# Haplotype and Population Structure Inference using Neural Networks in Whole-Genome Sequencing Data

Jonas Meisner & Anders Albrechtsen
Department of Biology, Bioinformatics Centre
University of Copenhagen

September 1, 2021

**Abstract**

Accurate inference of population structure is important in many studies of population genetics. In this paper we present, `HaploNet`, a novel method for performing dimensionality reduction and clustering in genetic data. The method is based on local clustering of phased haplotypes using neural networks from whole-genome sequencing or genotype data. By utilizing a Gaussian mixture prior in a variational autoencoder framework, we are able to learn a low-dimensional latent space in which we cluster haplotypes along the genome in a highly scalable manner. We demonstrate that we can use haplotype clusters in the latent space to infer global population structure utilizing haplotype information by exploiting the generative properties of our framework. Based on fitted neural networks and its latent haplotype clusters, we can perform principal component analysis and estimate the ancestry proportions based on a maximum likelihood framework. Using sequencing data from closely related human populations, we demonstrate that our approach is better at distinguishing closely related populations than standard admixture and principal component analysis software. We further show that `HaploNet` is fast and highly scalable by applying it to genotype array data of the UK Biobank.

## 1 Introduction

Understanding population structure is a cornerstone in population and evolutionary genetics as it provides insights into demographic events and evolutionary processes that have affected a population. The most common approaches for inferring population structure from genetic data are using principal component analysis (PCA) [33] and clustering algorithms such as `STRUCTURE` [34], or derivations thereof. PCA infers continuous axes of genetic variation that summarize the genetic relationship between samples while clustering algorithms assign samples to a fixed or variable number of ancestral sources while allowing for fractional membership. The inferred axes of PCA are very useful to account for population or cryptic structure in association studies or even to simply visualize the genetic data. A limitation for most PCA and clustering algorithms is that they assume all single-nucleotide polymorphisms (SNPs) to be independent, and they do therefore not benefit from the information of correlated sites or they may be biased thereof in their global estimates [45, 33]. A notable exception is `ChromoPainter` [26], which employs the Li and Stephens hidden markov model [27] for haplotype sampling in order to model and utilize correlations between SNPs, by

letting samples be a mosaic of each other's haplotypes. This has improved the fine scale resolution and `ChromoPainter` has become state-of-the-art for inferring fine-scale population structure.

Gaussian mixture models and $k$-Means are other commonly used methods for performing unsupervised clustering. However, these methods suffer from the curse of dimensionality where distances between samples will become almost equidistant in high-dimensional space [51]. A popular approach to overcome the curse of dimensionality is to perform dimensionality reduction, e.g. using PCA, and then perform clustering in the low-dimensional space that still captures most of the variation in the full dataset [12]. Recently, deep autoencoder methods have been very successful for large-scale datasets as they perform dimensionality reduction and clustering either sequentially or jointly to benefit from induced non-linearity and scalability of deep learning architectures using neural networks [49, 48]. Deep autoencoders have also been introduced in generative models, e.g. variational autoencoders, where the unknown data generating distribution is learnt by introducing latent random variables, such that new samples can be generated from this distribution [24, 37].

Most studies in population genetics utilizing neural networks for parameter inference have mainly focused on supervised learning through simulations from demographic models [41, 39, 13, 14, 8]. Here, an overall demography is assumed, based on previous literature, and a lot of different datasets are simulated using small variations in model parameters, e.g. selection coefficient or recombination rate, with evolutionary simulators (e.g. `msprime` [20]). The studies are usually converting a simulated haplotype matrix into a downscaled fixed sized image with rows and/or columns sorted based on some distance measure. The network is then trained on the simulated datasets to learn the specified model parameters with feature boundaries in convolutional layers, and in the end, the model is tested on a real dataset. However, recently more studies have instead focused on deep generative models for data-driven inference or simulation using unsupervised learning approaches which will also be suitable for the growing number of unlabelled large-scale genetic datasets [31, 50, 4, 47].

We here present `HaploNet`, a method for inferring haplotype clusters and population structure using neural networks in an unsupervised approach for phased haplotypes of whole-genome sequencing (WGS) or genotype data. We utilize a variational autoencoder (VAE) framework to learn mappings to and from a low-dimensional latent space in which we will perform indirect clustering of haplotypes with a Gaussian mixture prior. Therefore, we do not have to rely on simulated training data from demographic models with a lot of user-specified parameters, but we are able to construct a fully data-driven inference framework. We demonstrate that we can locally cluster haplotypes and exploit the generative nature of the variational autoencoder framework to infer fine-scale population structure based on computed neural network likelihoods. State-of-the-art software `ChromoPainter` infers fine-scale population structure based on pairwise distances between haplotypes through a hidden markov model, whereas we rely on local clusterings in a low-dimensional space to non-linearly model correlated SNPs in windows in a highly scalable manner. The computed neural network likelihoods allow us to build a global clustering model similar to NGSadmix [43], which we fit using an accelerated expectation-maximization (EM) algorithm to estimate ancestry proportions as well as frequencies of our neural network inferred haplotype clusters. We show that `HaploNet` has much higher resolution than the widely used `ADMIXTURE` software [2] within the human super populations of the 1000 Genomes Project. We also use the computed likelihoods to perform PCA where we show that `HaploNet` finds similar fine-scale structure as `ChromoPainter` on the 1000 Genomes Project as well. Finally, we demonstrate the scalability of `HaploNet` by applying it to and inferring population structure in the UK Biobank utilizing haplotype information. `HaploNet` is freely available at `https://github.com/rosemeis/HaploNet`.

# 2 Material and Methods

The method is based on phased haplotype data from diallelic markers. We define the data matrix $\mathbf{X}$ as a $2N \times M$ haplotype matrix for a given chromosome, where $N$ is the number of individuals and $M$ is the number of SNPs along the chromosome. The entries of the matrix are encoded as either 0 or 1, referring to the major and minor allele, respectively.

For each chromosome, we divide the sites into windows of a fixed length of $L$ SNPs, which we assume is much smaller than $M$. The windows are non-overlapping and we will further assume that the parameters estimated in a window are independent from parameters estimated in adjacent windows. The length of the genomic windows can also be defined by a recombination map, however we have kept it fixed for the matter of generalizability and ease of application in this study. For each defined window along a chromosome, we will independently train neural networks in a variational autoencoder framework to learn haplotype clusters and encodings using a Gaussian mixture prior. From these trained networks, we are able to calculate a likelihood for an observed haplotype given an inferred haplotype cluster which resembles calculating genotype likelihoods in WGS data.

## 2.1 Variational autoencoder

An autoencoder is a state-of-the-art approach for performing dimensionality reduction by learning a mapping of the space of the input data, $\mathcal{X}$, to a low-dimensional space $\mathcal{Z}$ and a mapping back to the input data [38, 3]. More formally, we can describe the two mapping functions as $g : \mathcal{X} \mapsto \mathcal{Z}$ and $f : \mathcal{Z} \mapsto \mathcal{X}$, which are commonly called the encoder and the decoder, respectively. Both the encoder and the decoder are parameterized by (deep) neural networks to learn the mapping, as multilayer feed-forward neural networks are universal function approximators [16].

A probabilistic variant of this neural network architecture is used in a variational autoencoder (VAE) to learn the unknown generating process of the input data by introducing latent variables and modelling the joint probability through variational inference. As an optimization method, variational inference is often used to approximate the posterior distribution of a latent variable $\mathbf{z}$, $p(\mathbf{z} \,|\, \mathbf{x})$, by fitting a function that describes a chosen family of distributions, $q_\phi(\mathbf{z} \,|\, \mathbf{x})$. Thus, variational inference turns it into an optimization problem, where the objective is to maximize the evidence lower bound (ELBO) of the marginal log-likelihood of the data using an iterative procedure in contrast to Monte Carlo Markov Chain methods that approximate it by sampling from the posterior distribution [6]. The function approximating the posterior distribution is parameterized with variational parameters $\phi$. Kingma and Welling introduced the Stochastic Gradient Variational Bayes (SGVB) estimator [24] of the ELBO for approximate posterior inference in a VAE framework (as well as [37]), where a set of parameters, $(\theta, \phi)$, are optimized with amortized inference using mappings parameterized by neural networks. Here, the marginal log-likelihood, $p_\theta(\mathbf{x})$, is parameterized with parameters $\theta$. This amortization means that the number of parameters does not depend on sample size as in traditional variational inference, but it only depends on the network size [42]. The VAE can be seen as an autoencoder with its latent space being regularized by a chosen prior distribution to make the inferred latent space more interpretable and to preserve global structure. A standard Gaussian prior is the most common choice, however, it is often too simple and a lot of effort has been made to make the approximate posterior richer with normalizing flows and additional stochastic layers [36, 23, 44].

In our proposed method, `HaploNet`, we will construct a generative model and use a variational autoencoder framework to learn low-dimensional encodings of haplotypes in windows along the

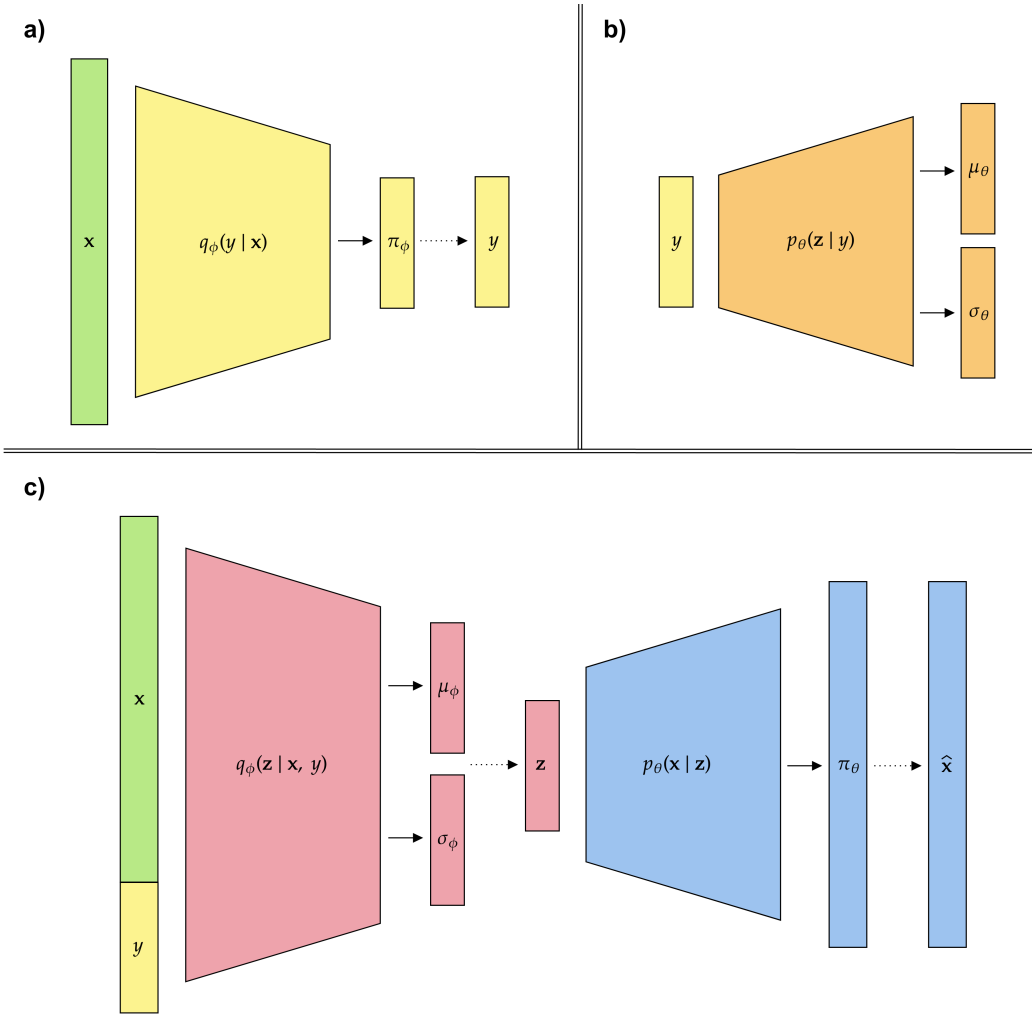


**Figure 1:** The neural network architecture of `HaploNet` split into three major substructures. Here the solid lines represent the estimation of distribution parameters, while the dashed lines represent sampling of latent variables. ***a)*** Displays the neural network parameterizing the distribution $q_\phi(y \,|\, \mathbf{x})$, for sampling the haplotype cluster, ***b)*** the network parameterizing the regularizing distribution of the sampled encoding, $p_\theta(\mathbf{z} \,|\, y)$, and ***c)*** shows the network parameterizing the distribution $q_\phi(\mathbf{z} \,|\, \mathbf{x}, y)$, for sampling the haplotype encoding, as well as the network decoding the sampled encoding to reconstruct our input. Note that the colors of the network blocks are coherent across substructures such that the sampled $y$ in ***a)*** is used in both ***b)*** and ***c)***.

genome. However, we also introduce an additional categorical variable $y$ to represent haplotype clusters with different Gaussian mixtures for approximating the posterior distribution. In this way, we are able to jointly perform dimensionality reduction and clustering of the haplotypes in a highly

scalable approach. In the following model descriptions, we will follow the mathematical notation used in the machine learning literature and Kingma and Welling (2013) [24], where $p_\theta$ and $q_\phi$ are probability functions that define the decoder and encoder part, respectively. $\theta$ and $\phi$ are the parameters (biases and weights) in the neural networks. We have provided a visualization of our overall network architecture, including descriptions of the major substructures, in Figure 1. We define the following latent variable model for a single haplotype in a genomic window with data $\mathbf{x} \in \{0,1\}^L$, Gaussian latent variable $\mathbf{z} \in \mathbb{R}^D$ and categorical latent variable $y \in \{0,1\}^C$ (one-hot encoded) such that $\mathbf{x}$ is conditionally independent of $y$:

$$p_\theta(\mathbf{x}, \mathbf{z}, y) = p_\theta(\mathbf{x} \,|\, \mathbf{z}) p_\theta(\mathbf{z} \,|\, y) p(y), \tag{1}$$

with generative processes defined as:

$$p(y) = \mathrm{Cat}(y;\, C^{-1}\mathbf{1}), \tag{2}$$

$$p_\theta(\mathbf{z} \,|\, y) = \mathcal{N}(\mathbf{z};\, \mu_\theta(y),\, \sigma^2_\theta(y)\mathbf{1}), \tag{3}$$

$$p_\theta(\mathbf{x} \,|\, \mathbf{z}) = \mathrm{Ber}(\mathbf{x};\, \pi_\theta(\mathbf{z})). \tag{4}$$

Here $\mathbf{z}$ is a $D$-dimensional vector representing the latent haplotype encoding and $C$ is the number of haplotype clusters, while $\mu_\theta : \{0,1\}^C \mapsto \mathbb{R}^D$, $\sigma^2_\theta : \{0,1\}^C \mapsto \mathbb{R}^D$ and $\pi_\theta : \mathbb{R}^D \mapsto [0,1]^L$ are mapping functions parameterized by neural networks with network parameters $\theta$. In this case, $\mathrm{Ber}(\mathbf{x};\, \pi_\theta(\mathbf{z}))$ is a vectorized notation of Bernoulli distributions and each of the $L$ sites will have an independent probability mass function. We assume that the covariance matrix of the multivariate Gaussian distribution is a diagonal matrix which will promote disentangled factors. We assume the following inference (encoder) model that constitutes the approximate posterior distribution:

$$q_\phi(\mathbf{z}, y \,|\, \mathbf{x}) = q_\phi(\mathbf{z} \,|\, \mathbf{x}, y) q_\phi(y \,|\, \mathbf{x}), \tag{5}$$

$$q_\phi(y \,|\, \mathbf{x}) = \mathrm{Cat}(y;\, \pi_\phi(\mathbf{x})), \tag{6}$$

$$q_\phi(\mathbf{z} \,|\, \mathbf{x}, y) = \mathcal{N}(\mathbf{z};\, \mu_\phi(\mathbf{x}, y),\, \sigma^2_\phi(\mathbf{x}, y)\mathbf{1}), \tag{7}$$

where $\mu_\phi : \{0,1\}^{L+C} \mapsto \mathbb{R}^D$, $\sigma^2_\phi : \{0,1\}^{L+C} \mapsto \mathbb{R}^D$ and $\pi_\phi : \{0,1\}^L \mapsto [0,1]^C$ again are mapping functions parameterized by neural networks with network parameters $\phi$. Therefore the marginal posterior distribution and marginal approximate posterior distribution of $\mathbf{z}$ will both be a mixture of Gaussians. Thus, $q_\phi(\mathbf{z}, y \,|\, \mathbf{x})$ and $p_\theta(\mathbf{x} \,|\, \mathbf{z})$ will constitute the probabilistic encoder and decoder, respectively, in comparison to the deterministic encoder and decoder of the standard autoencoder.

From the marginal log-likelihood of the data, we derive the following ELBO of our variational autoencoder model for haplotype $i$,

$$\log p_\theta(\mathbf{x}_i) \geq \mathcal{L}(\phi, \theta; \mathbf{x}_i) = \mathbb{E}_{q_\phi(\mathbf{z}, y \,|\, \mathbf{x}_i)} \left[ \log p_\theta(\mathbf{x}_i \,|\, \mathbf{z}) - \log \frac{q_\phi(\mathbf{z} \,|\, \mathbf{x}_i, y)}{p_\theta(\mathbf{z} \,|\, y)} - \log \frac{q_\phi(y \,|\, \mathbf{x}_i)}{p(y)} \right], \tag{8}$$

where the marginal log-likelihood of the full data in a window is given by:

$$\log p_\theta(\mathbf{x}_1, \ldots, \mathbf{x}_{2N}) = \sum_{i=1}^{2N} \log p_\theta(\mathbf{x}_i). \tag{9}$$

The derivation of this ELBO is described in the supplementary material as well as reparameterization tricks that are needed to approximate and optimize it through Monte Carlo samples of the latent variables. We immediately see that the first term in Equation 8 describes the reconstruction error of mapping from the latent space back to the input space as in an autoencoder framework. The next two terms act as regularization on the learnt latent spaces, where the second term encourages the variational Gaussian distribution to be close to the estimated prior distribution, while the last term encourages anti-clustering behaviour to prevent all haplotypes to cluster in one component. This is a modification of the unsupervised loss of the M2 model [22], as described by Rui Shu[1], where information of the haplotype cluster is also propagated through $p_\theta(\mathbf{z} \mid y)$. However, we further approximate the categorical latent variable with samples from a Gumbel-Softmax distribution [18, 28] instead of the categorical distribution. The Gumbel-Softmax distribution is a continuous approximation to the categorical distribution that can be easily reparameterized for differentiable sampling and gradient estimations. In this way, we can avoid an expensive computational step of having to marginalize over the categorical latent variable in the SGVB estimator of the ELBO as is done in the original model. A lot of different interpretations and implementations of the Gaussian Mixture Variational Autoencoder (GMVAE) have been proposed [11, 10, 7, 19], and a similar architecture to ours has been implemented[2].

## 2.2 Neural network likelihoods

We can exploit and utilize the generative nature of our GMVAE model based on the parameters of the trained neural networks, $\mu_\theta(y)$ and $\pi_\theta(\mathbf{z})$, to construct likelihoods from the mean latent encodings of the estimated haplotype clusters through reconstruction. We define $p(\mathbf{x}_{i,a}^{(w)} \mid y = c)$, with $y$ being one-hot encoded, as the neural network (NN) likelihood that the data is generated from the $c$-th haplotype cluster, for the $a$-th haplotype of individual $i$ in window $w$. The NN likelihoods are calculated as follows using the probability mass function of the Bernoulli distribution and the properties, $\mathbb{E}[p_\theta(\mathbf{z} \mid y)] = \mu_\theta(y)$ and $\mathbb{E}[p_\theta(\mathbf{x} \mid \mu_\theta(y))] = \pi_\theta(\mu_\theta(y))$:

$$p(\mathbf{x}_{i,a}^{(w)} \mid y = c) \propto \prod_{l=1}^{L} \pi_\theta(\mu_\theta(y = k))^{x_{i,a,l}^{(w)}} (1 - \pi_\theta(\mu_\theta(y = k)))^{1 - x_{i,a,l}^{(w)}}, \tag{10}$$

for $c = 1, \ldots, C$ (one-hot encoded) and $\mathbf{x}_{i,a}^{(w)} \in \{0, 1\}^L$ is the data of the $a$-th haplotype of individual $i$ in window $w$ with $L$ being the number of SNPs. Here $\mathbb{E}[p_\theta(\mathbf{z} \mid y)] = \mu_\theta(y)$ describes the learnt mean latent encoding of a given haplotype cluster, $y$, in the generative model and $\mathbb{E}[p_\theta(\mathbf{x} \mid \mu_\theta(y))] = \pi_\theta(\mu_\theta(y))$ is the learnt mean reconstruction of the mean latent encoding.

## 2.3 Ancestry proportions and haplotype cluster frequencies

A widely used approach for inferring population structure is estimating ancestry proportions. We propose a model for estimating ancestry proportions and haplotype cluster frequencies assuming $K$ ancestral components based on the model introduced in `NGSadmix` [43], as an extension to the `ADMIXTURE` model [2], where instead of latent states of unobserved genotypes, we have latent states of haplotype clusters. We can then construct the following likelihood model of the data $\mathbf{X}$ using

[1] http://ruishu.io/2016/12/25/gmvae/
[2] https://github.com/jariasf/GMVAE

the above defined NN likelihoods given the genome-wide ancestry proportions $\mathbf{Q}$ and window-based ancestral haplotype frequencies $\mathbf{F}$:

$$\mathcal{L}(\mathbf{Q}, \mathbf{F}; \mathbf{X}) \propto \prod_{w=1}^{W} \prod_{i=1}^{N} \prod_{a=1}^{2} \sum_{k=1}^{K} \sum_{c=1}^{C} p(\mathbf{x}_{i,a}^{(w)} \mid y = c) f_{wkc} q_{ik} \tag{11}$$

with $k$ describing the ancestral state, for $\mathbf{Q} \in [0,1]^{N \times K}$ with constraint $\sum_{k=1}^{K} q_{ik} = 1$ and $\mathbf{F} \in [0,1]^{W \times K \times C}$ with constraint $\sum_{c=1}^{C} f_{wkc} = 1$. Here $W$ is the total number of windows across chromosomes, $N$ is the number of individuals and $C$ is the number of haplotype clusters. Maximum likelihood estimates of F and Q are obtained using an expectation-maximization (EM) algorithm. The full description of the EM algorithm is detailed in the supplementary material. We use the S3 scheme of the SQUAREM methods [46] for accelerating our EM implementation, such that one large step is taken in parameter space based on the linear combination of two normal steps.

## 2.4 Inference of population structure using PCA

We can also use the NN likelihoods to infer population structure using PCA, however, the approach is not as straightforward as for the model-based ancestry estimation. We use a similar approach as in microsatellite studies where we assume that all clusters are an independent marker, and we simply sum the cluster counts for each individual in each window by taking the most probable cluster for each haplotype to construct a $N \times W \times C$ tensor, $\mathbf{Y}$.

$$\hat{y}_{i,a}^{(w)} = \arg\max_{c} \, p(\mathbf{x}_{i,a}^{(w)} \mid y = c), \tag{12}$$

$$y_{i,w,c} = \mathbb{I}(\hat{y}_{i,1}^{(w)} = c) + \mathbb{I}(\hat{y}_{i,2}^{(w)} = c) \tag{13}$$

.

We can now treat the task as a standard PCA approach in population genetics based on a binomial model [33] such that the pairwise covariance is estimated as follows for individual $i$ and $j$:

$$\mathrm{cov}(i,j) = \frac{1}{WC} \sum_{w=1}^{W} \sum_{c=1}^{C} \frac{(y_{i,w,c} - 2\hat{y}_{w,c})(y_{j,w,c} - 2\hat{y}_{w,c})}{2\hat{y}_{w,c}(1 - \hat{y}_{w,c})}, \tag{14}$$

where $\hat{y}_{w,c}$ is the frequency of the $c$-th haplotype cluster in window $w$. We filter out haplotype clusters with a frequency less than $10^{-3}$, and we can finally perform eigendecomposition on the covariance matrix to extract principal components.

## 2.5 Implementation

We have implemented `HaploNet` as a `Python` program using the `PyTorch` library (v.1.7) [32], and it is freely available on https://github.com/rosemeis/HaploNet. We have used the `NumPy` [15] and `scikit-allel` [30] libraries for preprocessing the data from Variant Call Format (VCF) into data structures to be used in `HaploNet`. The EM algorithm for estimating ancestry proportions and the algorithm for performing PCA have been implemented in `Cython` [5] for speed and parallelism.

An overall detailed description of the network architectures used in different scenarios can be found in the supplementary material. We have used fully-connected layers throughout the network, and for all inner layers in our neural networks, we are using rectified linear unit (ReLU($x$) =

$\max(0, x)$) activations to induce non-linearity into the networks, followed by batch normalization [17], while all outer layers are modelled with linear activations. The usage of linear activations means that the networks are estimating the logits of the probabilities instead the probabilities directly in $\pi_\theta(\mathbf{z})$ and $\pi_\phi(\mathbf{x})$ for computational stability, as well as for $\sigma^2_\theta(y)$ and $\sigma^2_\phi(\mathbf{x}, y)$ that represent $\log \sigma^2$ in inner computations.

We are training our networks with the Adam optimizer [21] using default parameters with a learning rate of $1.0 \times 10^{-3}$, $\beta_1 = 0.9$ and $\beta_2 = 0.999$. The batch-sizes and number of epochs are detailed in the supplementary material for each of the different datasets. We have used a fixed temperature in the sampling from the Gumbel-Softmax distribution of $\tau = 0.1$ to approximate and encourage a categorical sampling, and we use one Monte Carlo sample of the latent variables to approximate the expectation in equation 8. In practice we have an additional weight term on the categorical loss term in the ELBO, $\beta_{\text{cat}}$, such that we are able to adjust the anti-clustering effect of the uniform prior $p(y)$, by weighing the impact of this regularization term in the ELBO (see Equation S11). In all analyses, we have used $\beta_{\text{cat}} = 0.1$.

## 2.6 1000 Genomes Project

We apply `HaploNet` to the phase 3 data of the 1000 Genomes Project (TGP) [1]. The entire dataset consists of phased genotype data of 2504 unrelated individuals from 26 different populations that are assigned to 5 super populations, which are African (AFR), Admixed American (AMR), East Asian (EAS), European (EUR) and South Asian (SAS), and we inferred local haplotype structure and global population structure for each super population.

For each of the super populations, we have filtered their SNPs with a minor allele frequency threshold of 0.05. The AFR dataset consists of 661 individuals and 8.4 million SNPs from the following seven populations, African Caribbean in Barbados (ACB), Gambian in Western Division (GWD), Esan in Nigeria (ESN), Mende in Sierra Leone (MSL), Yoruba in Ibadan (YRI), Luhya in Webuye (LWK) and African Ancestry in Southwest US (ASW). The AMR dataset consists of 347 individuals and 6.2 million SNPs from the following four populations, Puerto Rican in Puerto Rico (PUR), Colombian in Medellin (CLM), Peruvian in Lima (PEL) and Mexican Ancestry in Los Angeles. The EAS dataset consists of 504 individuals and 5.6 million SNPs from the following five populations, Han Chinese South (CHS), Chinese Dai in Xishuangbanna (CDX), Kinh in Ho Chi Minh City (KHV), Han Chinese in Beijing (CHB) and Japanese in Tokyo (JPT). The EUR dataset consists of 503 individuals and 6 million SNPs from the following five populations, British in England and Scotland (GBR), Finnish in Finland (FIN), Iberian populations in Spain (IBS), Utah residents with Northern and Western European ancestry (CEU) and Toscani in Italy (TSI). The SAS dataset consists of 489 individuals and 6.2 million SNPs from the following seven populations, Punjabi in Lahore (PJL), Bengali in Bangladesh (BEB), Sri Lankan Tamil in the UK (STU), Indian Telugu in the UK (ITU) and Gujarati Indians in Houston (GIH).

We have used a fixed window size of 1024 SNPs (corresponding to a mean window size of 0.45 Mb across all super populations), a dimension of 64 for the multivariate Gaussian latent variables (haplotype encodings) and 20 latent states (number of haplotype clusters) in each of the datasets when training `HaploNet` on the five super populations of the 1000 Genomes Project.

We also apply `HaploNet` to the EUR super population, where we have filtered sites to overlap with SNPs on the available high density genotype chip data of the 1000 Genomes Project to explore our capabilities on genotype chip datasets using a lower window size of 256. Finally, we test different window sizes (512, 2024) for the EUR dataset to investigate their effect on the performance of our

method in inferring global population structure.

### 2.7 UK Biobank

To test the scalabilty of `HaploNet`, we also apply it to the array data of the UK Biobank using unrelated individuals who are self-reported as 'White British' as well as having similar genetic ancestry based on PCA from genotype data. We only use SNPs from the UK Biobank Axiom array, where we have filtered the SNPs based on available QC information, a minor allele frequency threshold of 0.01, a maximum missingness rate of 0.1 and additionally removed variants in located known high LD regions. The final dataset consists of 276732 individuals and 567499 SNPs. We perform phasing on the genotype data using `SHAPEIT4` without using a reference panel. Further details of the sample and variant filtering are described in the supplementary material.

We have used a fixed window size of 128 SNPs (corresponding to a mean window size of 0.63 Mb), a dimension of 64 for the multivariate Gaussian latent variables (haplotype encodings) and a dimension of 20 for the categorical latent variables (haplotype clusters) for the UK Biobank dataset when training `HaploNet`.

### 2.8 Computational comparisons

All models of `HaploNet`, as well as other analyses, presented in this study have been run on a machine with a NVIDIA GeForce RTX 2080 Ti GPU (11GB VRAM), using CUDA v.10.2 and cuDNN v.7.6.5, and a Intel Core i9-10920X CPU (3.5 GHz, 24 threads). We compare `HaploNet` with widely used software based on performance as well as runtime.

In each super population of the 1000 Genomes Project, we compare the estimated ancestry proportions of `HaploNet` with the results of the widely used software `ADMIXTURE` (v.1.3) [2] that uses unphased genotypes. We evaluate their performances to distinguish populations based on a signal-to-noise ratio measure using the available population labels. We use the average within-population distance of ancestry proportions in comparison to the average between-population distance as a measure of how well populations are separable, similarly to the approach in [25]. For estimating covariance matrix and performing PCA, we compare `HaploNet` to `ChromoPainter` (v.4.1.0) [26] and standard PCA in `PLINK` (v.2.0) [9] on unphased genotypes. We used `ChromoPainter` to estimate the shared genome chunks between individuals in an unsupervised manner such that no population information is given and all individuals can be used as donors of each other. We are using their linked model that utilizes pre-estimated recombination rates from genetic maps of the human chromosomes to model the correlation between SNPs using default parameters. We used their own R library for performing PCA on the estimated chunk-counts matrix. We heuristically evaluate the population structure inferred by the different methods based on their clustering accuracy of population labels in principal component space using Gaussian mixture models (GMMs) with the `mclust` R library [40]. We have only used the principal components that capture population structure. We further compare the computational runtime of `HaploNet` and `ChromoPainter` on chromosome 2 for each of the super populations.

## 3 Results

We apply `HaploNet` to the five super populations (AFR, AMR, EAS, EUR, SAS) of the 1000 Genomes Project and inferred local haplotype structure and global population structure utilizing

haplotype information to identify fine-scale structure. We compare the estimated individual ancestry proportions from our EM algorithm with results from ADMIXTURE based on signal-to-noise ratio measures (Table S1), and we compare the inferred population structure from our PCA approach with the PCA from ChromoPainter and standard PCA using PLINK based on clustering in the inferred PC space (Table S2). Additionally, we compare the runtimes of HaploNet and ChromoPainter for chromosome 2 in each of the five super populations using both GPU and CPU for training HaploNet, which is summarized in Table 1. We further apply our method to the UK Biobank genotype data to evaluate how HaploNet scales to large-scale genetic datasets as it would not be possible to run ADMIXTURE or ChromoPainter directly on it, and we compare the results to PLINK.

## 3.1 1000 Genomes Project

We first analyzed the African (AFR) super population, which includes seven populations with two populations having European admixture (ACB and ASW). After filtering, the dataset consists of 661 individuals and 8.4 million SNPs. We have displayed the inferred population structure in Figure S2 and S3. The benefit of using haplotype information is immediately clear, as the estimated ancestry proportions from HaploNet are much better at separating the populations as also seen by the signal-to-noise ratio measure in comparison to ADMIXTURE (Table S1). This is clearly visible for $K = 5$ where the MSL population is represented by its own component and its ancestry is also seen in the GWD and YRI populations, which makes sense from a geographical viewpoint. For PCA, the effect of using the haplotype methods is not apparent as the high structure makes it hard to distinguish fine-scale structure where the admixture from European sources in ACB and ASW drives most of the variation. We further validated the European admixture by running HaploNet on the 1000 Genomes Project dataset using $K = 9$, displayed in Figure S10. It only took 1.9 hours to train HaploNet on chromosome 2 in comparison to a runtime of 4 days for ChromoPainter (Table 1).

The results of the Admixed American (AMR) super population containing 347 individuals and 6.2 million SNPs are visualized in Figure S4 and S5. Interestingly, we observe that HaploNet and ADMIXTURE picks up two different signals, where HaploNet splits the PUR and CLM populations that could correspond to two different European ancestry sources while ADMIXTURE splits the PEL and MXL populations, for $K = 3, 4$. In $K = 4$, we see that HaploNet captures an African ancestry signal, mostly seen for PUR, as validated from running our method on the full dataset (Figure S10). In the PCA plots, we see that HaploNet and ChromoPainter are clustering the populations much better than standard PCA by making them more separable as illustrated by their clustering accuracies (Table S2).

We next analyzed the East Asian (EAS) super population that contains 504 individuals and 5.6 million SNPs with results shown in Figure S6 and S7. From the estimated ancestry proportions, it is clear that HaploNet outperforms ADMIXTURE, where it is able to separate the Vietnamese ancestry signal of the KHV population from the ancestry signal of the Dai people (CDX) for $K = 4$. We see a very similar pattern on the PCA plots, where the haplotype based methods are able to separate the two populations as well, while the standard PCA approach can not. The computational runtime of HaploNet was 0.9 hours in comparison to a runtime of 1.7 days for ChromoPainter (Table 1).

The results of the European (EUR) super population are visualized in Figure 2 and 3 for ancestry proportions and PCA, respectively. The dataset contains 503 individuals and 6 million SNPs and it took 1 hour to train HaploNet while the runtime of ChromoPainter was 1.9 days

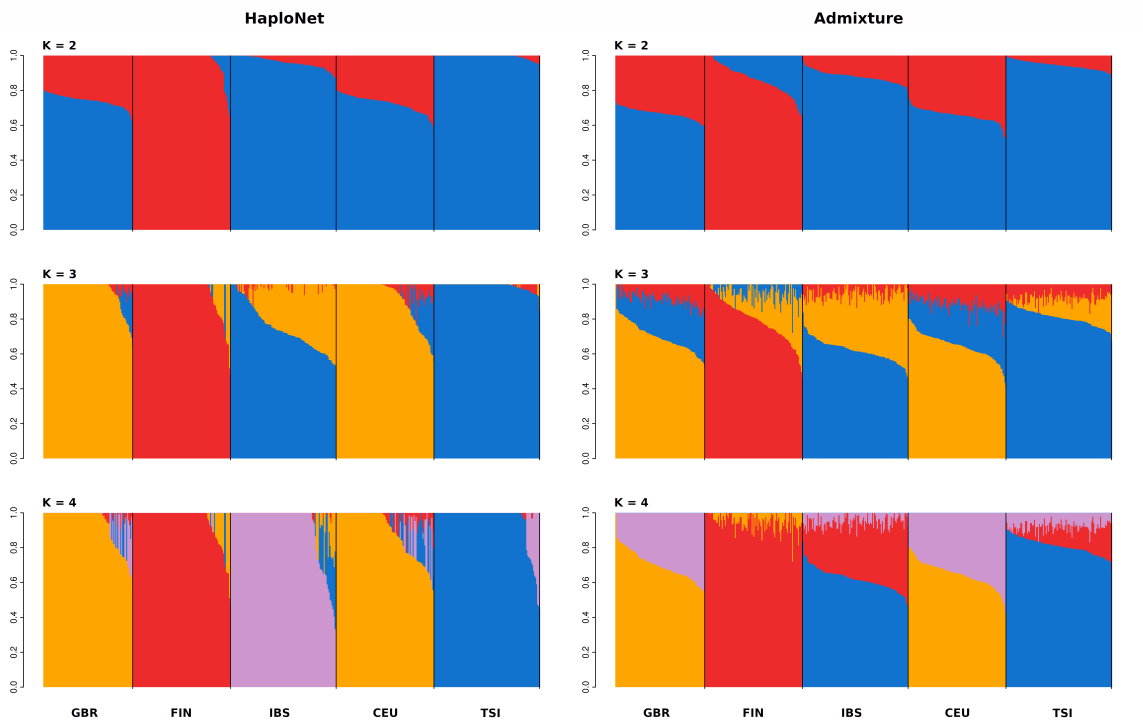


**Figure 2:** Estimated ancestry proportions in the EUR super population of the 1000 Genomes Project using `HaploNet` (left column) and `ADMIXTURE` (right column) for $K = 2, 3, 4$.

(Table 1). For $K = 4$, we observe that `HaploNet` is able to separate the two Southern European populations (IBS and TSI) which is not possible for `ADMIXTURE` at all. We also see from the signal-to-noise ratio measures that `HaploNet` is much better at separating the five populations (Table S1). A similar pattern is observed in the population structure inferred using PCA for `HaploNet` and `ChromoPainter` where they are able to separate the Southern European populations on PC3 as also verified with their clustering accuracies (Table S2). The results suggest that the fine-scale structure is only detectable with the methods utilizing haplotype information.

Finally, we have the South Asian (SAS) super population with results visualized in Figure S8 and S9. Here the training time of `HaploNet` is 1 hour in comparison to a runtime of 42 hours for `ChromoPainter` for 489 individuals and 6.2 million SNPs (Table 1). We see that the complex population structure of the SAS super population makes it hard for the haplotype based methods to separate the populations on a finer-scale than the standard approaches. However, we do observe that `HaploNet` and `ChromoPainter` have tighter clusters than standard PCA.

### 3.1.1 Robustness

To test the robustness of `HaploNet` in terms of model hyperparameters, we applied it to the European super population with varying window sizes and evaluated its performance on inferring population structure based on both ancestry proportions and PCA. The results are displayed in

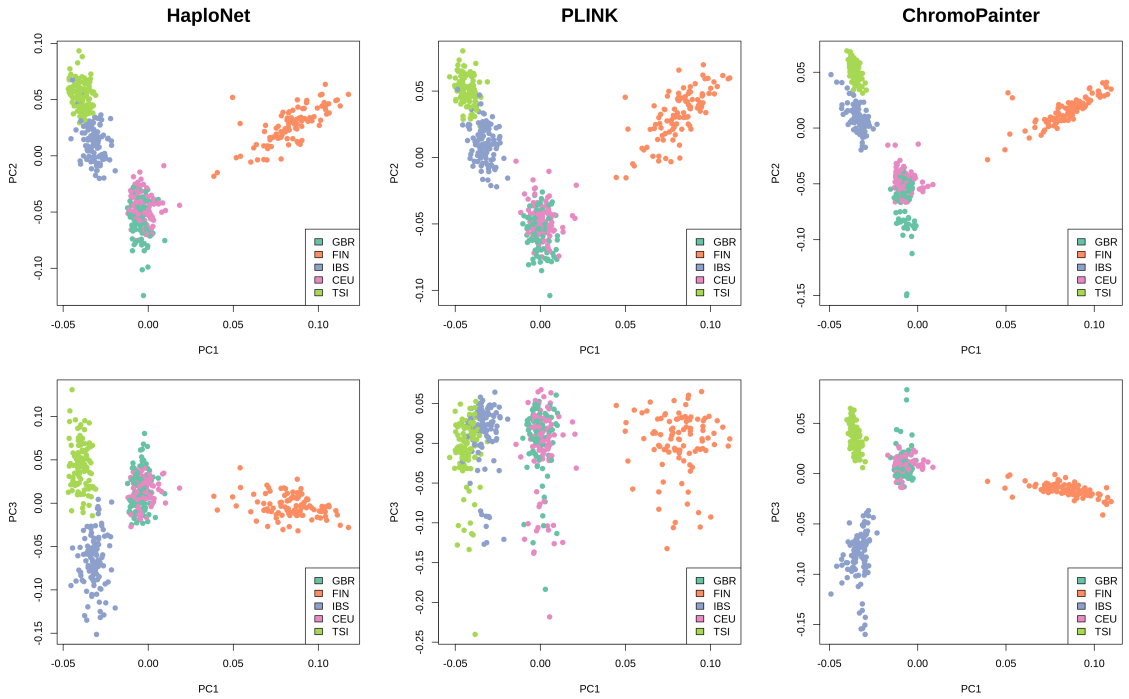


**Figure 3:** Inferred population structure in the EUR super population based on PCA using `HaploNet` (left column), `PLINK` (middle column) and `ChromoPainter` (right column). First row displays PC1 plotted against PC2, while the second row shows PC1 against PC3.

Figure S11 and S12. `HaploNet` estimates very similar ancestry proportions for each of the window sizes where we are able to separate the Southern European populations. However, we see a decreasing degree of resolution of the ancestry proportions as the window size is larger. For the PCA plots, we also observe the decreasing resolution as well where `HaploNet` has problems with separating the Southern European populations for $L = 2048$.

We filtered out SNPs not overlapping with the high density genotype chip of the Omni platform [1], as well in the European super population (EUR), to test how well `HaploNet` would perform in this scenario. After filtering, the dataset contained of 1.3 million SNPs and we have used a window size of 256 in `HaploNet`. The results are visualized in Figure S13 and S14, where `HaploNet` has similar performance in comparison to when it is used on the full dataset, however, it has a slight loss of resolution for estimating ancestry proportions but it is again still able to split the two Southern European populations which was not possible for `ADMIXTURE` with the full dataset.

## 3.2 UK Biobank

We also applied `HaploNet` to SNP array data of the UK Biobank dataset and infer population structure on a subset of 276732 unrelated with self-reported ethnicity as 'White-British' with a total of 567499 SNPs after quality control and filtering. It took 9.6 hours to train `HaploNet` on

| | HaploNet (GPU) | HaploNet (CPU) | ChromoPainter |
|---|---|---|---|
| **AFR** | 1.9 hours | 3.9 hours | 95.7 hours |
| **AMR** | 0.8 hours | 1.6 hours | 21.8 hours |
| **EAS** | 0.9 hours | 1.9 hours | 41.7 hours |
| **EUR** | 1.0 hours | 2.1 hours | 45.6 hours |
| **SAS** | 1.0 hours | 2.1 hours | 41.9 hours |

**Table 1:** Runtimes for chromosome 2 in the five super populations of the 1000 Genomes Project using `HaploNet` and `ChromoPainter`. `HaploNet` has both been trained on a GPU and a CPU on a machine with 24 threads. In all scenarios, models were trained with a fixed window size of 1024 SNPs, a batch-size of 128 for 200 epochs.

the chromosome 2 with a window size of 128, a batch-size of 4096 for 30 epochs. The results are displayed for estimating ancestry proportions and inferring population structure using PCA in Figure 4 and S15, respectively. For $K = 3$, `HaploNet` infers three clear ancestral components that reflect English, Scottish and Welsh sources, where the English component has interestingly the strongest signal in the north/center of England, the Scottish component has the strongest signal in the Outer Hebrides and the Welsh component has the strongest signal in Southern Wales. For the population structure inferred using our PCA method, we capture similar structure in the top PCs, where we additionally see a component capturing the variation between North and South Wales. On PC4, we capture structure that does not reflect genetic variation due to population structure but instead captures variation that could be strong LD structure as previously observed for the UK Biobank [35]. When performing standard PCA in `PLINK` (Figure S16), we can clearly see the bias of potentially strong LD structure or ascertainment bias being captured instead of population structure in this homogeneous subset of the UK Biobank. It is only after applying strict LD pruning of SNPs that `PLINK` captures population structure (Figure S17) from only 145732 SNPs.

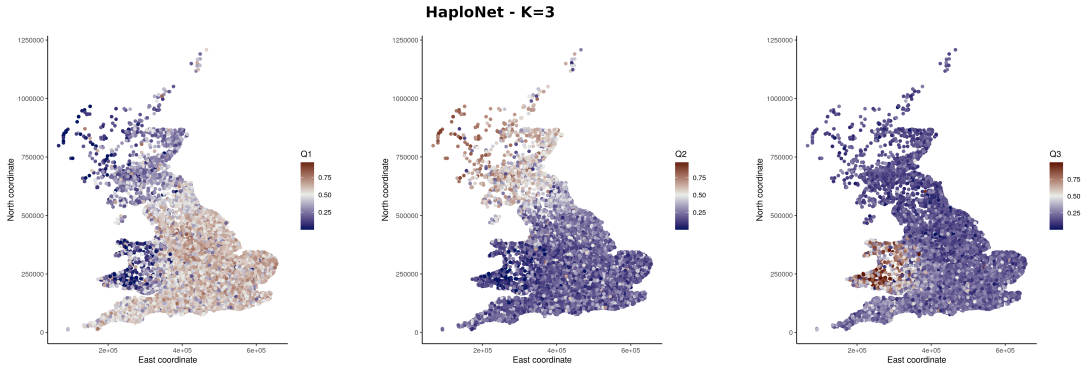


**Figure 4:** Estimated ancestry proportions of the subset of unrelated self-identified 'White-British' individuals in the UK Biobank dataset using `HaploNet` for $K = 3$. Individuals are plotted by their birthplace coordinates and colored by their ancestry proportion for each of the three sources.

# 4 Discussion

We have presented our new framework, `HaploNet`, which performs dimensionality reduction and clusters haplotypes using neural networks. We explored its capability to infer population structure based on local haplotype clusterings, which are performed in windows along the genome, as well as its generative properties for estimating neural network likelihoods. This architecture further allows us to explore and understand the processes for haplotype clustering through the learnt mappings. We show the benefits of merging machine learning with traditional statistical frameworks, as we develop a novel method for estimating ancestry proportions from neural networks in a likelihood model. We tested `HaploNet` on data from the 1000 Genomes Project and compared its results to commonly used software for inferring population structure based on ancestry proportions or PCA. We demonstrate that `HaploNet` is capable of utilizing haplotype information for inferring fine-scale population structure that outperforms `ADMIXTURE` with respect to signal-to-noise ratio, and `HaploNet` infers similar population structure to `ChromoPainter` using PCA, while being much faster. We further demonstrate the scalability of `HaploNet` by applying it to genotype data of the UK Biobank with hundreds of thousands of individuals that would not be possible with `ADMIXTURE` or `ChromoPainter`.

The number of clusters, $C$, is usually a non-trivial hyperparameter to set in a Gaussian mixture model or in a genetic clustering setting. Multiple runs of varying $C$ are usually performed and evaluated based on some criteria. In our study, we saw that `HaploNet` seemed capable of inferring the number of haplotype clusters to use by setting $C$ to a high fixed number ($C = 20$ in all analyses). It would then only use a subset of the haplotype clusters to model the haplotype encodings, which has also been observed in a different application of the GMVAE model [7]. This is especially beneficial for our approach, as it would not be feasible to verify or validate $C$ along the genome of varying window sizes, though future work into this behaviour would be of high interest for the GMVAE model.

A limitation of our model, as for all deep learning models, is that only relatively small genetic datasets are usually available to researchers, which introduces problems with training convergence and overfitting. We have therefore kept the number of parameters in our model low, while using an autoencoder architecture that naturally regularizes its reconstruction performance. Another advantage of having a small model configuration is observed with the low training times on both GPU and CPU setups that broadens the application opportunities of `HaploNet`. However, the difference between GPU and CPU will be larger when running chromosomes in parallel. For all analyses in this study, we have been using the entire data for training our neural networks to model all available haplotypes, whereas with larger labelled datasets, one would usually partition the data in three; one for training, one for tuning hyperparameters of the model and one for ensuring that the trained model generalizes to new data. We have also limited ourselves to using fixed window lengths across chromosomes for the matter of simplicity and ease of use, where we instead could have included external information from genetic maps to define windows of variable size. We show that `HaploNet` is somewhat robust to changes in window size when inferring global population structure, either using PCA or ancestry proportions, however for fixed-size windows, there is a trade-off in resolution and training time that is subject for future research. We further show that we are still able to capture much of the fine-scale structure when only evaluating variable sites available on a common genotype chip that allows for broader applications of our method.

Our model serves as a proof-of-concept and an exploration for how non-linear neural networks and specialized architectures can be utilized to learn haplotype clusters and encodings across a full

genome in a very scalable procedure. We hypothesize that as the number of large-scale genetic datasets are growing, we will see the increasing importance of deep learning in population genetics, as deeper models can be trained and more bespoke architectures will be developed. As shown in our study, we can even use learnt mappings together with standard statistical frameworks to further improve our understanding of genetic variation. Future developments of our framework is to use the haplotype clusters in sequential models, to e.g. infer local ancestry and IBD tracts in HMMs, as well as investigate its potential integration into imputation based on haplotype reconstruction and association studies.

# 5 Data availability statement

The 1000 Genomes Project phase 3 data used in this study is publicly available at `https://www.internationalgenome.org/category/phase-3/`. The genotype data from the UK Biobank can be obtained by application `https://www.ukbiobank.ac.uk/`.

# 6 Acknowledgements

The study was supported by the Lundbeck foundation (R215-2015-4174). This research was conducted using the UK Biobank Resource under application 32683.

# References

[1] 1000 Genomes Project Consortium et al. A global reference for human genetic variation. *Nature*, 526(7571):68–74, 2015.

[2] D. H. Alexander, J. Novembre, and K. Lange. Fast model-based estimation of ancestry in unrelated individuals. *Genome research*, 19(9):1655–1664, 2009.

[3] P. Baldi. Autoencoders, unsupervised learning, and deep architectures. In *Proceedings of ICML workshop on unsupervised and transfer learning*, pages 37–49, 2012.

[4] C. Battey, G. C. Coffing, and A. D. Kern. Visualizing population structure with variational autoencoders. *BioRxiv*, 2020.

[5] S. Behnel, R. Bradshaw, C. Citro, L. Dalcin, D. S. Seljebotn, and K. Smith. Cython: The best of both worlds. *Computing in Science & Engineering*, 13(2):31–39, 2011.

[6] D. M. Blei, A. Kucukelbir, and J. D. McAuliffe. Variational inference: A review for statisticians. *Journal of the American statistical Association*, 112(518):859–877, 2017.

[7] Y. Bozkurt Varolgunes, T. Bereau, and J. F. Rudzinski. Interpretable embeddings from molecular simulations using gaussian mixture variational autoencoders. *arXiv e-prints*, pages arXiv–1912, 2019.

[8] J. Chan, V. Perrone, J. Spence, P. Jenkins, S. Mathieson, and Y. Song. A likelihood-free inference framework for population genetic data using exchangeable neural networks. In *Advances in neural information processing systems*, pages 8594–8605, 2018.

[9] C. C. Chang, C. C. Chow, L. C. Tellier, S. Vattikuti, S. M. Purcell, and J. J. Lee. Second-generation plink: rising to the challenge of larger and richer datasets. *Gigascience*, 4(1):s13742–015, 2015.

[10] M. Collier and H. Urdiales. Scalable deep unsupervised clustering with concrete gmvaes. *arXiv preprint arXiv:1909.08994*, 2019.

[11] N. Dilokthanakul, P. A. Mediano, M. Garnelo, M. C. Lee, H. Salimbeni, K. Arulkumaran, and M. Shanahan. Deep unsupervised clustering with gaussian mixture variational autoencoders. *arXiv preprint arXiv:1611.02648*, 2016.

[12] C. Ding and X. He. K-means clustering via principal component analysis. In *Proceedings of the twenty-first international conference on Machine learning*, page 29, 2004.

[13] L. Flagel, Y. Brandvain, and D. R. Schrider. The unreasonable effectiveness of convolutional neural networks in population genetic inference. *Molecular biology and evolution*, 36(2):220–238, 2019.

[14] G. R. Gower, P. I. Picazo, M. Fumagalli, and F. Racimo. Detecting adaptive introgression in human evolution using convolutional neural networks. *bioRxiv*, 2020.

[15] C. R. Harris, K. J. Millman, S. J. van der Walt, R. Gommers, P. Virtanen, D. Cournapeau, E. Wieser, J. Taylor, S. Berg, N. J. Smith, et al. Array programming with numpy. *Nature*, 585(7825):357–362, 2020.

[16] K. Hornik, M. Stinchcombe, and H. White. Universal approximation of an unknown mapping and its derivatives using multilayer feedforward networks. *Neural networks*, 3(5):551–560, 1990.

[17] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015.

[18] E. Jang, S. Gu, and B. Poole. Categorical reparameterization with gumbel-softmax. *arXiv preprint arXiv:1611.01144*, 2016.

[19] Z. Jiang, Y. Zheng, H. Tan, B. Tang, and H. Zhou. Variational deep embedding: An unsupervised and generative approach to clustering. *arXiv preprint arXiv:1611.05148*, 2016.

[20] J. Kelleher and K. Lohse. Coalescent simulation with msprime. *Statistical Population Genomics*, page 191, 2020.

[21] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[22] D. P. Kingma, S. Mohamed, D. J. Rezende, and M. Welling. Semi-supervised learning with deep generative models. In *Advances in neural information processing systems*, pages 3581–3589, 2014.

[23] D. P. Kingma, T. Salimans, R. Jozefowicz, X. Chen, I. Sutskever, and M. Welling. Improved variational inference with inverse autoregressive flow. In *Advances in neural information processing systems*, pages 4743–4751, 2016.

[24] D. P. Kingma and M. Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[25] D. J. Lawson and D. Falush. Population identification using genetic data. *Annual review of genomics and human genetics*, 13:337–361, 2012.

[26] D. J. Lawson, G. Hellenthal, S. Myers, and D. Falush. Inference of population structure using dense haplotype data. *PLoS Genet*, 8(1):e1002453, 2012.

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

[28] C. J. Maddison, A. Mnih, and Y. W. Teh. The concrete distribution: A continuous relaxation of discrete random variables. *arXiv preprint arXiv:1611.00712*, 2016.

[29] C. J. Maddison, D. Tarlow, and T. Minka. A* sampling. In *Advances in Neural Information Processing Systems*, pages 3086–3094, 2014.

[30] A. Miles and N. Harding. scikit-allel: A python package for exploring and analysing genetic variation data, 2017.

[31] D. M. Montserrat, C. Bustamante, and A. Ioannidis. Class-conditional vae-gan for local-ancestry simulation. *arXiv preprint arXiv:1911.13220*, 2019.

[32] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. In *Advances in neural information processing systems*, pages 8026–8037, 2019.

[33] N. Patterson, A. L. Price, and D. Reich. Population structure and eigenanalysis. *PLoS genet*, 2(12):e190, 2006.

[34] J. K. Pritchard, M. Stephens, and P. Donnelly. Inference of population structure using multilocus genotype data. *Genetics*, 155(2):945–959, 2000.

[35] F. Privé, K. Luu, M. G. Blum, J. J. McGrath, and B. J. Vilhjálmsson. Efficient toolkit implementing best practices for principal component analysis of population genetic data. *Bioinformatics*, 36(16):4449–4457, 2020.

[36] D. J. Rezende and S. Mohamed. Variational inference with normalizing flows. *arXiv preprint arXiv:1505.05770*, 2015.

[37] D. J. Rezende, S. Mohamed, and D. Wierstra. Stochastic backpropagation and approximate inference in deep generative models. *arXiv preprint arXiv:1401.4082*, 2014.

[38] D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Learning internal representations by error propagation. Technical report, California Univ San Diego La Jolla Inst for Cognitive Science, 1985.

[39] D. R. Schrider and A. D. Kern. Supervised machine learning for population genetics: a new paradigm. *Trends in Genetics*, 34(4):301–312, 2018.

[40] L. Scrucca, M. Fop, T. B. Murphy, and A. E. Raftery. mclust 5: clustering, classification and density estimation using gaussian finite mixture models. *The R journal*, 8(1):289, 2016.

[41] S. Sheehan and Y. S. Song. Deep learning for population genetic inference. *PLoS computational biology*, 12(3):e1004845, 2016.

[42] R. Shu, H. H. Bui, S. Zhao, M. J. Kochenderfer, and S. Ermon. Amortized inference regularization. *Advances in Neural Information Processing Systems*, 31:4393–4402, 2018.

[43] L. Skotte, T. S. Korneliussen, and A. Albrechtsen. Estimating individual admixture proportions from next generation sequencing data. *Genetics*, 195(3):693–702, 2013.

[44] C. K. Sønderby, T. Raiko, L. Maaløe, S. K. Sønderby, and O. Winther. Ladder variational autoencoders. In *Advances in neural information processing systems*, pages 3738–3746, 2016.

[45] H. Tang, J. Peng, P. Wang, and N. J. Risch. Estimation of individual admixture: analytical and study design considerations. *Genetic Epidemiology: The Official Publication of the International Genetic Epidemiology Society*, 28(4):289–301, 2005.

[46] R. Varadhan and C. Roland. Simple and globally convergent methods for accelerating the convergence of any em algorithm. *Scandinavian Journal of Statistics*, 35(2):335–353, 2008.

[47] Z. Wang, J. Wang, M. Kourakos, N. Hoang, H. H. Lee, I. Mathieson, and S. Mathieson. Automatic inference of demographic parameters using generative adversarial networks. *bioRxiv*, 2020.

[48] J. Xie, R. Girshick, and A. Farhadi. Unsupervised deep embedding for clustering analysis. In *International conference on machine learning*, pages 478–487, 2016.

[49] B. Yang, X. Fu, N. D. Sidiropoulos, and M. Hong. Towards k-means-friendly spaces: Simultaneous deep learning and clustering. In *international conference on machine learning*, pages 3861–3870. PMLR, 2017.

[50] B. Yelmen, A. Decelle, L. Ongaro, D. Marnetto, F. Montinaro, C. Furtlehner, L. Pagani, and F. Jay. Creating artificial human genomes using generative models. 2019.

[51] A. Zimek, E. Schubert, and H.-P. Kriegel. A survey on unsupervised outlier detection in high-dimensional numerical data. *Statistical Analysis and Data Mining: The ASA Data Science Journal*, 5(5):363–387, 2012.

# Supplementary Material

## S1 Derivation of the evidence lower bound (ELBO)

We here derive the ELBO from the marginal log-likelihood of a haplotype, $\mathbf{x}$, using Jensen's inequality and our following definitions, $p_\theta(\mathbf{x}, \mathbf{z}, y) = p_\theta(\mathbf{x} \mid \mathbf{z}) p_\theta(\mathbf{z} \mid y) p(y)$, $q_\phi(\mathbf{z}, y \mid \mathbf{x}) = q_\phi(\mathbf{z} \mid \mathbf{x}, y) q_\phi(y \mid \mathbf{x})$. We follow the mathematical used in the machine learning literature and Kingma and Welling [24].

$$\log p_\theta(\mathbf{x}) = \log \sum_y \int_{\mathbf{z}} p_\theta(\mathbf{x}, \mathbf{z}, y) \, d\mathbf{z} \tag{S1}$$

$$= \log \sum_y \int_{\mathbf{z}} p_\theta(\mathbf{x}, \mathbf{z}, y) \frac{q_\phi(\mathbf{z}, y \mid \mathbf{x})}{q_\phi(\mathbf{z}, y \mid \mathbf{x})} \, d\mathbf{z} \tag{S2}$$

$$= \log \mathbb{E}_{q_\phi(\mathbf{z}, y \mid \mathbf{x})} \left[ \frac{p_\theta(\mathbf{x}, \mathbf{z}, y)}{q_\phi(\mathbf{z}, y \mid \mathbf{x})} \right] \tag{S3}$$

$$\geq \mathbb{E}_{q_\phi(\mathbf{z}, y \mid \mathbf{x})} \left[ \log \frac{p_\theta(\mathbf{x}, \mathbf{z}, y)}{q_\phi(\mathbf{z}, y \mid \mathbf{x})} \right] \tag{S4}$$

$$= \mathbb{E}_{q_\phi(\mathbf{z}, y \mid \mathbf{x})} \left[ \log \frac{p_\theta(\mathbf{x} \mid \mathbf{z}) p_\theta(\mathbf{z} \mid y) p(y)}{q_\phi(\mathbf{z} \mid \mathbf{x}, y) q_\phi(y \mid \mathbf{x})} \right] \tag{S5}$$

$$= \mathbb{E}_{q_\phi(\mathbf{z}, y \mid \mathbf{x})} \left[ \log p_\theta(\mathbf{x} \mid \mathbf{z}) - \log \frac{q_\phi(\mathbf{z} \mid \mathbf{x}, y)}{p_\theta(\mathbf{z} \mid y)} - \log \frac{q_\phi(y \mid \mathbf{x})}{p(y)} \right]. \tag{S6}$$

We can rewrite the last term in the expectation using $p(y = c) = C^{-1}$, for $c = 1, \ldots, C$:

$$\mathbb{E}_{q_\phi(\mathbf{z}, y \mid \mathbf{x})} \left[ \log \frac{q_\phi(y \mid \mathbf{x})}{p(y)} \right] = \mathbb{E}_{q_\phi(y \mid \mathbf{x})} \mathbb{E}_{q_\phi(\mathbf{z} \mid \mathbf{x}, y)} \left[ \log \frac{q_\phi(y \mid \mathbf{x})}{p(y)} \right] \tag{S7}$$

$$= \mathbb{E}_{q_\phi(y \mid \mathbf{x})} \left[ \log \frac{q_\phi(y \mid \mathbf{x})}{p(y)} \right] \tag{S8}$$

$$= \mathbb{E}_{q_\phi(y \mid \mathbf{x})} \left[ \log q_\phi(y \mid \mathbf{x}) \right] - \mathbb{E}_{q_\phi(y \mid \mathbf{x})} \left[ \log p(y) \right] \tag{S9}$$

$$= -H(q_\phi(y \mid \mathbf{x})) + \log C, \tag{S10}$$

where $H(X)$ is the entropy of a given distribution. In that way, both the second and third term will act as regularization on the reconstruction term through the latent space distributed by $q_\phi(\mathbf{z}, y \mid \mathbf{x})$.

**ELBO with $\beta_{\text{cat}}$ weight term**

$$\log p_\theta(\mathbf{x}) \geq \mathbb{E}_{q_\phi(\mathbf{z}, y \mid \mathbf{x})} \left[ \log p_\theta(\mathbf{x} \mid \mathbf{z}) - \log \frac{q_\phi(\mathbf{z} \mid \mathbf{x}, y)}{p_\theta(\mathbf{z} \mid y)} \right] - \beta_{\text{cat}} \left( -H(q_\phi(y \mid \mathbf{x})) + \log C \right). \tag{S11}$$

## S2 Derivation of EM algorithm

Here we derive the EM algorithm for estimating ancestry proportions $\mathbf{Q} \in [0,1]^{N \times K}$ and window-based haplotype frequencies $\mathbf{F} \in [0,1]^{W \times K \times C}$, and we will use the following notation:

$W$ number of windows

$N$ number of individuals

$C$ number of haplotype states

$K$ number of ancestral states

$\mathbf{x}_{i,a}^{(w)}$ observed data of the $a$-th haplotype of individual $i$ in window $w$

$\psi^{(n)}$ auxiliary variable in iteration $n$

$q_{ik}^{(n)}$ ancestry proportion of state $k$ for individual $i$ in iteration $n$

$f_{wkc}^{(n)}$ ancestral frequency of haplotype state $c$ in ancestral state $k$ in window $w$ in iteration $n$

We define the likelihood function:

$$\mathcal{L}(\mathbf{Q}, \mathbf{F}; \mathbf{X}) \propto \prod_{w=1}^{W} \prod_{i=1}^{N} \prod_{a=1}^{2} p(\mathbf{x}_{i,a}^{(w)} \mid \psi) \tag{S12}$$

$$p(\mathbf{x}_{i,a}^{(w)} \mid \psi) = \sum_{k=1}^{K} \sum_{c=1}^{C} p(\mathbf{x}_{i,a}^{(w)} \mid \psi, k, y = c) p(y = c \mid k, \psi) p(k \mid \psi) \tag{S13}$$

$$= \sum_{k=1}^{K} \sum_{c=1}^{C} p(\mathbf{x}_{i,a}^{(w)} \mid y = c) f_{wkc} q_{ik}. \tag{S14}$$

The EM update for $q_{ik}$ is defined as follows for iteration $n+1$:

$$q_{ik}^{(n+1)} = \frac{\sum_{w=1}^{W} \sum_{a=1}^{2} p(k \mid \mathbf{x}_{i,a}^{(w)}, \psi^{(n)})}{\sum_{k=1}^{K} \sum_{w=1}^{W} \sum_{a=1}^{2} p(k \mid \mathbf{x}_{i,a}^{(w)}, \psi^{(n)})} \tag{S15}$$

$$= \frac{\sum_{w=1}^{W} \sum_{a=1}^{2} p(k \mid \mathbf{x}_{i,a}^{(w)}, \psi^{(n)})}{2W}, \tag{S16}$$

and the update for $f_{wkc}$:

$$f_{wkc}^{(n+1)} = \frac{\sum_{i=1}^{N} \sum_{a=1}^{2} p(k, y = c \mid \mathbf{x}_{i,a}^{(w)}, \psi^{(n)})}{\sum_{c'=1}^{C} \sum_{i=1}^{N} \sum_{a=1}^{2} p(k, y = c' \mid \mathbf{x}_{i,a}^{(w)}, \psi^{(n)})}, \tag{S17}$$

where

$$p(k \,|\, \mathbf{x}_{i,a}^{(w)}, \psi^{(n)}) = \sum_{c=1}^{C} p(k, y = c \,|\, \mathbf{x}_{i,a}^{(w)}, \psi^{(n)}), \tag{S18}$$

and

$$p(k, y = c \,|\, \mathbf{x}_{i,a}^{(w)}, \psi^{(n)}) = \frac{p(\mathbf{x}_{i,a}^{(w)} \,|\, k, y = c, \psi^{(n)}) p(y = c, \mathbf{x}_{i,a}^{(w)} \,|\, \psi^{(n)})}{\sum_{k'=1}^{K} \sum_{c'=1}^{C} p(\mathbf{x}_{i,a}^{(w)} \,|\, k', y = c', \psi^{(n)}) p(y = c', \mathbf{x}_{i,a}^{(w)} \,|\, \psi^{(n)})} \tag{S19}$$

$$= \frac{p(\mathbf{x}_{i,a}^{(w)} \,|\, y = c) p(y = c \,|\, k, \psi^{(n)}) p(k \,|\, \psi^{(n)})}{\sum_{k'=1}^{K} \sum_{c'=1}^{C} p(\mathbf{x}_{i,a}^{(w)} \,|\, y = c') p(y = c' \,|\, k', \psi^{(n)}) p(k' \,|\, \psi^{(n)})} \tag{S20}$$

$$= \frac{p(\mathbf{x}_{i,a}^{(w)} \,|\, y = c) f_{wkc}^{(n)} q_{ik}^{(n)}}{\sum_{k'=1}^{K} \sum_{c'=1}^{C} p(\mathbf{x}_{i,a}^{(w)} \,|\, y = c') f_{wk'c'}^{(n)} q_{ik'}^{(n)}}. \tag{S21}$$

## S3 Reparameterization tricks

End-to-end learning is a desired trait in deep learning models as it enables joint training of model parameters through backpropagation. However with the network architectures of variational autoencoders, the sampling steps of the latent variables are not letting information flow end-to-end as they are non-differentiable. The reparameterization trick is a simple approach to express a random variable as deterministic using the introduction of an auxiliary variable as defined in the SGVB estimator [24]. In this way, the stochastic process of sampling is removed from the computation graph and we can therefore efficiently estimate gradients and perform end-to-end learning of our network parameters, $(\phi, \theta)$, through Monte Carlo samples of the latent variables to approximate the ELBO in Equation 8.

A $D$-dimensional sample from a multivariate Gaussian distribution, $\mathbf{z}$, can be expressed using the introduction of auxiliary noise variable $\boldsymbol{\epsilon}$, with $\epsilon_d \sim \mathcal{N}(0, 1)$, for $d = 1, \ldots, D$:

$$z_d = \mu_d + \sigma_d \epsilon_d. \tag{S22}$$

Similarly, the reparameterization trick for the Gumbel-Softmax distribution is an approach for providing differentiable samples that approximate samples from a categorical distribution. It is an extension of the Gumbel-Max trick [29] for sampling from a categorical distribution, where the softmax function is used as a continuous differentiable approximation to the arg max function. The Gumbel-Softmax distribution was derived simultaneously in two independent studies [18, 28]. A $C$-dimensional one-hot encoded categorical sample $y$ can be approximated as sample $\tilde{y}$ from the Gumbel-Softmax distribution, and it is defined as follows for its entry $c$:

$$\tilde{y}_c = \frac{\exp((\log(\pi_c) + g_c)/\tau)}{\sum_{c'=1}^{C} \exp((\log(\pi_{c'}) + g_{c'})/\tau)}. \tag{S23}$$

Here $g_c, \ldots, g_C$ are i.i.d. samples from Gumbel$(0, 1)$ (auxiliary variables) and $\pi_c$ is the probability of haplotype cluster $c$. The temperature parameter, $\tau$, is governing the smoothness of the distribution such that when $\tau$ approaches 0, samples become identical to samples from the categorical distribution. We have provided examples of the Gumbel-Softmax distribution in the next section to provide the reader with a better intuition of it. Note that $\tilde{y} \in [0, 1]^C$ unlike $y$, and the functions $\mu_\theta, \sigma_\theta, \mu_\phi, \sigma_\phi$ parameterized by the neural networks are therefore redefined as mapping to or from a continuous space in the practical implementation.

# S4 Gumbel-Softmax sampling

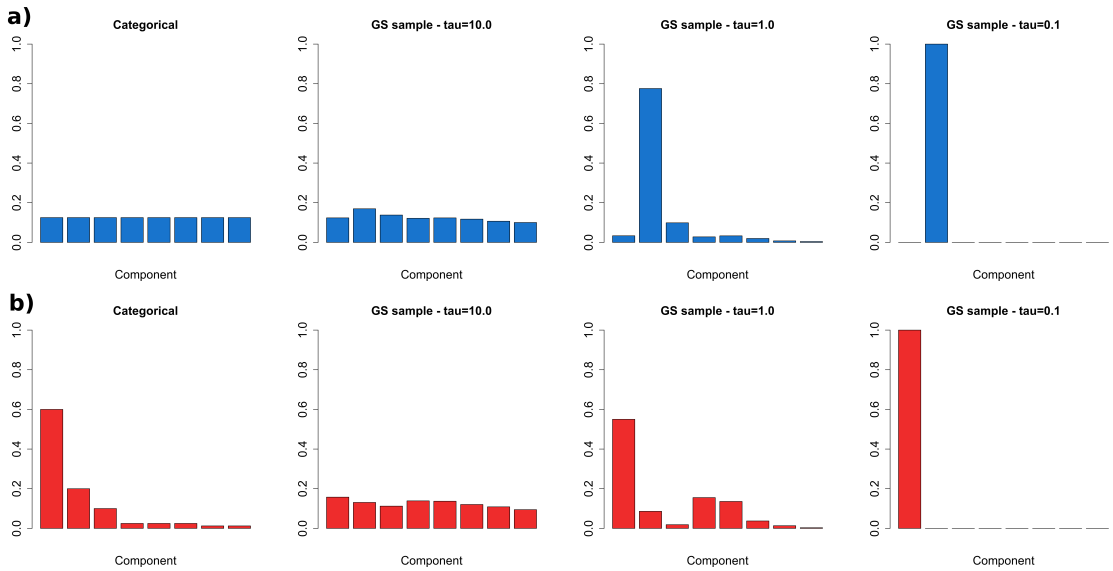


**Figure S1:** Two examples of approximate sampling from different categorical distributions using the Gumbel-Softmax distribution. First column displays the two categorical distributions with the two examples distinguishable by color, and the next three columns display random samplings from the distributions for different temperatures. The random sampling is performed using the reparameterization trick as described in Equation S23 with $\tau = 10, 1, 0.1$. ***a)*** A discrete uniform distribution, ***b)*** a categorical distribution with decreasing probabilities. Here it can be seen that as $\tau$ approaches 0, the sample from the Gumbel-Softmax distribution will converge to a categorical sample, while as $\tau$ approaches $\infty$, it will converge towards a uniform sample.

# S5 Network architectures

In all models, the architectures for the networks of `HaploNet` were defined as follows:

- $\pi_\phi(\mathbf{x})$: $L$-$H$-ReLU-BN-$C$
- $\mu_\phi(\mathbf{x}, y)$, $\sigma^2_\phi(\mathbf{x}, y)$: $(L + C)$-$H$-ReLU-BN-$D$
- $\mu_\theta(y)$, $\sigma^2_\theta(y)$: $C$-$D$
- $\pi_\theta(\mathbf{z})$: $D$-$H$-ReLU-BN-$L$

with a hidden layer of 256 units ($H = 256$), ReLU being a ReLU activation layer, and BN a batch normalization layer.

## S5.1 1000 Genomes Project (AFR, AMR, EAS, EUR, SAS)

- Dimensions: $L = 1024$ [3], $D = 64$, $C = 20$
- Batch size: 128
- Epochs: 200

## S5.2 UK Biobank

- Dimensions: $L = 128$, $D = 64$, $C = 20$
- Batch size: 4096
- Epochs: 30

# S6 UK Biobank filtering

## S6.1 Samples

- Self-reported 'White-British' ethnicity (Data-field 21000)
- Similar genetic ancestry on basis of PCA (Data-field 22006)
- No kinship to other participants (Data-field 22021)

All plots of UK Biobank analyses only shows 265616 individuals which have birth place coordinates available (Data-field 129 and 130).

## S6.2 Sites

The sites filtering is mainly based on the SNP quality control process shared by the UK Biobank resource (file[4]).

- Sites on UK Biobank Axiom Array
- Sites passing all batch QC filters
- Removed sites in long range high LD regions (list)
- Sites with minor allele frequency $> 0.01$
- Sites with missingness rate $< 0.1$

[3] $L = 512$ and $L = 2048$ were used for evaluating the effect of window size for the EUR super population, while $L = 256$ was used for the EUR super population when only using sites on the high density genotype chip.

[4] The list of SNPs after strict LD pruning used for an analysis with PLINK is in field 'in_PCA'.

# S7 Supplementary figures and tables

## S7.1 Comparisons

| | K=2 | K=3 | K=4 | K=5 |
|---|---|---|---|---|
| | | **AFR** | | |
| HaploNet | 5.57303 | 6.88566 | 7.13076 | 7.28126 |
| ADMIXTURE | 4.05225 | 5.45065 | 6.13837 | 4.87878 |
| | | **AMR** | | |
| HaploNet | 2.96490 | 2.29751 | 2.58676 | - |
| ADMIXTURE | 2.80215 | 2.13846 | 2.28060 | - |
| | | **EAS** | | |
| HaploNet | 8.58650 | 8.05124 | 7.94284 | - |
| ADMIXTURE | 7.41655 | 5.98762 | 3.21533 | - |
| | | **EUR** | | |
| HaploNet | 11.91787 | 8.36979 | 7.49894 | - |
| ADMIXTURE | 7.74677 | 5.04841 | 1.86814 | - |
| | | **SAS** | | |
| HaploNet | 2.74721 | 2.90364 | - | - |
| ADMIXTURE | 2.23998 | 2.06078 | - | - |

**Table S1:** Mean between-population distance relative to the mean within-population distance of estimated ancestry proportions. A higher value means a higher signal-to-noise ratio, thus more separable populations, while the measures between the different values of $K$ are not comparable. Both software were run multiple times to ensure that convergence was achieved.

| | HaploNet | PLINK | ChromoPainter |
|---|---|---|---|
| **AFR (3)** | 0.77307 | 0.78366 | 0.82602 |
| **AMR (3)** | 0.67435 | 0.54179 | 0.71758 |
| **EAS (3)** | 0.88492 | 0.76786 | 0.89286 |
| **EUR (3)** | 0.84493 | 0.74751 | 0.85487 |
| **SAS (2)** | 0.46830 | 0.46217 | 0.47648 |

**Table S2:** Clustering accuracy in inferred PC-space based on population labels using GMMs in `mclust` using `HaploNet`, standard PCA in `PLINK` and `ChromoPainter` for the super populations of the 1000 Genomes Project. The number of PCs used for the clustering for each method is specified in parentheses, where only PCs capturing global structure are used.

## S7.2 1000 Genomes Project

### S7.2.1 AFR

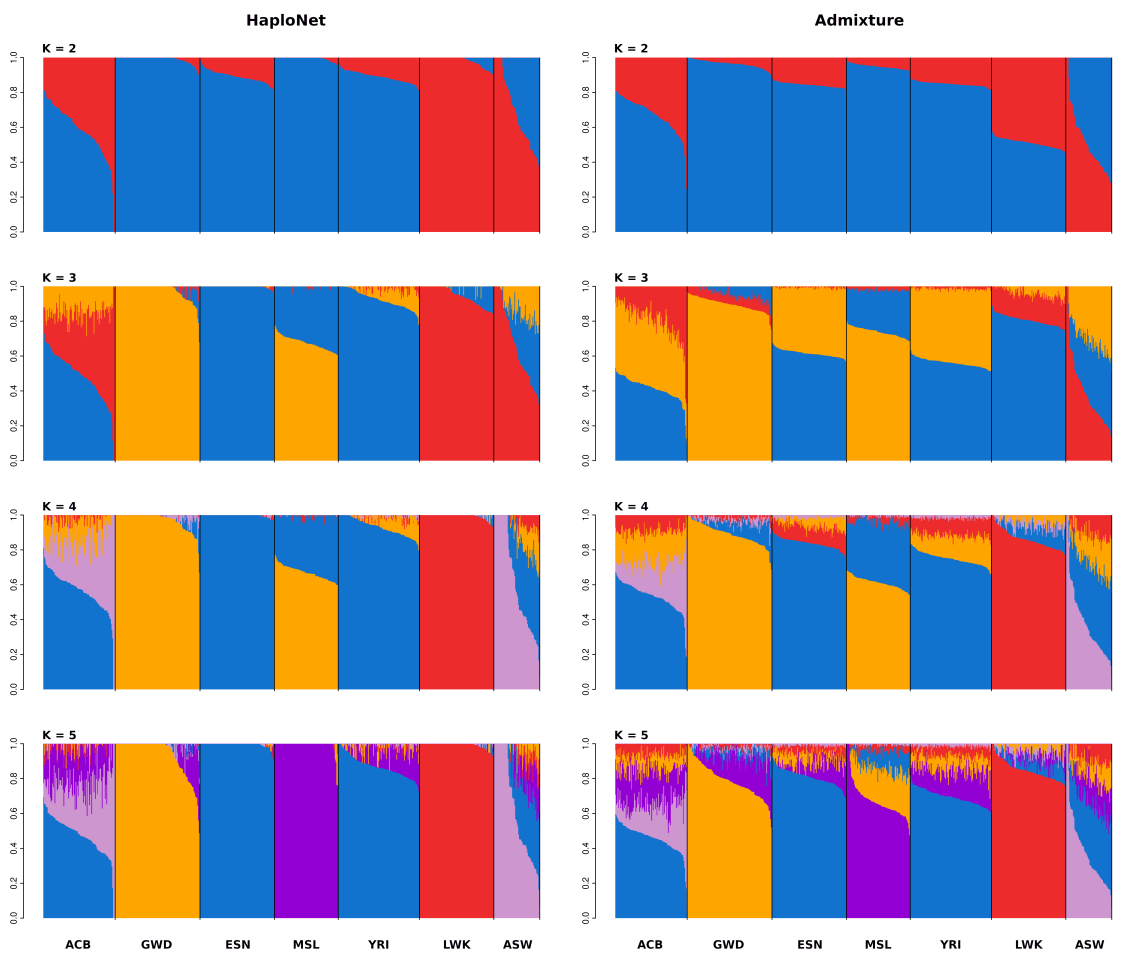


**Figure S2:** Estimated ancestry proportions in the AFR super population of the 1000 Genomes Project using `HaploNet` (left column) and `ADMIXTURE` (right column) for $K = 2, 3, 4, 5$.

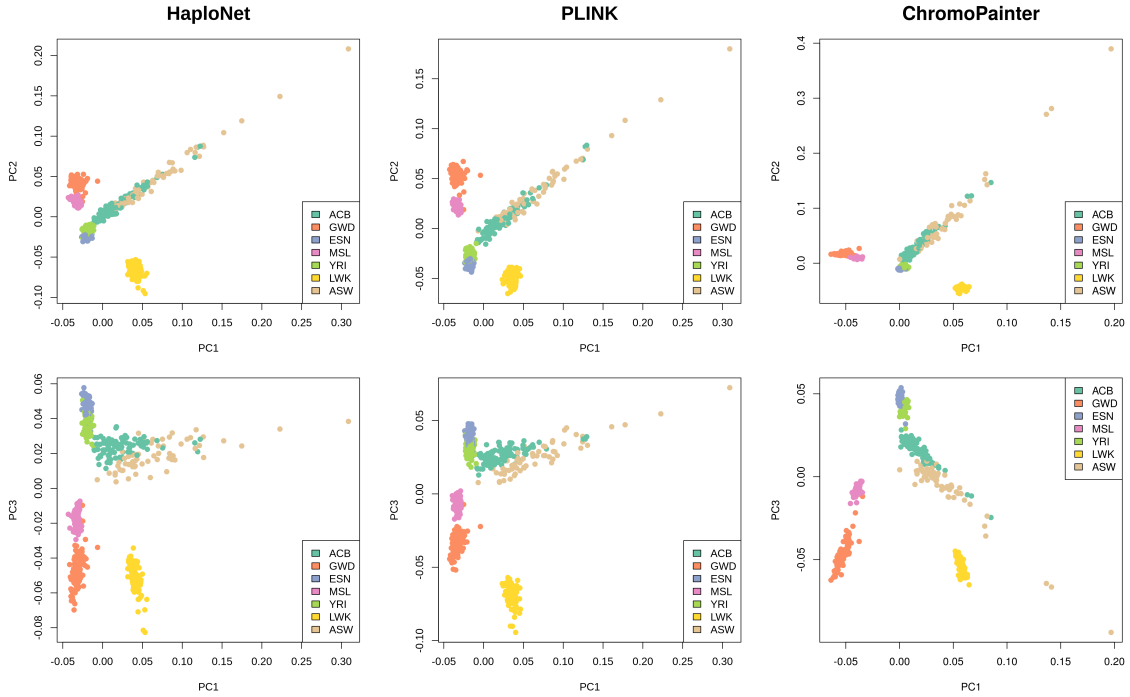


**Figure S3:** Inferred population structure in the AFR super population based on PCA using `HaploNet` (left column), `PLINK` (middle column) and `ChromoPainter` (right column). First row displays PC1 plotted against PC2, while the second row shows PC1 against PC3.

### S7.2.2 AMR

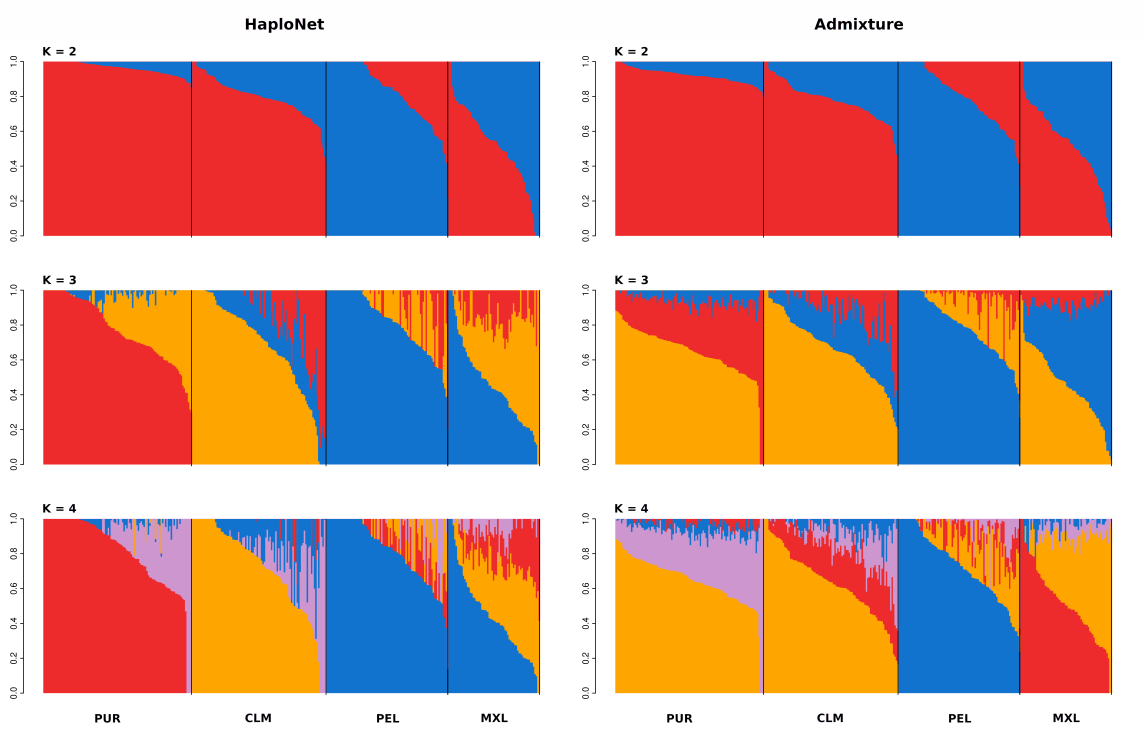


**Figure S4:** Estimated ancestry proportions in the AMR super population of the 1000 Genomes Project using `HaploNet` (left column) and `ADMIXTURE` (right column) for $K = 2, 3, 4$.

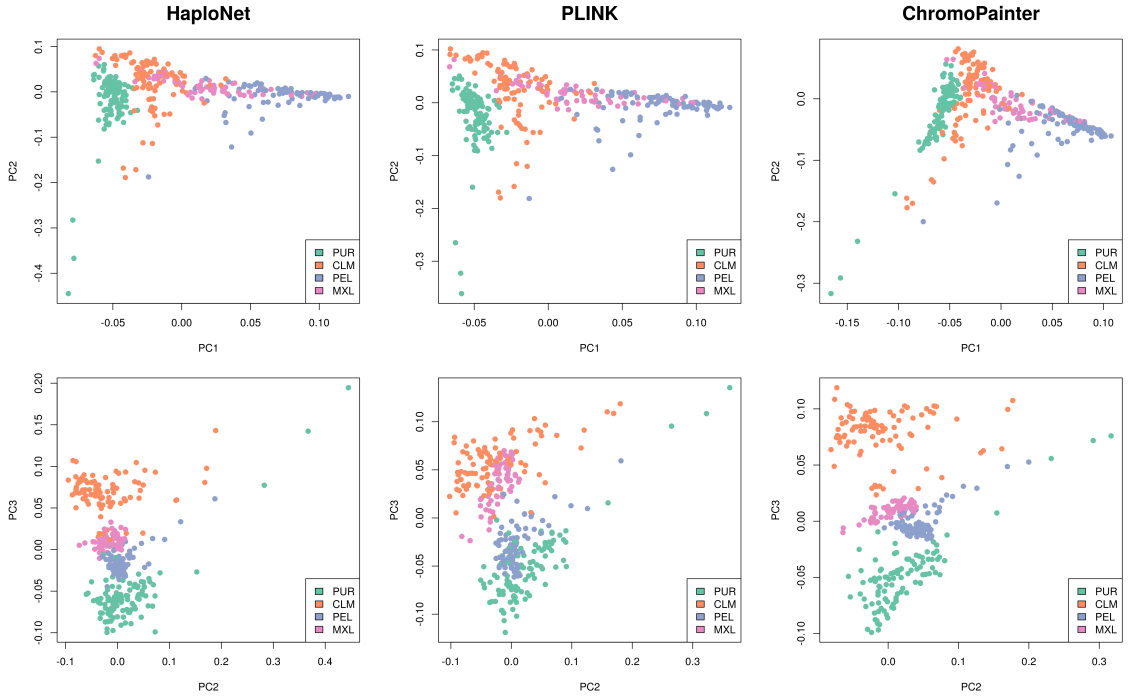


**Figure S5:** Inferred population structure in the AMR super population based on PCA using `HaploNet` (left column), `PLINK` (middle column) and `ChromoPainter` (right column). First row displays PC1 plotted against PC2, while the second row shows PC2 against PC3.

### S7.2.3 EAS

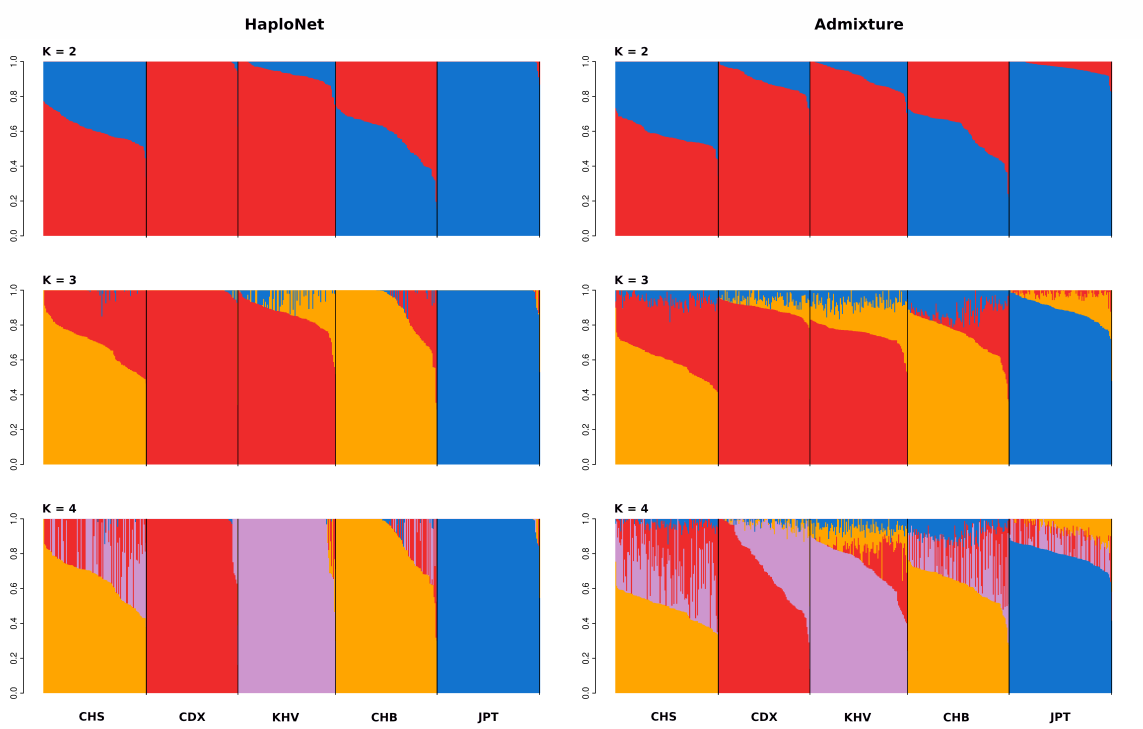


**Figure S6:** Estimated ancestry proportions in the EAS super population of the 1000 Genomes Project using `HaploNet` (left column) and `ADMIXTURE` (right column) for $K = 2, 3, 4$.

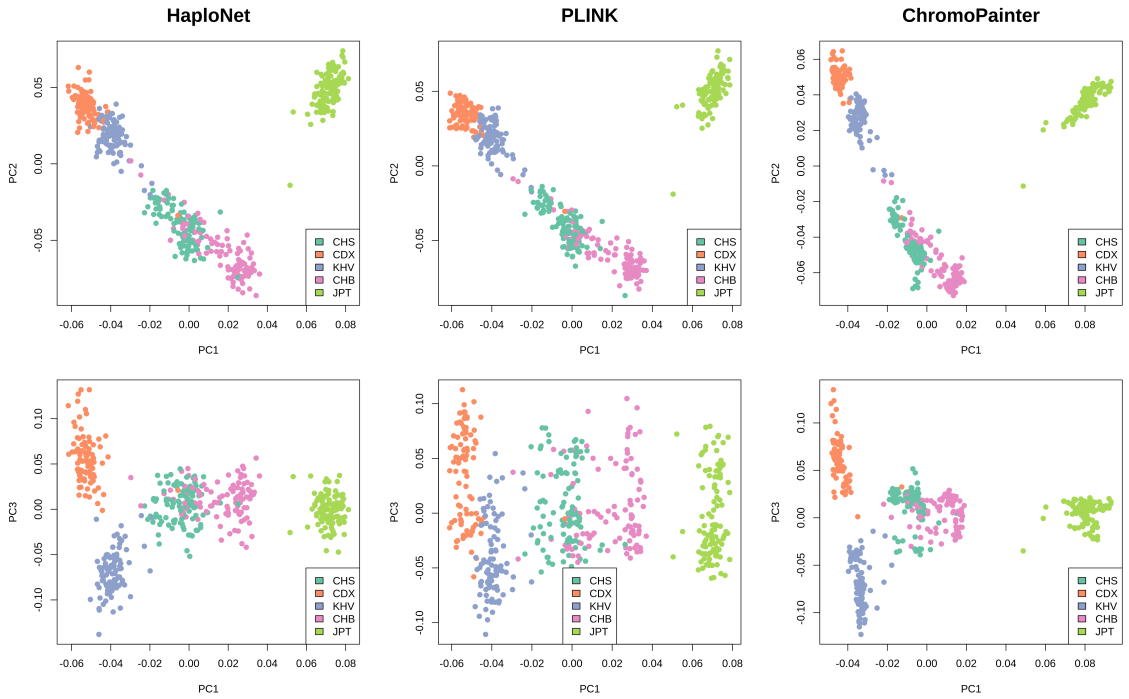


**Figure S7:** Inferred population structure in the EAS super population based on PCA using `HaploNet` (left column), `PLINK` (middle column) and `ChromoPainter` (right column). First row displays PC1 plotted against PC2, while the second row shows PC1 against PC3.

### S7.2.4 SAS

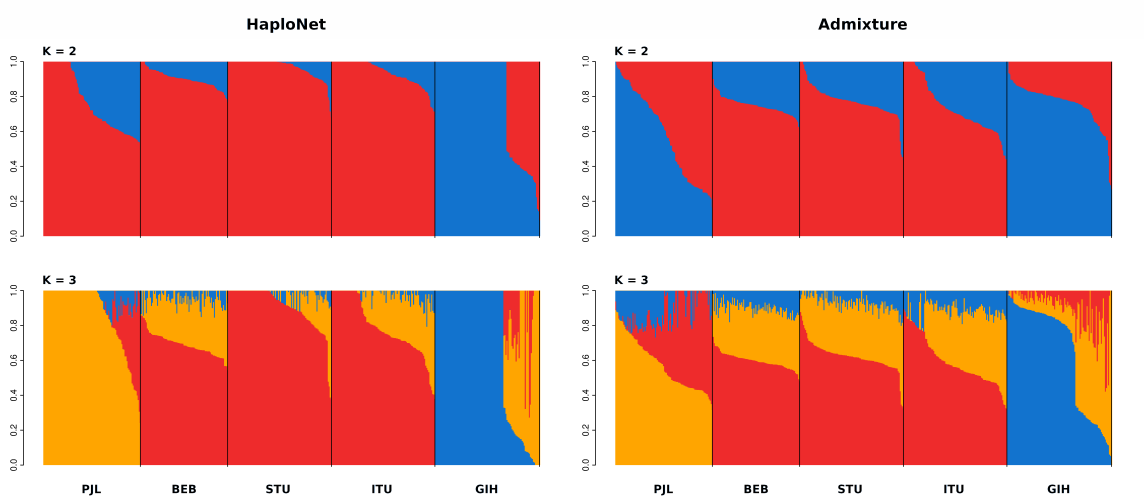


**Figure S8:** Estimated ancestry proportions in the SAS super population of the 1000 Genomes Project using `HaploNet` (left column) and `ADMIXTURE` (right column) for $K = 2, 3$.

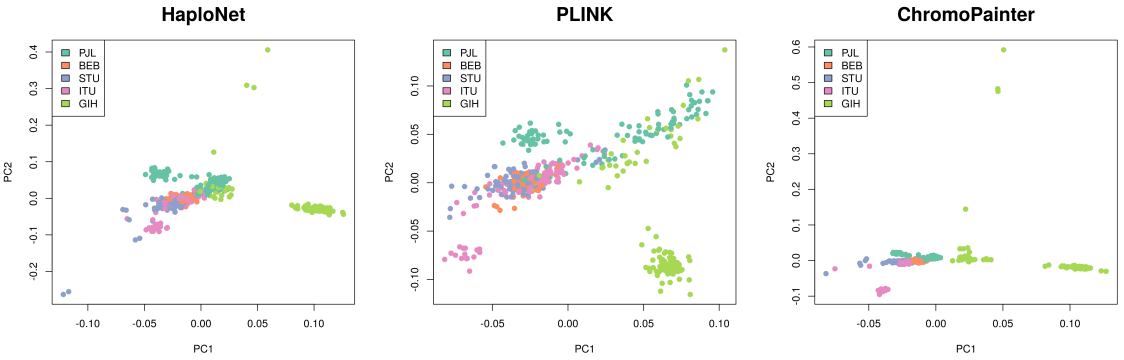


**Figure S9:** Inferred population structure in the SAS super population based on PCA using `HaploNet` (left column), `PLINK` (middle column) and `ChromoPainter` (right column). The plots display PC1 plotted against PC2.

### S7.2.5 Full dataset

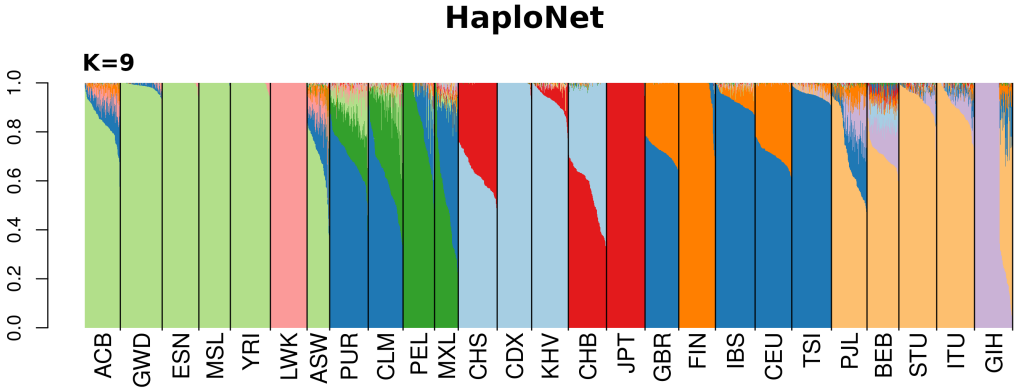


**Figure S10:** Estimated ancestry proportions in the full dataset of the 1000 Genomes Project using `HaploNet` for $K = 9$ with all hyperparameters set as for each super population, $L = 1024$, $D = 64$, $C = 20$. The full dataset consists of 2504 individuals and 6.9 million SNPs after MAF filtering.

## S7.3 Window size in `HaploNet`

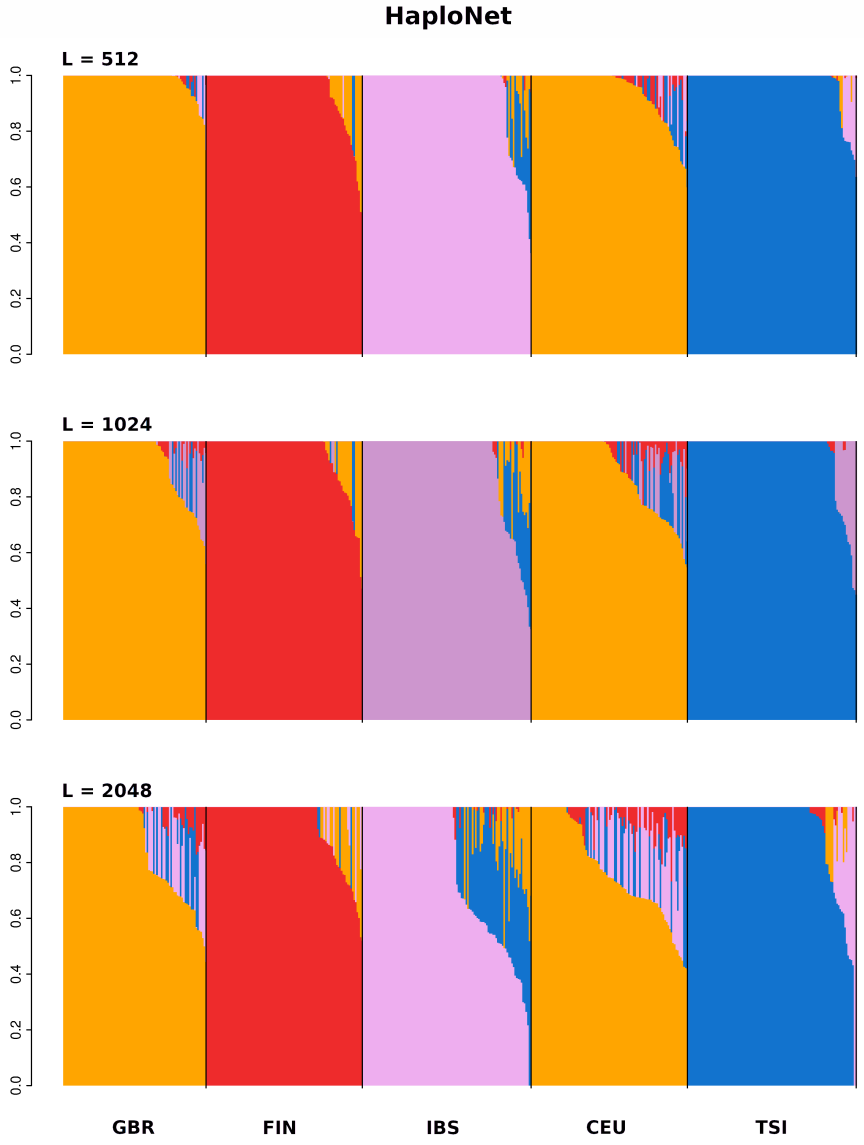


**Figure S11:** Estimated ancestry proportions in the EUR super population of the 1000 Genomes Project using `HaploNet` for $K = 4$ with three different window sizes ($L = 512, 1024, 2048$).

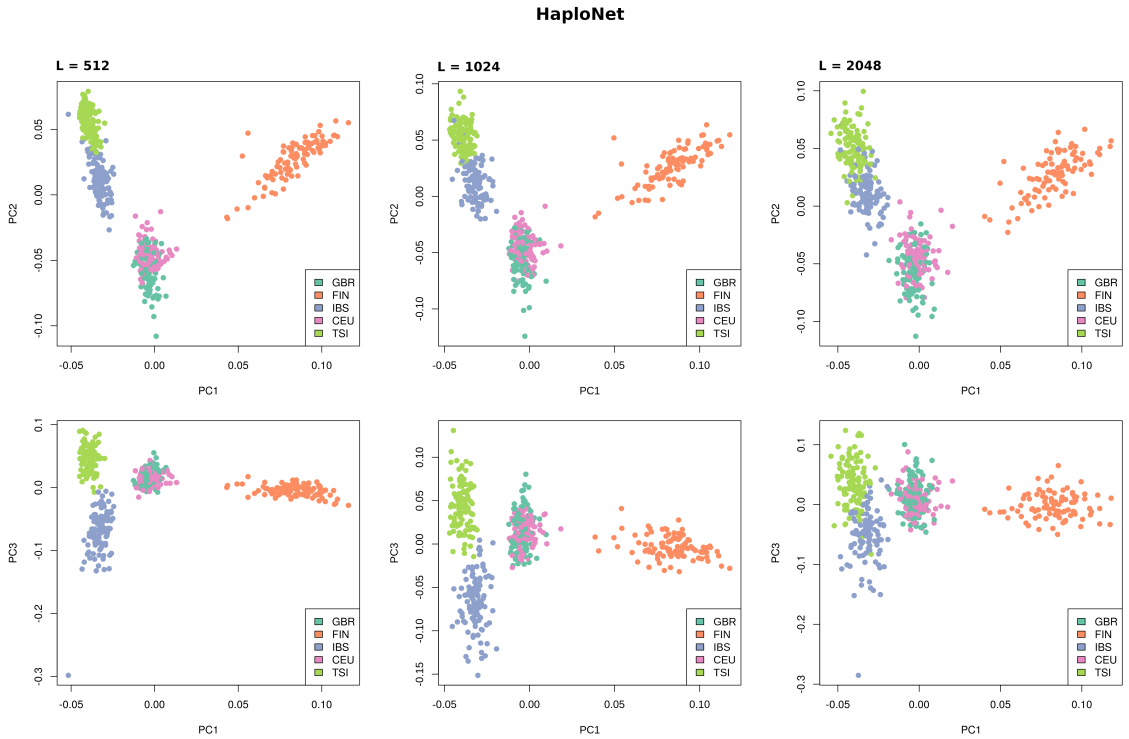


**Figure S12:** Inferred population structure in the EUR super population based on PCA using `HaploNet` with three different window sizes, $L = 512$ (first column), $L = 1024$ (second column) and $L = 2048$ (third column). First row displays PC1 plotted against PC2, while the second row shows PC1 against PC3.

## S7.4 EUR - Omni-chip sites

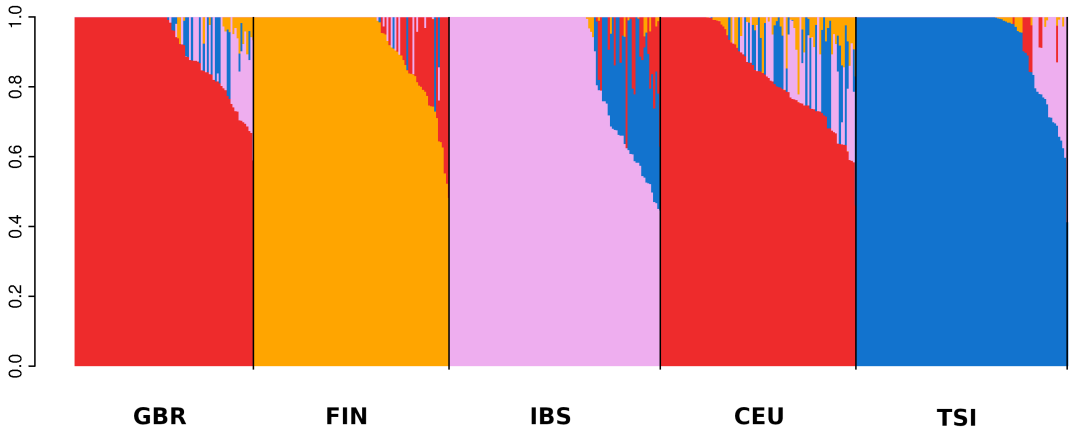


**Figure S13:** Estimated ancestry proportions in the EUR super population of the 1000 Genomes Project using `HaploNet` for $K = 4$ based on the high density genotype chip of the Omni platform used in the 1000 Genomes Project.

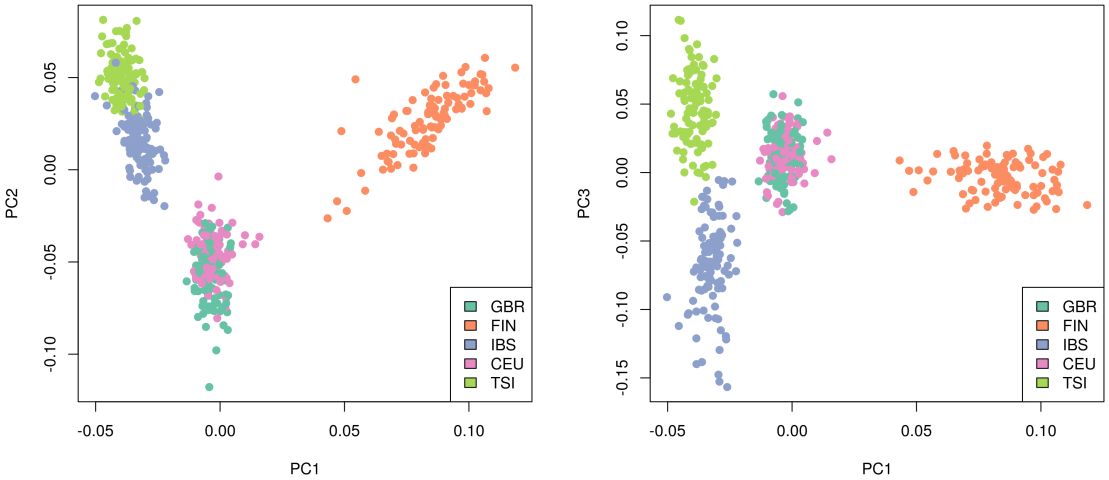


**Figure S14:** Inferred population structure in the EUR super population based on PCA using `HaploNet` based on the high density genotype chip of the Omni platform used in the 1000 Genomes Project. First plot displays PC1 plotted against PC2 and second plot shows PC1 plotted against PC3.

## S7.5 UK Biobank

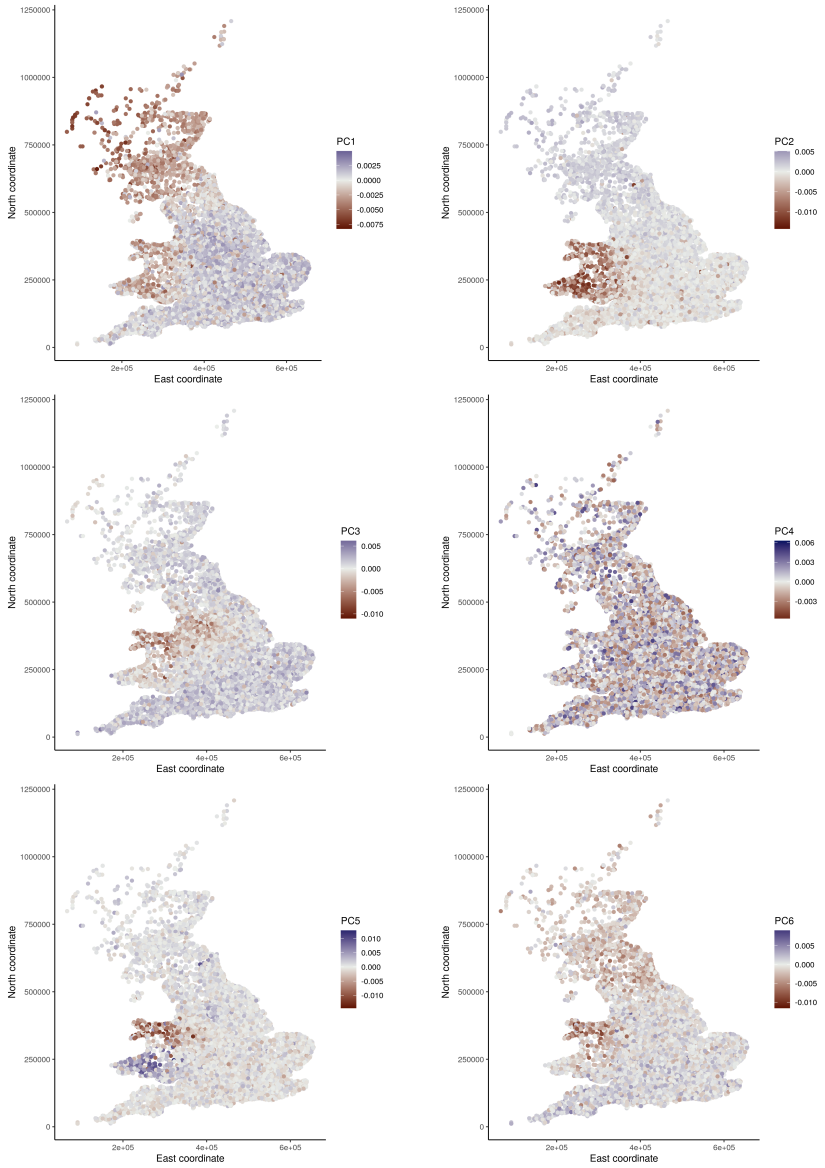


**Figure S15:** Inferred population structure in the subset of unrelated self-identified 'White-British' based on PCA using `HaploNet`. Individuals are plotted by their birthplace coordinates and colored by their inferred PC scores from PC1-PC6.

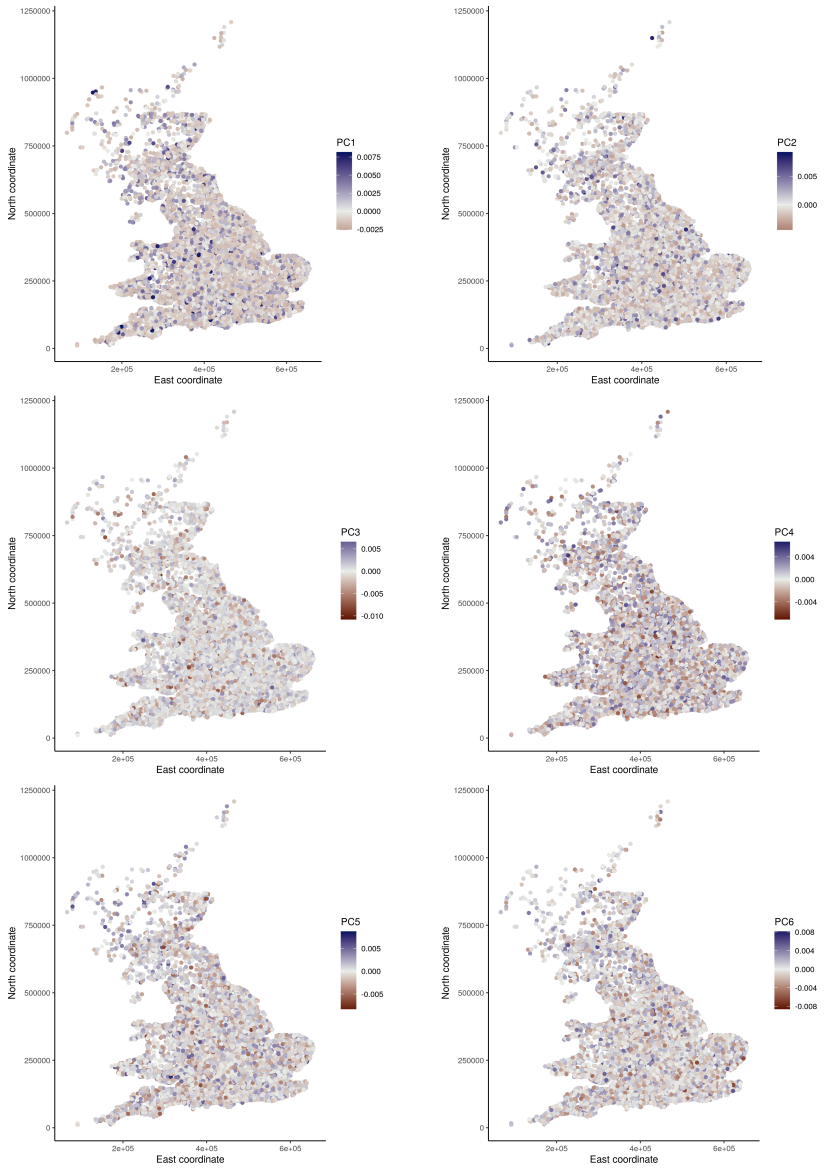


**Figure S16:** Inferred population structure in the subset of unrelated self-identified 'White-British' based on PCA using `PLINK`. Individuals are plotted by their birthplace coordinates and colored by their inferred PC scores from PC1-PC6.

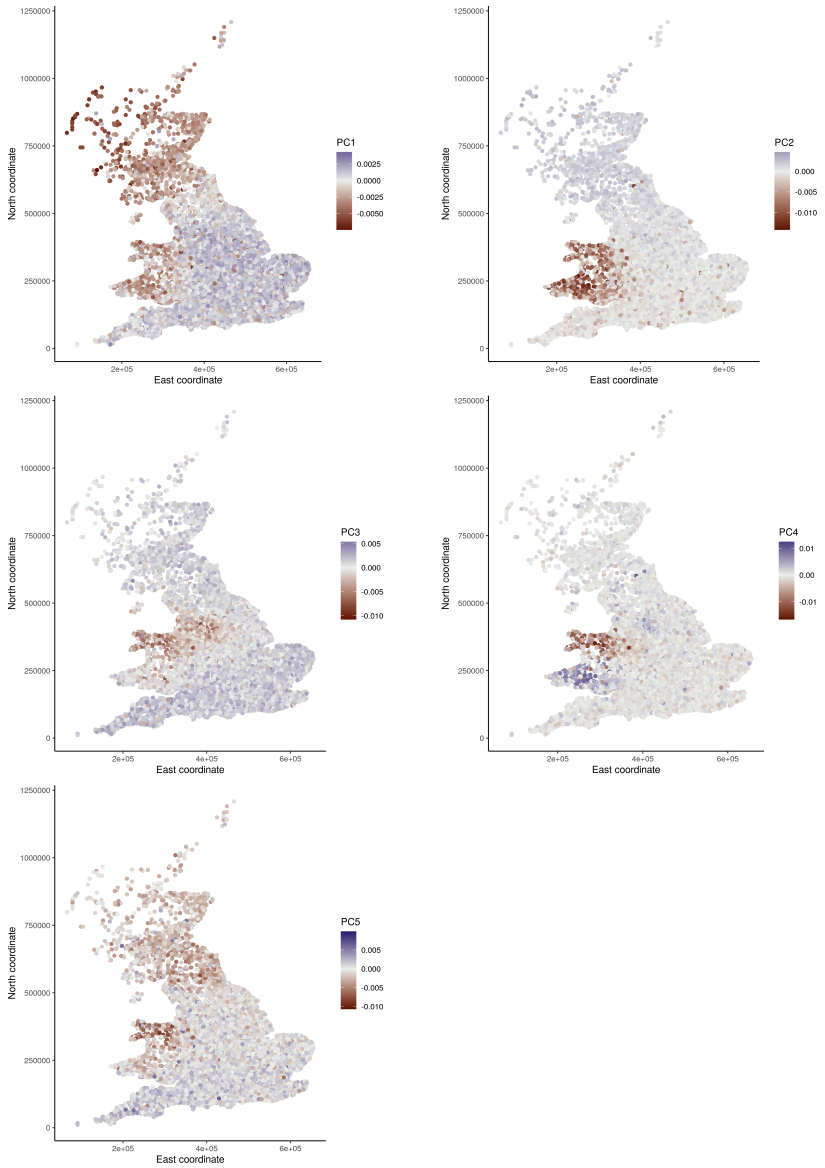


**Figure S17:** Inferred population structure in the subset of unrelated self-identified 'White-British' based on PCA using `PLINK` with strict LD pruning. Individuals are plotted by their birthplace coordinates and colored by their inferred PC scores from PC1-PC6.