per bin



(f) UKBB: absolute error per bin

Figure 3: **Patterns of linkage disequilibrium (LD) in the AGs generated by models.** We compare LD betweens pairs of SNPs in the AGs generated by each of the models trained on the 1KG (left) and the UKBB (right) datasets respectively. We plot mean squared correlation r^2 within bins of physical distance (error bars denote the standard error of the mean); truth represents the LD distribution of the training data. We plot the decay of LD across the full length scale of the locus (top row); at shorter length scales (1–1000 bp; middle row); and the absolute error in mean r^2 per bin relative to the test data (bottom row) (see Table S2 for detailed metrics).

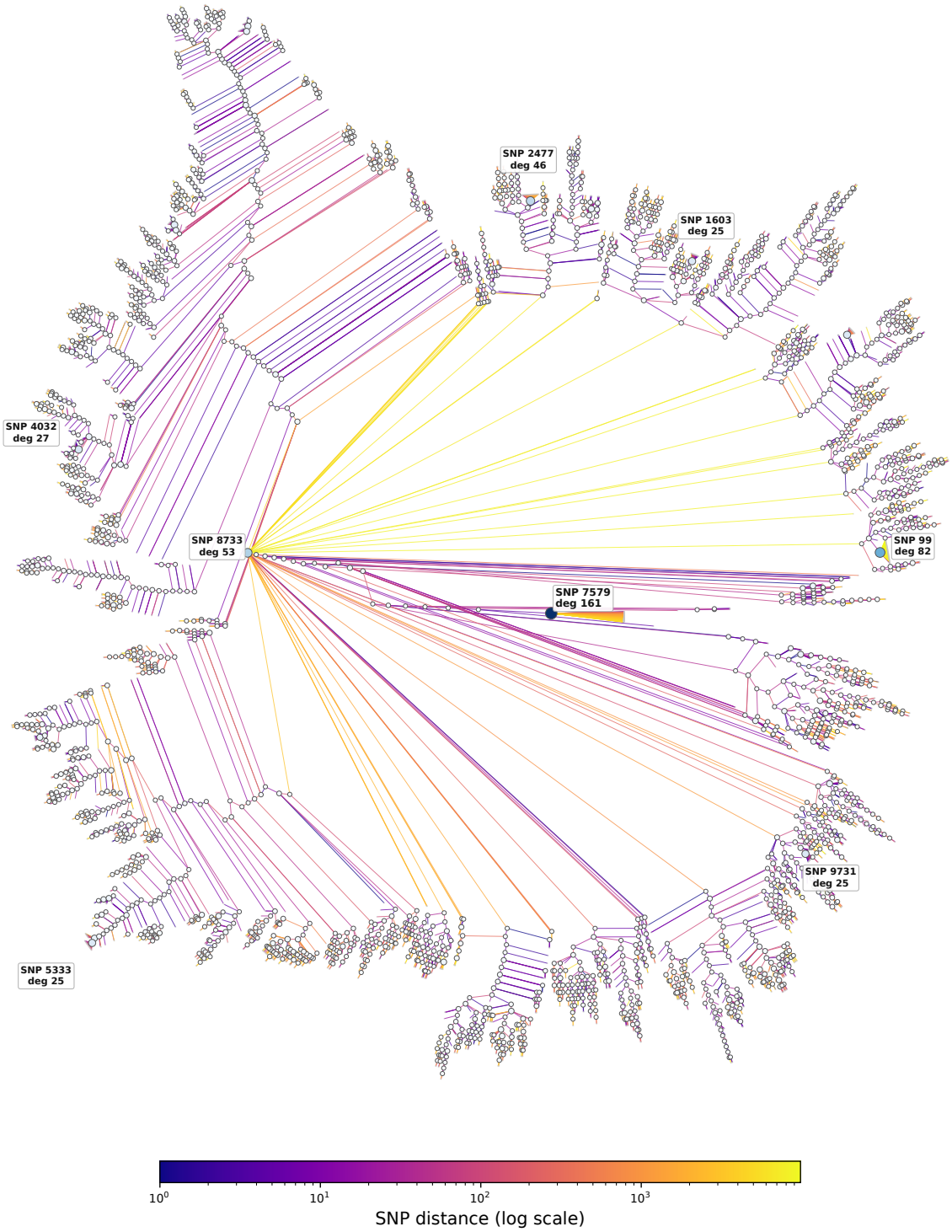
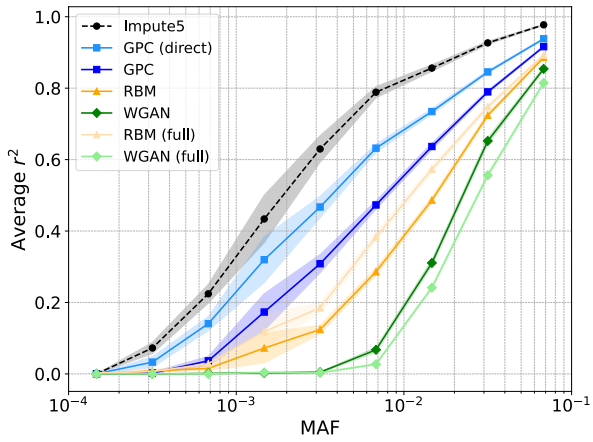
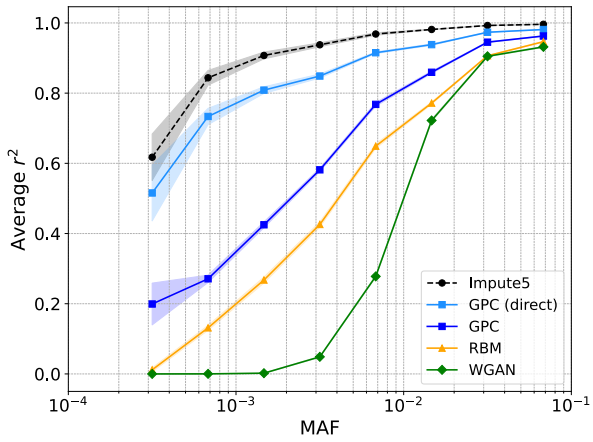


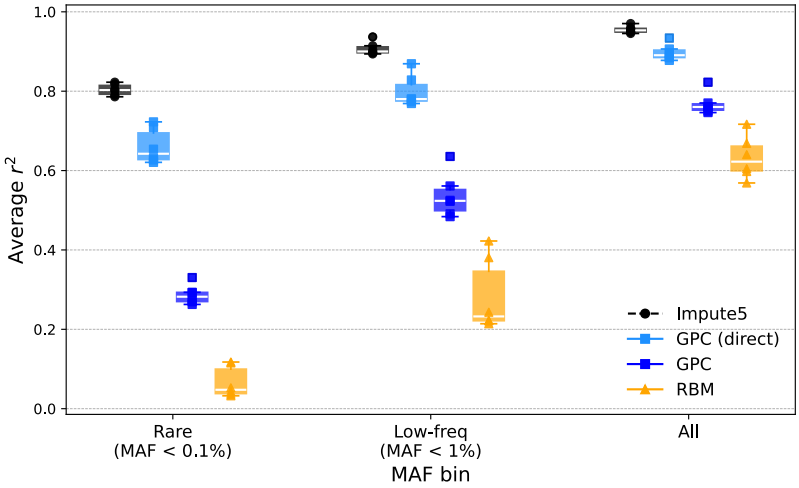
Figure 4: **Compact radial visualization of the Chow–Liu Tree (CLT) learned from the 1KG dataset.** Starting from the full 10,000-SNP tree (Supplementary Figure S4), all degree-2 nodes forming unbranched chains are contracted into single edges between their endpoints, retaining only branching points and leaves. Edge color encodes genomic distance $|i-j|$ on a log scale (purple: nearby; yellow: distant). Node size scales with degree; the top 8 highest-degree nodes are labeled. Unlike an HMM, which forms a single unbranched chain, the CLT exhibits rich branching structure with long-range edges connecting SNPs thousands of positions apart, capturing distal LD directly rather than routing it through all intermediate variables.



(a) 1KG

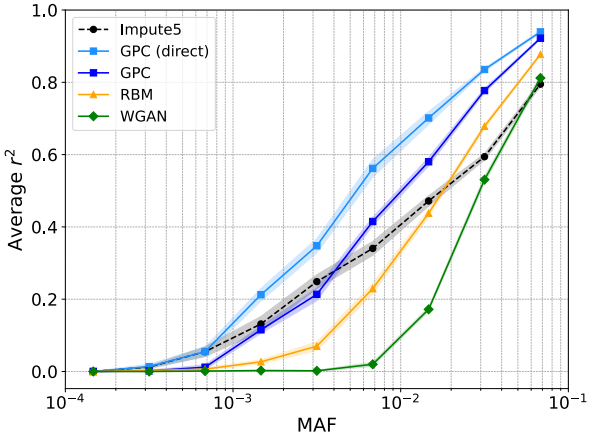


(b) UKBB

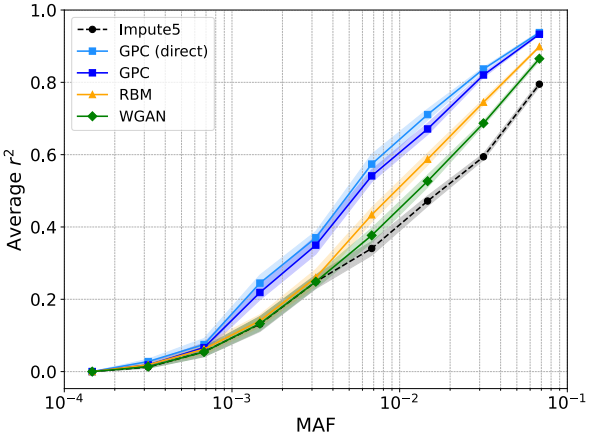


(c) Summary across UKBB regions

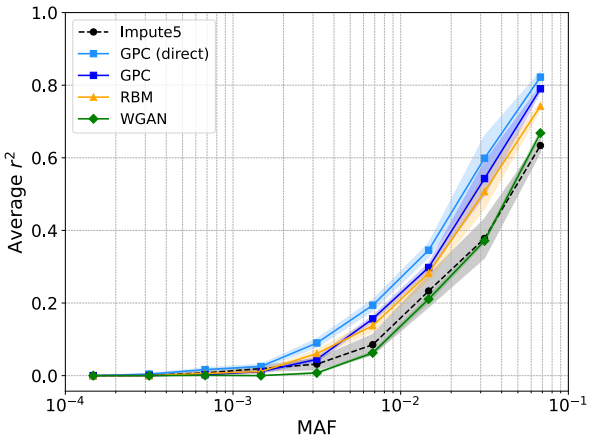
Figure 5: **General imputation.** The light blue lines denote direct imputation using GPC while the other lines show the results of imputation using Impute5 with AG reference panels in the 1KG (left) and the UKBB (right) datasets respectively. We also plot the results of imputation using real reference genomes (black dotted lines) while RBM/WGAN (full) show results using AGs provided by the authors of [25] in their analysis of the same locus in the 1KG dataset (see Table S3 for detailed metrics). (c) summarizes UKBB imputation performance for the best-performing methods (excludes WGAN) by MAF bin, averaged across chr22 (the main UKBB region shown in (b)) and five additional chromosomes (chr2, 11, 12, 14, 18; see Supplementary Figure S5), demonstrating that results are robust across genomic regions.



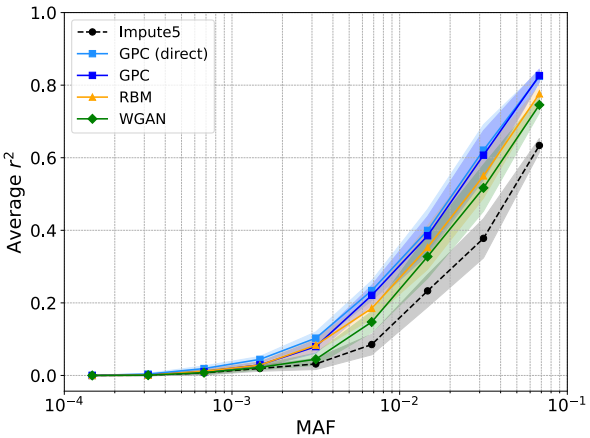
(a) Non-European target



(b) Non-European target



(c) African target



(d) African target

Figure 6: **Population-specific imputation in the 1KG dataset.** On the left, the AG lines show imputation results using population-specific AGs (where the target genomes correspond to non-European populations in part a and African populations in part c) while the light blue line shows direct imputation with GPC trained on population-specific genomes. On the right, the AG lines show results using combined reference panels (real European genomes plus population-specific AGs) while the light blue line shows direct imputation with GPC trained on both European and population-specific genomes (where the target genomes correspond to non-European populations in part b and African populations in part d). The black line shows results using Impute5 with real European genomes as the reference panel (see Table S5 for detailed metrics).

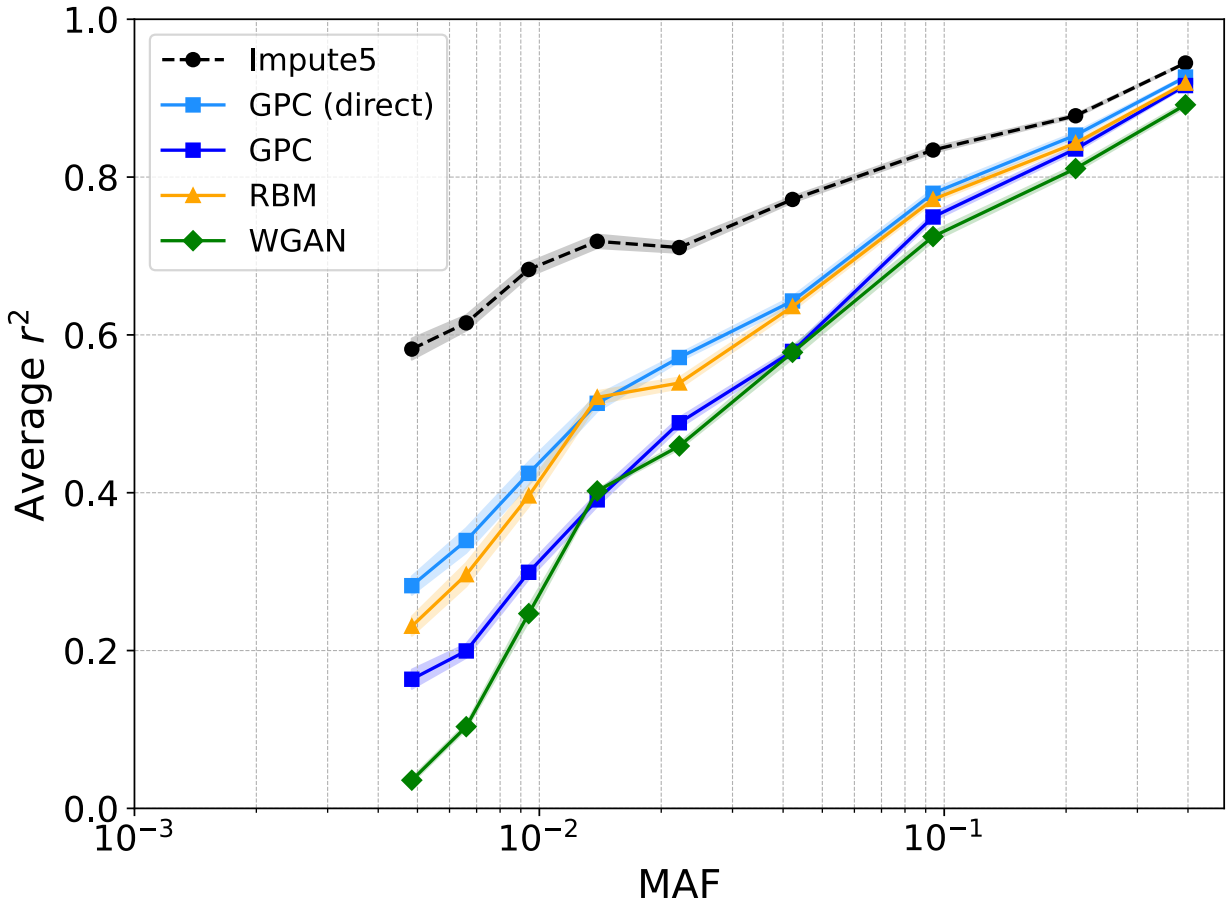
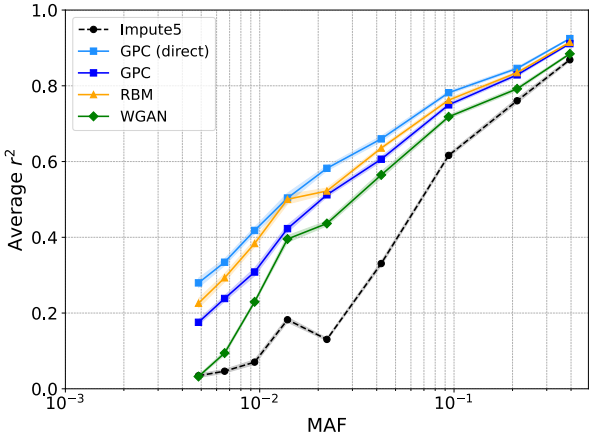
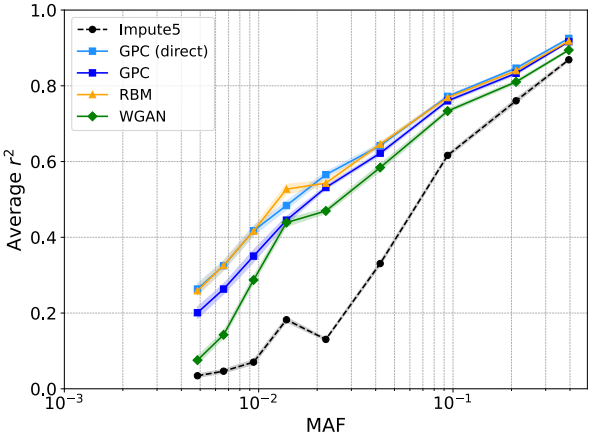


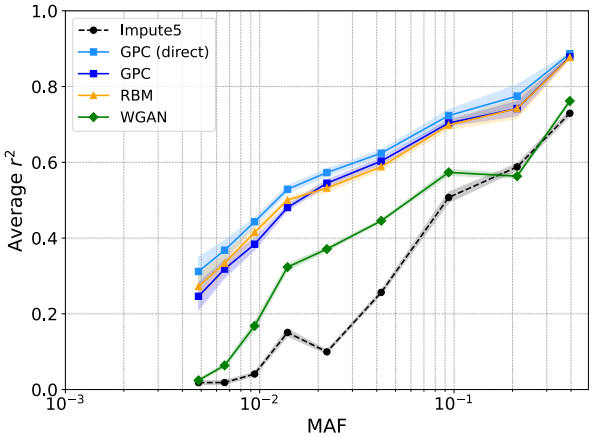
Figure 7: **Array-based imputation.** This task aims to impute SNPs from the HumanOmni5Exome array using high-coverage 1KG data. The black and light blue lines show results using Impute5 with real training data as the reference panel and direct imputation using GPC, respectively. The other lines show results using Impute5 with AG reference panels (see Table S7 for detailed metrics).



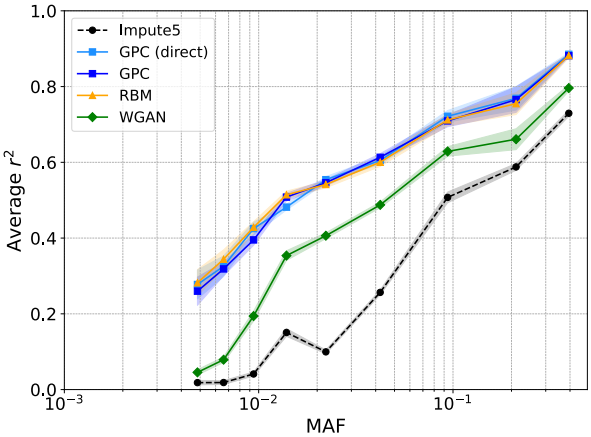
(a) Non-European target



(b) Non-European target



(c) African target



(d) African target

Figure 8: **Array-based imputation (population-specific).** The black line shows results using Impute5 with real European reference genomes. On the left (parts a and c), the AG lines show results using population-specific AGs alone, and the light blue line shows direct imputation with GPC trained on population-specific data. On the right (parts b and d), the AG lines show results using combined reference panels (real European data plus population-specific AGs), and the light blue line shows direct imputation with GPC trained on both European and population-specific data (see Table S8 for detailed metrics).

Dataset	Category	Indep	Markov	HMM	GPC
1KG	**train LL**	-2386.81	-1806.33	-591.08	**-202.51**
	test LL	-2404.51	-1819.96	-599.88	**-265.06**
UKBB	**train LL**	-1642.62	-1360.86	-554.88	**-120.10**
	test LL	-1648.03	-1362.16	-554.38	**-127.75**

Table 1: **Log-likelihoods of probabilistic models that support tractable likelihood computation in 1KG and UKBB datasets.** Averaged training and test log-likelihoods for Indep, Markov HMM, and GPC. The bold values highlight the best averaged log-likelihoods.

Dataset	Metric	RBM	WGAN	GPC
1KG	AA_{TRUTH} (Train)	0.9561	0.8103	**0.7185**
	AA_{TRUTH} (Test)	0.9916	**0.8183**	0.8199
	AA_{SYN} (Train)	0.0024	0.7356	**0.4225**
	AA_{SYN} (Test)	0.0212	0.7680	**0.4912**
UKBB	AA_{TRUTH} (Train)	0.9934	0.9664	**0.9096**
	AA_{TRUTH} (Test)	0.9970	0.9692	**0.9240**
	AA_{SYN} (Train)	0.0094	0.7828	**0.5380**
	AA_{SYN} (Test)	0.0114	0.7730	**0.4686**

Table 2: **Nearest neighbor adversarial accuracy (AA_{TRUTH}, AA_{SYN})** for AGs generated by RBM, WGAN, and GPC, evaluated on training and test splits from 1KG and UKBB. Distances are standardized Euclidean. This upweights rare variants relative to common ones, exposing failures in rare-variant modeling. Values closest to 0.5 are bolded.

References

[1] Magnus Nordborg. Coalescent theory. *Handbook of Statistical Genomics: Two Volume Set*, pages 145–30, 2019.

[2] John Wakeley. Developments in coalescent theory from single loci to chromosomes. *Theoretical population biology*, 133:56–64, 2020.

[3] Jonathan Marchini and Bryan Howie. Genotype imputation for genome-wide association studies. *Nature Reviews Genetics*, 11(7):499–511, 2010.

[4] Sharon R Browning and Brian L Browning. Haplotype phasing: existing methods and new developments. *Nature Reviews Genetics*, 12(10):703–714, 2011.

[5] Daniel Mas Montserrat, Carlos Bustamante, and Alexander Ioannidis. Class-conditional vae-gan for local-ancestry simulation. *arXiv preprint arXiv:1911.13220*, 2019.

[6] Richard R Hudson. Generating samples under a wright–fisher neutral model of genetic variation. *Bioinformatics*, 18(2):337–338, 2002.

[7] Garrett Hellenthal and Matthew Stephens. mshot: modifying hudson's ms simulator to incorporate crossover and gene conversion hotspots. *Bioinformatics*, 23(4), 2007.

[8] Laurent Excoffier and Matthieu Foll. Fastsimcoal: a continuous-time coalescent simulator of genomic diversity under arbitrarily complex evolutionary scenarios. *Bioinformatics*, 27(9):1332–1334, 2011.

[9] Jerome Kelleher, Alison M. Etheridge, and Gilean McVean. Efficient Coalescent Simulation and Genealogical Analysis for Large Sample Sizes. *PLoS Computational Biology*, 12(5):1–22, 2016.

[10] Franz Baumdicker, Gertjan Bisschop, Daniel Goldstein, Graham Gower, Aaron P Ragsdale, Georgia Tsambos, Sha Zhu, Bjarki Eldon, Castedo E Ellerman, Jared G Galloway, et al. Efficient ancestry and mutation simulation with msprime 1.0. *bioRxiv*, 2021.

[11] Richard R Hudson. Properties of a neutral allele model with intragenic recombination. *Theoretical Population Biology*, 23(2):183–201, 1983.

[12] Robert C. Griffiths and Paul Marjoram. An Ancestral Recombination Graph. In P. Donnelly and S. Tavare, editors, *Progress in Population Genetics and Human Evolution, IMA Volumes in Mathematics and its Applications, vol. 87*, pages 257–270. Springer, 1997.

[13] Carsten Wiuf and Jotun Hein. Recombination as a point process along sequences. *Theoretical Population Biology*, 55(3):248–259, 1999.

[14] Gilean A T McVean and Niall J Cardin. Approximating the coalescent with recombination. *Philosophical transactions of the Royal Society of London. Series B, Biological sciences*, 360(1459):1387–93, 2005.

[15] Paul Marjoram and Jeff D. Wall. Fast "coalescent" simulation. *BMC Genetics*, 7(1):16, 2006.

[16] Na Li and Matthew Stephens. Modeling linkage disequilibrium and identifying recombination hotspots using single-nucleotide polymorphism data. *Genetics*, 165(4):2213–2233, 2003.

[17] Lawrence Rabiner and Biinghwang Juang. An introduction to hidden markov models. *ieee assp magazine*, 3(1):4–16, 1986.

[18] Paul Scheet and Matthew Stephens. A fast and flexible statistical model for large-scale population genotype data: applications to inferring missing genotypes and haplotypic phase. *The American Journal of Human Genetics*, 78(4):629–644, 2006.

[19] Olivier Delaneau, Jonathan Marchini, and Jean-François Zagury. A linear complexity phasing method for thousands of genomes. *Nature methods*, 9(2):179–181, 2012.

[20] Bryan Howie, Christian Fuchsberger, Matthew Stephens, Jonathan Marchini, and Gonçalo R Abecasis. Fast and accurate genotype imputation in genome-wide association studies through pre-phasing. *Nature genetics*, 44(8):955–959, 2012.

[21] Yael Baran, Bogdan Pasaniuc, Sriram Sankararaman, Dara G Torgerson, Christopher Gignoux, Celeste Eng, William Rodriguez-Cintron, Rocio Chapela, Jean G Ford, Pedro C Avila, et al. Fast and accurate inference of local ancestry in latino populations. *Bioinformatics*, 28(10):1359–1367, 2012.

[22] Alkes L Price, Arti Tandon, Nick Patterson, Kathleen C Barnes, Nicholas Rafaels, Ingo Ruczinski, Terri H Beaty, Rasika Mathias, David Reich, and Simon Myers. Sensitive detection of chromosomal segments of distinct ancestry in admixed populations. *PLoS genetics*, 5(6):e1000519, 2009.

[23] CJ Battey, Gabrielle C Coffing, and Andrew D Kern. Visualizing population structure with variational autoencoders. *G3*, 11(1):1–11, 2021.

[24] Burak Yelmen, Aurélien Decelle, Linda Ongaro, Davide Marnetto, Corentin Tallec, Francesco Montinaro, Cyril Furtlehner, Luca Pagani, and Flora Jay. Creating artificial human genomes using generative neural networks. *PLOS Genetics*, 17(2):1–22, 02 2021.

[25] Burak Yelmen, Aurélien Decelle, Leila Lea Boulos, Antoine Szatkownik, Cyril Furtlehner, Guillaume Charpiat, and Flora Jay. Deep convolutional and conditional neural networks for large-scale genomic data generation. *PLOS Computational Biology*, 19(10):1–21, 10 2023.

[26] Antoine Szatkownik, Cyril Furtlehner, Guillaume Charpiat, Burak Yelmen, and Flora Jay. Towards creating longer genetic sequences with gans: Generation in principal component space. In David A. Knowles and Sara Mostafavi, editors, *Proceedings of the 18th Machine Learning in Computational Biology meeting*, volume 240 of *Proceedings of Machine Learning Research*, pages 110–122. PMLR, 30 Nov–01 Dec 2024.

[27] Rebecca Riley, Iain Mathieson, and Sara Mathieson. Interpreting generative adversarial networks to infer natural selection from genetic data. *Genetics*, 226(4):iyae024, 02 2024.

[28] Shaked Ahronoviz and Ilan Gronau. Genome-AC-GAN: Enhancing synthetic genotype generation through auxiliary classification. bioRxiv, Feb 2024.

[29] Margarita Geleta, Daniel Mas Montserrat, Xavier Giro-i Nieto, and Alexander G. Ioannidis. Autoencoders for genomic variation analysis. *Genome Research*, 36:348–360, Jan 2026.

[30] Callum Burnard, Alban Mancheron, and William Ritchie. Generating realistic artificial human genomes using adversarial autoencoders. *NAR Genomics and Bioinformatics*, 7(3):lqaf101, Jul 2025.

[31] Philip Kenneweg, Raghuram Dandinasivara, Xiao Luo, Barbara Hammer, and Alexander Schönhuth. Generating synthetic genotypes using diffusion models. *Bioinformatics*, 41(Supplement_1):i484–i492, Jul 2025.

[32] Simone Rubinacci, Olivier Delaneau, and Jonathan Marchini. Genotype imputation using the positional burrows wheeler transform. *PLOS Genetics*, 16(11):1–19, 11 2020.

[33] Brian L. Browning, Ying Zhou, and Sharon R. Browning. A one-penny imputed genome from next-generation reference panels. *The American Journal of Human Genetics*, 103(3):338–348, Sep 2018.

[34] Christian Fuchsberger, Gonçalo R. Abecasis, and David A. Hinds. Minimac2: Faster genotype imputation. *Bioinformatics*, 31(5):782–784, Oct 2014.

[35] Anji Liu and Guy Van den Broeck. Tractable regularization of probabilistic circuits. In *Advances in Neural Information Processing Systems 35 (NeurIPS)*, dec 2021.

[36] Jonathan K Pritchard and Molly Przeworski. Linkage disequilibrium in humans: models and data. *The American Journal of Human Genetics*, 69(1):1–14, 2001.

[37] Antonio Vergari, YooJung Choi, Robert Peharz, and Guy Van den Broeck. Probabilistic circuits: Representations, inference, learning and applications. *AAAI Tutorial*, 2020.

[38] YooJung Choi, Antonio Vergari, and Guy Van den Broeck. Probabilistic circuits: A unifying framework for tractable probabilistic models. oct 2020.

[39] C. K. Chow and C. N. Liu. Approximating discrete probability distributions with dependence trees. *IEEE Transactions on Information Theory*, 14:462–467, 1968.

[40] Anji Liu, Kareem Ahmed, and Guy Van den Broeck. Scaling tractable probabilistic circuits: A systems perspective. In *Proceedings of the 41th International Conference on Machine Learning (ICML)*, jul 2024.

[41] Adam Auton, Gonçalo R. Abecasis, David M. Altshuler, Richard M. Durbin, Gonçalo R. Abecasis, David R. Bentley, Aravinda Chakravarti, Andrew G. Clark, Peter Donnelly, Evan E. Eichler, and et al. A global reference for human genetic variation. *Nature*, 526(7571):68–74, Sep 2015.

[42] Clare Bycroft, Colin Freeman, Desislava Petkova, Gavin Band, Lloyd T. Elliott, Kevin Sharp, Allan Motyer, Damjan Vukcevic, Olivier Delaneau, Jared O'Connell, and et al. The uk biobank resource with deep phenotyping and genomic data. *Nature*, 562(7726):203–209, Oct 2018.

[43] Marta Byrska-Bishop, Uday S. Evani, Xuefang Zhao, Anna O. Basile, Haley J. Abel, Allison A. Regier, André Corvelo, Wayne E. Clarke, Rajeeva Musunuri, Kshithija Nagulapalli, Susan Fairley, Alexi Runnels, Lara Winterkorn, Ernesto Lowy, Evan E. Eichler, Jan O. Korbel, Charles Lee, Tobias Marschall, Scott E. Devine, William T. Harvey, Weichen Zhou, Ryan E. Mills, Tobias Rausch, Sushant Kumar, Can Alkan, Fereydoun Hormozdiari, Zechen Chong, Yu Chen, Xiaofei Yang, Jiadong Lin, Mark B. Gerstein, Ye Kai, Qihui Zhu, Feyza Yilmaz, Chunlin Xiao, Paul Flicek, Soren Germer, Harrison Brand, Ira M. Hall, Michael E. Talkowski, Giuseppe Narzisi, and Michael C. Zody. High-coverage whole-genome sequencing of the expanded 1000 genomes project cohort including 602 trios. *Cell*, 185(18):3426–3440.e19, 2022.

[44] Ketian Yu, Sayantan Das, Jonathon LeFaive, Alan Kwong, Jacob Pleiness, Lukas Forer, Sebastian Schönherr, Christian Fuchsberger, Albert Vernon Smith, and Gonçalo Rocha Abecasis. Meta-imputation: An efficient method to combine genotype data after imputation with multiple reference panels. *The American Journal of Human Genetics*, 109(6):1007–1015, Jun 2022.

[45] Andrew Yale, Saloni Dash, Ritik Dutta, Isabelle Guyon, Adrien Pavao, and Kristin P. Bennett. Generation and evaluation of privacy preserving synthetic health data. *Neurocomputing*, 416:244–255, 2020.

[46] Reza Shokri, Marco Stronati, Congzheng Song, and Vitaly Shmatikov. Membership inference attacks against machine learning models. In *2017 IEEE Symposium on Security and Privacy (SP)*, pages 3–18. IEEE, 2017.

[47] Nils Homer, Szabolcs Szelinger, Margot Redman, David Duggan, Waibhav Tembe, Jill Muehling, John V Pearson, Dietrich A Stephan, Stanley F Nelson, and David W Craig. Resolving individuals contributing trace amounts of dna to highly complex mixtures using high-density snp genotyping microarrays. *PLoS Genetics*, 4(8):e1000167, 2008.

[48] Matt Fredrikson, Somesh Jha, and Thomas Ristenpart. Model inversion attacks that exploit confidence information and basic countermeasures. In *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*, pages 1322–1333, 2015.

[49] Mijung Park, Jimmy Foulds, Kamalika Chaudhuri, and Max Welling. Dp-em: Differentially private expectation maximization, 2016.

[50] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014.

[51] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[52] Mahsan Nourani, Chiradeep Roy, Tahrima Rahman, Eric D. Ragan, Nicholas Ruozzi, and Vibhav Gogate. Don't explain without verifying veracity: An evaluation of explainable AI with video activity recognition. *CoRR*, abs/2005.02335, 2020.

[53] Kareem Ahmed, Zhe Zeng, Mathias Niepert, and Guy Van den Broeck. Simple: A gradient estimator for k-subset sampling. In *ICLR*, 2023.

[54] Pasha Khosravi, YooJung Choi, Yitao Liang, Antonio Vergari, and Guy Van den Broeck. On tractable computation of expected predictions. *Advances in Neural Information Processing Systems*, 32:11169–11180, 2019.

[55] Nikil Roashan Selvam, Guy Van den Broeck, and YooJung Choi. Certifying fairness of probabilistic circuits. In *Proceedings of the 37th AAAI Conference on Artificial Intelligence*, feb 2023.

[56] YooJung Choi, Meihua Dang, and Guy Van den Broeck. Group fairness by probabilistic modeling with latent fair decisions. In *Proceedings of the 35th AAAI Conference on Artificial Intelligence*, Feb 2021.

[57] Alvaro Correia, Robert Peharz, and Cassio P de Campos. Joints in random forests. In *Advances in Neural Information Processing Systems 33 (NeurIPS)*, 2020.

[58] Wenzhe Li, Zhe Zeng, Antonio Vergari, and Guy Van den Broeck. Tractable computation of expected kernels. In *Proceedings of the 37th Conference on Uncertainty in Aritifical Intelligence (UAI)*, jul 2021.

[59] Meihua Dang, Antonio Vergari, and Guy Van den Broeck. Strudel: Learning structured-decomposable probabilistic circuits. In *Proceedings of the 10th International Conference on Probabilistic Graphical Models (PGM)*, sep 2020.

[60] Honghua Zhang, Brendan Juba, and Guy Van den Broeck. Probabilistic generating circuits. In *Proceedings of the 38th International Conference on Machine Learning (ICML)*, jul 2021.

[61] Adnan Darwiche. A logical approach to factoring belief networks. In *Proceedings of KR*, pages 409–420, 2002.

[62] Hoifung Poon and Pedro Domingos. Sum-product networks: A new deep architecture. In *2011 IEEE International Conference on Computer Vision Workshops (ICCV Workshops)*, pages 689–690. IEEE, 2011.

[63] Tahrima Rahman, Prasanna Kothalkar, and Vibhav Gogate. Cutset networks: A simple, tractable, and scalable approach for improving the accuracy of chow-liu trees. In *Joint European conference on machine learning and knowledge discovery in databases*, pages 630–645. Springer, 2014.

[64] Antonio Vergari, YooJung Choi, Anji Liu, Stefano Teso, and Guy Van den Broeck. A compositional atlas of tractable circuit operations for probabilistic inference. In *Advances in Neural Information Processing Systems 35 (NeurIPS)*, dec 2021.

[65] Doga Kisa, Guy Van den Broeck, Arthur Choi, and Adnan Darwiche. Probabilistic sentential decision diagrams. In *Fourteenth International Conference on the Principles of Knowledge Representation and Reasoning*, 2014.

[66] Daphne Koller and Nir Friedman. *Probabilistic Graphical Models: Principles and Techniques - Adaptive Computation and Machine Learning*. The MIT Press, 2009.