1. Sparse factor analysis

Let n be the number of individuals in a sample and p be the number of genotypes. Represent each allele at a locus as a number (e.g., for SNPs from a diploid organism, as in our results above, represent AA as 0, AB as 1 and BB as 2). Our factor analysis model with K factors can be written as:

$$G_{i,j} = \mu_{i,j} + \sum_{k=1}^{K} \Lambda_{i,k} F_{k,j} + \epsilon_{i,j}, \tag{1}$$

or, equivalently,

$$G_{i,j} \sim \mathcal{N}(\mu_{i,j} + (\Lambda F)_{i,j}, \psi_{i,j}^{-1}) \tag{2}$$

where G is an $n \times p$ data matrix, the mean term $\mu_{i,j}$ is the sum of row- and column-specific means: $\mu_{i,j} = \nu_i + \xi_j$, Λ is the $n \times K$ matrix of *factor loadings*, F is the $K \times p$ matrix of *factors*, and ϵ is an $n \times p$ matrix with each element independently distributed $\epsilon_{i,j} \sim \mathcal{N}(0, \psi_{i,j}^{-1})$ and is the product of a row- and a column-specific variance term $\psi_{i,j} = \theta_i \eta_j$. We put a gamma prior on the inverse residual variance that acts as a regularizer: $\theta_i \sim Ga(\alpha, \beta)$ and $\eta_j \sim Ga(\kappa, \tau)$, which has mean $\alpha\beta$ ($\kappa\tau$, respectively) and variance $\alpha\beta^2$ ($\kappa\tau^2$, respectively). In practice, we set $\alpha = \kappa = 1$ and $\beta = \frac{20}{p}$, $\tau = \frac{20}{n}$. This model, with only mean term ξ_j, is referred to as *SFAm* in the main text; the SFA model is obtained by fixing the vector μ at zero. For both SFA and SFAm, as described in the paper, we fix $\eta_j = 1$ for $j = 1...p$. The ECME algorithm for fitting the general SFA model is described below; the ECME algorithm for fitting SFA is obtained by simply setting $\mu_{i,j} = 0$ and $\eta_j = 1$ throughout. Note that here we have chosen to have column-specific (i.e., SNP-specific) means and row-specific (i.e., individual-specific) variances Ψ. The other possible options are implemented in the software. In some contexts, including the population structure problem considered here, it might make sense to allow more general assumptions, such as variance terms on both the rows and columns of the matrix, but we leave their evaluation to future work.

To induce sparsity in the factor loadings Λ, we use an automatic relevance determination (ARD) prior (**?**). Specifically, we assume $\Lambda_{i,k} \sim \mathcal{N}(0, \sigma_{i,k}^2)$, where the matrix $\Sigma = (\sigma_{i,k}^2)_{i=1,\ldots,n,k=1,\ldots,K}$ is a parameter that we estimate, together with the other parameters, using maximum likelihood. If the estimate of $\sigma_{i,k}^2 = 0$, this implies that $\Lambda_{i,k} = 0$, thus inducing sparsity.

Integrating out Λ, the rows of G are conditionally independent given the other parameters, with:

$$G_{i,\cdot} \sim \mathcal{N}(\mu, F^t \Sigma_i F + \Psi_i^{-1}), \tag{3}$$

where $\Sigma_i = diag(\sigma_{i,\cdot}^2)$ (a diagonal matrix with the K-vector $\sigma_{i,\cdot}^2$ on the diagonal), and $\Psi_i^{-1} = diag(\theta_i^{-1}\eta_1^{-1}, \ldots, \theta_i^{-1}\eta_p^{-1})$. Thus the log marginal likelihood for the parameters μ, F, Σ, Ψ is:

$$\begin{aligned} \mathcal{L}(\mu, F, \Sigma, \Psi; G) &:= \log p(G|\mu, F, \Sigma, \Psi) \tag{4} \\ &= -\sum_{i=1}^{n} \frac{1}{2}\left[p \log(2\pi) + \log |F^t \Sigma_i F + \Psi_i^{-1}| + \tilde{G}_{i,\cdot}^t (F^t \Sigma_i F + \Psi_i^{-1})^{-1} \tilde{G}_{i,\cdot}\right], \tag{5} \end{aligned}$$

where $\tilde{G}_{i,j} := G_{i,j} - \mu_{i,j}$.

2. Sparse factor analysis ECME algorithm

We fit this model using an expectation conditional maximization either (ECME) algorithm (**?**) to maximize $\mathcal{L}(\mu, F, \Sigma, \Psi; G)$. This algorithm is similar to an EM algorithm, but each maximization step maximizes either the expected log likelihood, or the marginal log likelihood, for a subset of the parameters conditional on the others. Specifically, the updates to μ, F, and Ψ involve maximizing the expected log likelihood (with the expectation taken over Λ), whereas the updates to Σ directly maximize the log marginal likelihood.

To compute the expected log likelihood requires the first and second moments of the factor loadings $\Lambda_{i,\cdot}$. The data $G_{i,\cdot}$ and the loadings $\Lambda_{i,\cdot}$ are jointly normal (as in, e.g., Ghahramani and Hinton (**?**)):

$$\begin{bmatrix} G_{i,\cdot} \\ \Lambda_{i,\cdot} \end{bmatrix} \Bigg| \mu, F, \Sigma_i, \Psi_i \sim \mathcal{N}\left(\begin{bmatrix} \mu \\ \mathbf{0}_K \end{bmatrix}, \begin{bmatrix} F^t \Sigma_i F + \Psi_i^{-1} & F^t \Sigma_i \\ \Sigma_i F & \Sigma_i \end{bmatrix} \right), \tag{6}$$

where $\mathbf{0}_K$ is a K-vector of zeros. Standard results for joint Gaussian distributions give the conditional expectation for $\Lambda_{i,\cdot}$:

$$\overline{\Lambda_i} := E\left[\Lambda_{i,\cdot}|G_{i,\cdot}, \mu, F, \Sigma_i, \Psi_i\right] = \Omega_i \tilde{G}_{i,\cdot}, \tag{7}$$

where $\Omega_i = \Sigma_i F (F^t \Sigma_i F + \Psi_i^{-1})^{-1}$. Similarly, the conditional second moment is given by:

$$\overline{\Lambda_i^2} := E[\Lambda_{i,\cdot} \Lambda_{i,\cdot}^t | G_{i,\cdot}, \mu, F, \Sigma_i, \Psi_i] = \Sigma_i - \Omega_i F^t \Sigma_i + \Omega_i \tilde{G}_{i,\cdot} \tilde{G}_{i,\cdot}^t \Omega_i^t. \tag{8}$$

The updates for μ, F, and Ψ involve maximizing the expected complete data log likelihood, $\mathcal{Q}(\mu, F, \Sigma, \Psi; G) := E[\log(p(G|\Lambda, \mu, F, \Psi))|\Sigma]$, which from Equation 2 is given by:

$$\mathcal{Q}(\mu, F, \Sigma, \Psi; G) = \text{const} + \sum_{i=1}^{n} \mathcal{Q}_i(\mu, F, \Sigma_i, \Psi_i; G_{i,\cdot}) \tag{9}$$

where

$$\begin{aligned} \mathcal{Q}_i(\mu, F, \Sigma_i, \Psi_i; G_{i,\cdot}) &= \Big(\frac{p}{2} + p(\alpha - 1)\Big)\log(\theta_i) + \Big(\frac{p}{2} + p(\kappa - 1)\Big)\sum_{j=1}^{p}\log(\eta_j) \\ &- \psi_i \frac{1}{2}\sum_{j=1}^{p}\Big(\tilde{G}_{i,j}^2 - 2\tilde{G}_{i,j}F_{\cdot,j}^t\overline{\Lambda_i} + F_{\cdot,j}^t\overline{\Lambda_i^2}F_{\cdot,j} - \Big(\frac{\eta_j}{\tau}\Big)\Big) - \Big(\frac{\theta_i}{\beta}\Big). \end{aligned} \tag{10}$$

Taking the derivative of $\mathcal{Q}(\mu, F, \Sigma, \Psi; G)$ with respect to ν_i and setting to 0, we get the update for ν_i:

$$\frac{\partial \mathcal{Q}(F, \Sigma, \Psi, \mu; G_{i,\cdot})}{\partial \nu_i} = \frac{1}{2}\sum_{j=1}^{p}\psi_{i,j}\left(-2(G_{i,j} - \mu_{i,j}) + 2F_{j,\cdot}^t\overline{\Lambda_i}\right) = 0 \tag{11}$$

$$\hat{\nu}_i = \frac{\sum_{j=1}^{p}\eta_j(G_{i,j} - \xi_j - F_{j,\cdot}^t\overline{\Lambda_i})}{\sum_{j=1}^{p}\eta_j}. \tag{12}$$

Taking the derivative of $\mathcal{Q}(\mu, F, \Sigma, \Psi; G)$ with respect to ξ and setting to 0, we get the update for ξ:

$$\frac{\partial \mathcal{Q}(F, \Sigma, \Psi, \mu; G)}{\partial \xi} = \frac{1}{2}\sum_{i=1}^{n}\Psi_i\left(-2(G_{i,\cdot} - \mu) + 2F^t\overline{\Lambda_i}\right) = 0 \tag{13}$$

$$\hat{\mu} = \frac{\sum_{i=1}^{n}\theta_i(G_{i,\cdot} - \xi_i + F^t\overline{\Lambda_i})}{\sum_{i=1}^{n}\theta_i}. \tag{14}$$

In these expressions, and in what follows, we are assuming element-wise multiplication when a scalar multiplies a vector or a matrix.

Taking the derivative of $\mathcal{Q}(\mu, F, \Sigma, \Psi; G)$ with respect to $F_{\cdot,j}$ and setting to zero, we get the update for $F_{\cdot,j}$:

$$\frac{\partial \mathcal{Q}(F, \Sigma, \Psi, \mu; G)}{\partial F_{\cdot,j}} = \sum_{i=1}^{n}\Psi_i(\tilde{G}_{i,j}\overline{\Lambda_i} - \overline{\Lambda_i^2}F_{\cdot,j}) = 0$$

$$\hat{F}_{\cdot,j} = \left(\sum_{i=1}^{n}\theta_i\overline{\Lambda_i^2}\right)^{-1}\sum_{i=1}^{n}\theta_i\tilde{G}_{i,j}\overline{\Lambda_i}. \tag{15}$$

Taking the derivative of $\mathcal{Q}(F, \Sigma_i, \Psi_i, \mu; G_{i,\cdot})$ with respect to θ_i and setting to zero, we get the update for θ_i:

$$\hat{\alpha}_i = \left[\frac{1}{p + 2p(\alpha - 1)}\sum_{j=1}^{p}\eta_j\left(\tilde{G}_{i,j}^t\tilde{G}_{i,j} - 2\tilde{G}_{i,j}F_{\cdot,j}^t\overline{\Lambda_i} - F_{\cdot,j}^t\overline{\Lambda_i^2}F_{\cdot,j}\right) + \frac{2}{\beta}\right]^{-1}. \tag{16}$$

We find the updates for η_j similarly:

$$(17)\qquad \hat{\eta}_j = \left[\frac{1}{n + 2n(\kappa - 1)} \sum_{i=1}^{n} \theta_i \left(\tilde{G}_{i,j}^t \tilde{G}_{i,j} - 2\tilde{G}_{i,j} F_{\cdot,j}^t \overline{\Lambda_i} - F_{\cdot,j}^t \overline{\Lambda_i^2} F_{\cdot,j} \right) + \frac{2}{\tau} \right]^{-1}.$$

To update $\sigma_{i,k}^2$ we can use the result from Tipping and Faul (**?**) to obtain the values of Σ that maximize the log marginal likelihood $\mathcal{L}(\mu, F, \Sigma, \Psi; G)$ with fixed values of μ, F, and Ψ:

$$(18)\qquad \hat{\sigma}_{i,k}^2 = [(q_{i,k}^2 - s_{i,k})/s_{i,k}^2]_+$$

where $q_{i,k}^2 = F_k^t \beta_{\neg k,i}^{-1} \tilde{G}_{i,\cdot}$ and $s_{i,k} = F_k^t \beta_{\neg k,i}^{-1} F_k$, where $\beta_{\neg k,i} = (F^t \Sigma_{i,\neg k} F) + \Psi_i^{-1}$ and $\Sigma_{i,\neg k} = diag(\sigma_{i,1}^2, ..., \sigma_{i,k-1}^2, 0, \sigma_{i,k+1}^2, ..., \sigma_{i,K}^2)$. Note that $[a]_+ = a$ when $a > 0$ and $= 0$ otherwise. This works because, given F, the SFA model (Equation 1) is essentially the sparse regression model considered in Tipping and Faul (**?**) with F playing the role of the covariates. We implement the simplified version of this equation from **?**. In practice, we recomputed the C matrix (here, the β_i matrix) for each $k = 1...K$ for only the first few iterations of the method, then each iteration is much faster if we computed β_i once for all $k = 1...K$, with very minimal impact on the results or convergence time (in the code, the cutoff is set to 5 iterations).

Note that F and Σ are non-identifiable in that multiplying the k^{th} row of F by a constant c and dividing the k^{th} column of Σ by c^2 will not change the likelihood (Equation 4). To deal with this we impose an identifiability constraint, $\frac{1}{p}\sum_{j=1}^{p}(F_{k,j} - \bar{F}_{k,\cdot})^2 = 1$ for $k = 1, ..., K$, where $\bar{F}_{k,\cdot} = \frac{1}{p}\sum_{j=1}^{p} F_{k,j}$. Specifically, after each iteration we divide every element of $F_{k,\cdot}$ by its standard deviation c_k, and multiply the k^{th} column of Σ by c_k^2.

Similarly, the row and column means ν_j and ξ_j are non-identifiable in that adding a constant c to one set of means is equivalent to subtracting c from the other set of means in terms of the matrix product $\nu\mathbf{1}_p + \xi\mathbf{1}_n$. Intuitively, one might want to center the means for the individuals or samples at zero, allowing genes or SNPs to have non-centered means. We can choose to center one of the mean vectors at zero by subtracting the mean term $\bar{\nu}$ from each of the n elements of ν, which centers those mean terms, and then adding this same term $\bar{\nu}$ to each of the p elements of ξ, which results in the same matrix of mean terms as the uncentered terms.

The last non-identifiability constraint we must deal with are the residual variance terms, $\psi_{i,j} = \theta_i \eta_j$. When $p > n$ we look to see whether the largest element of θ minus the smallest element of θ is greater than 3.0; if so, we scale θ by a constant so that $\theta_{max} - \theta_{min} = 3.0$, and we scale η by this same constant. Similarly, when $n > p$, we check that $\eta_{max} - \eta_{min} \leq 3.0$, and, if not, we rescale the η vector so that this equation holds, and we scale θ by this same constant for consistency. We recognize that this is not an ideal way to induce identifiability, but it appears to give reasonable solutions in practice.

In some situations with certain data sets, we found SFA to converge to a spike in the likelihood, where a single individual's genotype is modeled almost perfectly by the factors (with a correspondingly low residual variance term). If this happens, there might be an issue with non-uniformity of variance across the loci (in which case, removing alleles with very low MAF may help). Also trying different random starting points might find a good solution. Another possibility is to put the variance terms on the loci instead of the individuals, or using SFAm. Finally, if the same individual continues to be modeled repeatedly with different random starting points, this person might be removed from the data set.

Because we choose not to update the expected values of the loading matrix Λ between the CM steps, monotone convergence of the log marginal likelihood is not guaranteed, although in practice it appears to converge well. We find that convergence is reached for the applications described here after fewer than 500 iterations. For each genotype data set, we run SFA multiple times with random seeds, setting the number of factors as described in the text; results presented in figures are a representative example. A C++ package containing the general SFA code is available for download at `http://stephenslab.uchicago.edu/software.html`, along with some data sets for application.