1 Introduction

This lecture concerns the problem of inferring the pair of haplotypes for each given genotype over multiple loci. We first consider the so-called pure parsimony criterion, which formulates the inference task as that of finding a minimum number of haplotypes that can "explain" all the given genotypes without errors. This problem turns out to be computationally hard. We then consider a softer variant of the problem, in which the task is to find a probability distribution over all possible haplotypes so as to maximize a natural likelihood function for the given genotypes; this model allows for errors. While this problem is hard as well, it can be successfully, even though somewhat heuristically, approached by an expectation–maximization algorithm.

Generic methodology to be learned in this lecture (by examples): problem formulation; hardness proof by reduction; greedy algorithm; exact exponential-time algorithm by integer linear programming; probabilistic modeling; EM algorithm; Yates's algorithm.

2 Pure parsimony

For simplicity we consider m biallelic markers and encode the two alleles at each locus by 0 and 1. Thus a haplotype is a binary string of length m. The genotype at a locus is encoded by the sum of the two alleles, that is, by 0 or 2 if the genotype is homozygous, and by 1 if it is heterozygous.[1] Thus a genotype is a string of length m over $\{0, 1, 2\}$. We say that the haplotypes h and h' *explain* a genotype g if $g = h + h'$; we may then also say that the *pair* $\{h, h'\}$ explains g, even if $h = h'$.

Example 1. *Let g be the genotype* 012 *and g' the genotype* 111*. Then g is explained by exactly one haplotype pair, namely,* $\{001, 011\}$*, but g' is explained by four different pairs:* $\{000, 111\}$*,* $\{001, 110\}$*,* $\{010, 101\}$*, and* $\{011, 100\}$*.*

More generally we say that a set of haplotypes H *explains* a set of genotypes G if for each genotype $g \in G$ there is a pair $\{h, h'\} \subseteq H$ that explains g.

Definition 1. *The* pure parsimony problem *is as follows. Given a set of genotypes G, find a smallest possible set of haplotypes H that explains G.*

Example 2. *Let G be the genotype set* $\{012, 111\}$*. Given G as input to the pure parsimony problem, there are exactly two valid solutions: the haplotype set* $\{001, 011, 110\}$ *and* $\{001, 011, 100\}$*.*

Since the pure parsimony problem is a kind of a set cover problem, it is not surprising that the problem is computationally hard. Indeed, we can show that the problem is, in a sense, as hard as the classic vertex cover problem, and thereby, as hard as any so-called NP-complete problem.[2]

[1] In the literature you may see a different encoding: 0 and 1 for homozygous and 2 for heterozygous genotypes

[2] We do not assume that the reader is familiar with the notions of NP-hardness and NP-completeness.

2.1 As hard as the vertex cover problem

Definition 2. *The* vertex cover problem *is as follows. Given a graph* (V, E) *and an integer* k*, does there exist a vertex subset* $C \subseteq V$ *of size at most* k *such that every edge in* E *has at least one of its ends in* C*?*

We show that the decision variant of the pure parsimony problem—that is, deciding whether there are k' haplotypes that explain a given set of genotypes—is as hard as the vertex cover problem under polynomial-time reductions.

Let (V, E) and k fix an instance of the vertex cover problem. We map (V, E) and k to an instance of the pure parsimony problem by letting $k' = |V| + k$ and defining a set G of $|V| + |E|$ genotypes over $m = |V| + 1$ SNPs indexed by the vertices in $V \cup \{t\}$, where t is a dummy vertex, as follows. For each $v \in V$ define the genotype $g^{(v)}$ by

$$g_u^{(v)} = 0 \text{ if } u = v \text{ or } u = t, \quad g_u^{(v)} = 2 \text{ otherwise.}$$

For each $e \in E$ define the genotype $g^{(e)}$ by

$$g_u^{(e)} = 1 \text{ if } u \text{ is an endpoint of } e \text{ or } u = t, \quad g_u^{(e)} = 2 \text{ otherwise.}$$

Note that each $g^{(v)}$ can be explained only by the respective haplotype in which each 2 is replaced by a 1.

Lemma 1. *Let* C *be a vertex cover of* (V, E)*. Then there exist a set of* $|V| + |C|$ *haplotypes explaining* G*.*

Proof. For each $v \in V$ define the haplotype $h^{(v)} \in \{0, 1\}^{|V|+1}$ by $h_u^{(v)} = 0$ if and only if $u = v$ or $u = t$, also define $h^{[v]} \in \{0, 1\}^{|V|+1}$ by $h_u^{[v]} = 0$ if and only if $u = v$. Let H consist of the haplotypes $h^{(v)}$ for $v \in V$ and $h^{[v]}$ for $v \in C$. Since C is a vertex cover of (V, E) each genotype $g^{(e)}$ is explained by the haplotype pair $(h^{(u)}, h^{[v]})$ where u and v are the two ends of e and $v \in C$. □

It remains to show the other direction.

Lemma 2. *Let* H *be a set of haplotypes that explains* G*. Then* (V, E) *admits a vertex cover of size* $|H| - |V|$*.*

Proof. (Omitted.) □

From the above lemmas if follows that, if one can solve the decision version of the pure parsimony problem in time polynomial in the number of genotypes and markers, then one can also show the vertex cover problem in time polynomial in the number of vertices and edges of the input graph, which in turn is considered unlikely (as many believe P is not equal to NP).

2.2 Clark's algorithm

The earliest algorithm for haplotype inference from genotype data is attributed to Clark, in 1990. Clark's algorithm can be viewed as a gredy heuristic for the pure parsimony problem. The algorithm infers the haplotypes in three steps:

1. identifying all unambiguous haplotypes from genotypes that all-site homozygotes or single-site heterozygotes, and considering them as “inferred;”
2. determining whether each of the inferred haplotypes could be one of haplotypes that explains a remaining yet-to-be-phased genotype; and
3. each time a new haplotype is identified as the other member in a pair that explains a genotype, the haplotype is considered “inferred” and added to the set of inferred haplotypes.

We leave the formulation of this method as a proper algorithm as an exercise.

The rationale for Clark’s algorithm is that haplotypes that form all-site homozygous genotypes are probably common, and that any other genotype is likely to contain at least one of these known common haplotypes.

2.3 Gusfield’s integer linear programming model

A straightforward algorithm for the pure parsimony problem goes through all the possible solutions $H \subseteq \{0,1\}^m$ and checks (efficiently) whether H explains the given genotype set G. This can be very slow if m is larger than, say, 5, since the number of possible subsets H grows as 2^{2^m}, that is, doubly exponentially in m.

However, Gusfield observed that the pure parsimony problem can be solved to optimum on relatively large instances (say, $m = 50$ markers and $n = 50$ genotypes) by formulating it as an integer linear program (ILP) and resorting to existing (generic) ILP solvers.

In general, an IPL program consists of (i) *integer-valued* variables, (ii) *linear* inequalities on the variables, and (iii) a *linear* cost function to be minimized. The state-of-the-art generic methods to solve ILPs use branch-and-bound recurrence, the bounds obtained efficiently by relaxation to an associated (real) linear program. The existing ILP solvers often run relatively fast on typical problem instances. The method is beyond the scope of this lecture.

In Gusfield’s ILP model, there is one indicator variable x_h for each possible haplotype $h \in \{0,1\}^m$ and one indicator variable $y_{h,h'}$ for each pair (h, h') of distinct haplotypes in $\{0,1\}^m$. Each x_h and $y_{h,h'}$ takes values from $\{0,1\}$, stemming from the idea that $x_h = 1$ if and only h is in the optimal solution H, and that $y_{h,h'} = 1$ if and only if $\{h, h'\}$ explains at least one of the genotypes in the given set G. *Gusfield’s ILP model*:

$$\begin{array}{rcl}
\text{Minimize} & & \displaystyle\sum_{h\in\{0,1\}^m} x_h \\
\text{subject to} & & \\
\displaystyle\sum_{h+h'=g} y_{h,h'} & \geq & 1 \quad \forall g \in G \\
y_{h,h'} & \leq & x_h \quad \forall h, h' \in \{0,1\}^m \\
y_{h,h'} & \leq & x_{h'} \quad \forall h, h' \in \{0,1\}^m \\
x_h & \in & \{0,1\} \quad \forall h \in \{0,1\}^m \\
y_{h,h'} & \in & \{0,1\} \quad \forall h, h' \in \{0,1\}^m .
\end{array}$$

The following proposition states that the solution to Gusfield’s ILP model, indeed, readily corresponds to a solution to the pure parsimony problem.

Proposition 1. *Let x_h be as in Gusfield's ILP model. Then the set $H = \{h : x_h = 1\}$ is a valid solution to the pure parsimony problem given G as the input.*

Proof. (Exercise.) □

3 Maximum likelihood

The pure parsimony problem formulates the haplotype inference task in plain combinatorial terms. As usual, there is a probabilistic alternative that is similar in spirit but is derived by treating the input genotypes, the *data*, as an outcome of a random sampling process, governed by parameters whose values are unknown and to be inferred from the data. A key object in this approach is the *likelihood function* that maps the parameters (their values) to the probability of the data. Usually the parameter inference is implemented by maximizing the likelihood function, whence the name *maximum likelihood* (ML) method.

The ML method for haplotype inference that we will next consider treats the population frequencies p_h of the all possible haplotypes $h \in \{0,1\}^m$ as the parameters and compose the probability model for the genotypes by assuming that (i) the genotypes are independent draws from the population, (ii) the haplotype frequencies are in Hardy–Weinberg equilibrium, and (iii) the given genotypes at each marker may be corrupted versions of the underlying true genotypes due to noise (genotyping errors or sporadic mutations), independent over the loci and with fixed parameter values. We will also consider a practically relevant variant where instead of (i) we assume that (i') the genotypes form independent mother–father–child trios in each of which the transmission of the haplotypes from the parents to the child follow Mendelian rules. Once the popluation frequencies of the haplotypes are inferred from the genotype data, the most-probable haplotype pair can be efficiently inferred for each input genotype.

3.1 Unrelated genotypes

Consider the two haplotypes of an individual drawn randomly from the population. Under HWE, the probability that the individual's paternal haplotype is h and the maternal haplotype is h' is given by $p_h p_{h'}$. Thus, the probability that the individual's genotype is g is obtained by adding up the products $p_h p_{h'}$ over all haplotype pairs (h, h') such that $h + h' = g$. When the population is very large, the genotypes in the sampled data G can be reasonably modelled as independent draws by replacement. The probability of the data G, in other words, the likelihood of the haplotype frequencies, p, becomes

$$L(p; G) = \prod_{g \in G} \Pr(g; p) = \prod_{g \in G} \sum_{h+h'=g} p_h p_{h'} , \tag{1}$$

where the notation $\Pr(g; p)$ makes it explicit that the probability of the genotype depends on the haplotype frequency parameters p. Note that, strictly speaking, this expression ignores a multinomial term that can be viewed constant when G is considered fixed.

While the above likelihood expression assumes that the haplotype pair explains the genotype without errors, it is easy to extend the expression to allow for such errors. Assuming that the errors at locus s is governed by the conditional probability distributions $\tau_s(g_s; h_s + h'_s) =$

$\Pr(g_s|h_s, h'_s)$ and that the errors are independent over the loci and the sampled genotypes, the likelihood function becomes

$$L(p;G) = \prod_{g \in G} \sum_{h,h'} p_h p_{h'} \tau(g; h + h'), \qquad \text{with} \quad \tau(g; h+h') = \prod_{s=1}^{m} \tau_s(g_s; h_s + h'_s). \tag{2}$$

One easily notes that if we set $\tau_s(g_s; g_s) = 1$, then (1) is obtained as a special case.

The inference of the error probabilities $\tau_s(g_s; h_s + h'_s)$ from genotype data is a problem in its own; however, we will assume that these probabilities are fixed and known.

We are ready to define the maximum-likelihood variant of the haplotype inference problem.

Definition 3. *The* ML problem *is as follows. Given a set of genotypes G, find haplotype frequencies $p = (p_h : h \in \{0,1\}^m)$ so as to maximize the likelihood $L(p;G)$.*

No efficient (i.e., polynomial-time) algorithm is known for this problem.

In the remainder of this section we consider the so-called expectation–maximization (EM) algorithm that often works well in practice, even though it is only guaranteed to find a local maximum, not necessarily a global maximum of the likelihood function.

An EM algorithm for unrelated genotypes

Expectation–maximization (EM) is a generic method for maximizing a likelihood function. The method relies on the assumption that there are *hidden data* that, if observed together with the actual data, would make it relatively easy to find the parameter values that maximize the joint probability (density) of the actual and the hidden data, that is, the joint likelihood. Having such hidden data introduced, the EM algorithm starts from some initial values for the parameters and iteratively updates the parameter values so as to maximize the expectation of the logarithm of the joint likelihood, where the expectation is taken with respect to the distribution of the hidden data given the actual data and the current parameter values.

In our case, it is natural to let the hidden data, H, consist of the true haplotype pairs, each genotype in the data associated with one such pair; clearly, once the haplotype pairs are known, the maximum-likelihood haplotype frequencies are obtained by straightforward counting. Now that the haplotype pairs are, however, unknown, it turns out that the expected log-likelihood is maximized by haplotype frequencies that are proportional to the expected haplotype counts. We next put this into a formal statement (Proposition 2), for which we then give a proof.

First note that the joint probability of the genotype data G and the hidden haplotype data H is given by

$$\Pr(G,H) = \Pr(H)\Pr(G|H) = \prod_{g \in G} \Big(p_h p_{h'} \tau(g; h+h') \Big).$$

Let then $p^{(t)}$ denote the haplotype frequency parameters at the tth iteration. The function to be maximized at iteration $t+1$ is given as

$$Q(p^{(t+1)}; p^{(t)}) = \sum_{H} \Pr(H|G; p^{(t)}) \log \Pr(G, H; p^{(t+1)});$$

here we show explicitly the parameters that specify the probability model.

Proposition 2. *The function $Q(p^{(t+1)}; p^{(t)})$ is maximized by setting*

$$p_h^{(t+1)} = \frac{1}{n} \sum_{g \in G} I_h(g)/L(g)\,,$$

with

$$\begin{aligned} I_h(g) &= \Pr(h, g; p^{(t)}) = \sum_{h'} p_h^{(t)} p_{h'}^{(t)} \tau(g; h + h')\,, \\ L(g) &= \Pr(g; p^{(t)}) = \sum_h I_h(g)\,, \end{aligned}$$

where h is the maternal haplotype of the observed genotype g.

Proof. Write p for the "current parameters" $p^{(t)}$ and p' for the "new parameters" $p^{(t+1)}$. Now

$$\begin{aligned} Q(p'; p) &= \mathrm{E}(\log(\Pr(H; p') \Pr(G|H))|G; p) \\ &= \mathrm{E}(\log \Pr(H; p')|G; p) + \mathrm{E}(\log \Pr(G|H))|G; p)\,, \end{aligned}$$

where the expectations are over H given G and p. Since the second term in the sum is constant with respect to p', it suffices to consider the first term. By letting $h(g)$ and $h'(g)$ denote, respectively, the maternal and the paternal haplotype associated in H with a genotype g in G, we write

$$\begin{aligned} Q'(p'; p) &= \mathrm{E}(\log \Pr(H; p')|G; p) \\ &= \mathrm{E}(\sum_{g \in G} \log p'_{h(g)} p'_{h'(g)}|G; p) \\ &= \sum_{g \in G} \mathrm{E}(\log p'_{h(g)} p'_{h'(g)}|G; p) \\ &= \sum_{g \in G} \mathrm{E}(\log p'_{h(g)}|G; p) + \mathrm{E}(\log p'_{h'(g)}|G; p) \\ &= 2 \sum_{g \in G} \mathrm{E}(\log p'_{h(g)}|G; p)\,. \end{aligned}$$

To find p' that maximizes the above function, we introduce the Lagrange multiplier λ to represent the constraint $\sum_h p'_h = 1$. That is, we maximize the function $\tilde{Q}(p'; p) = Q'(p'; p) + \lambda(1 - \sum_h p'_h)$ by setting its partial derivatives $\partial \tilde{Q}(p'; p)/\partial p'_h$ to zero. This yields

$$\lambda = 2 \sum_{g \in G} \Pr(h(g) = h|G; p)(1/p'_h)\,.$$

Put otherwise,

$$\begin{aligned} p'_h &= (2/\lambda) \sum_{g \in G} \sum_{h'} \Pr(h(g) = h, h'(g) = h'|G; p) \\ &= (2/\lambda) \sum_{g \in G} \Pr(h, g; p)/\Pr(g; p)\,. \end{aligned}$$

Since $2 \sum_{g \in G} \Pr(h(g) = h|G; p)$ is the expected number of genotypes g in G that have $h(g) = h$, the sum of these terms over all possible haplotypes h must be $2n$. This means that the normalizing constant λ must be equal to $2n$. This completes the proof. □

How fast can we implement the maximization step as given in Proposition 2? A straightforward algorithm computes the values $I_h(g)$ for each $h \in \{0,1\}^m$ and $g \in G$, which takes time $O(|G|4^m)$. This is quadratic in the output size, the number of possible haplotypes. Later we will give a nearly linear-time algorithm.

3.2 Trios

The ideas we just developed for unrelated genotype data can be extended to data that consist of mother–father–child trios, that is, genotype trios $\mathbf{g} = (g^{\mathrm{m}}, g^{\mathrm{f}}, g^{\mathrm{c}})$, where g^{m}, g^{f}, and g^{c} are the genotypes of a mother, a father, and their child. (For pedagocical reasons, we intentionally structure the following presentation differently than in the previous section.)

While a single genotype is determined by two unobserved haplotypes, the three genotypes in a trio are determined by four unobserved haplotypes: the two haplotypes of the mother and the father that are transmitted to the child, denoted by i and j , respectively, and the untransmitted haplotypes of the mother and the father, denoted by i and j', respectively; the child's two haplotypes are thus i and j.

Assuming HWE the probability of observing $\mathbf{g}$ becomes

$$L(\mathbf{g}) = L(p; \mathbf{g}) = \sum_{i,i',j,j'} p_i p_{i'} p_j p_{j'} \tau(g^{\mathrm{m}}; i + i') \tau(g^{\mathrm{f}}; j + j') \tau(g^{\mathrm{c}}; i + j) \,,$$

where i, i', j, j' run through all haplotypes in $\{0,1\}^m$; we call this the likelihood for $\mathbf{g}$. The following "partial likelihoods," in which one of the haplotypes is fixed, will be useful later when we develope an EM algorithm:

$$\begin{aligned}
I_i(\mathbf{g}) &= \sum_{i',j,j'} p_i p_{i'} p_j p_{j'} \tau(g^{\mathrm{m}}; i + i') \tau(g^{\mathrm{f}}; j + j') \tau(g^{\mathrm{c}}; i + j) \,, \\
I'_{i'}(\mathbf{g}) &= \sum_{i,j,j'} p_i p_{i'} p_j p_{j'} \tau(g^{\mathrm{m}}; i + i') \tau(g^{\mathrm{f}}; j + j') \tau(g^{\mathrm{c}}; i + j) \,, \\
J_j(\mathbf{g}) &= \sum_{i,i',j'} p_i p_{i'} p_j p_{j'} \tau(g^{\mathrm{m}}; i + i') \tau(g^{\mathrm{f}}; j + j') \tau(g^{\mathrm{c}}; i + j) \,, \\
J_{j'}(\mathbf{g}) &= \sum_{i,i',j} p_i p_{i'} p_j p_{j'} \tau(g^{\mathrm{m}}; i + i') \tau(g^{\mathrm{f}}; j + j') \tau(g^{\mathrm{c}}; i + j) \,.
\end{aligned}$$

The likelihood for the entire trio data G, denoted as $L(p; G)$, is simply the product of the $L(\mathbf{g})$ over the trios genotypes $\mathbf{g}$ in G.

The maximum-likelihood problem for unrelated data is modified to trio data in an obvious way:

Definition 4. *The* ML problem for trios *is as follows. Given a set of trio genotypes G, find haplotype frequencies $p = (p_h : h \in \{0,1\}^m)$ so as to maximize the likelihood $L(p; G)$.*

An EM algorithm for trios

The developement of an EM algorithm for trios is analogous to the one for unrelated genotypes. Now, the hidden data, H, consist of the true haplotypes quadruples (i, i', j, j'), one for each trio. The joint probability of the genotype trio data G and the hidden haplotype data H is given as

$$\Pr(G,H) = \Pr(H)\Pr(G|H) = \prod_{\mathbf{g}\in G}\Big(p_i p_{i'} p_j p_{j'} \tau(g^{\mathrm{m}}; i+i')\tau(g^{\mathrm{f}}; j+j')\tau(g^{\mathrm{c}}; i+j)\Big)\,.$$

The function to maximized at iteration $t+1$ is given by

$$Q_{\mathrm{trios}}(p^{(t+1)}; p^{(t)}) = \sum_H \Pr(H|G; p^{(t)}) \log \Pr(G,H; p^{(t+1)})\,.$$

The following results gives a way to carry out the maximization step.

Proposition 3. *The function $Q_{\mathrm{trios}}(p^{(t+1)}; p^{(t)})$ is maximized by setting*

$$p_h^{(t+1)} = \frac{1}{4n}\sum_{\mathbf{g}\in G}\big(I_h(\mathbf{g}) + I'_h(\mathbf{g}) + J_h(\mathbf{g}) + J'_h(\mathbf{g})\big)/L(\mathbf{g})\,.$$

Proof. (Omitted.) □

A straightforward algorithm implements the maximization step by computing the partial likelihoods for each trio genotype by summing over all possible quadruples in time $O\big((2^m)^4 m\big) = O(16^m m)$, from which the complete likelihood is easily computed. However, it is not difficult to reduce the time requirement to $O(4^m m)$. A factor of m comes from the need of comparing a haplotype to e genotype marker by marker. In the next section we show that time $O(2^m m)$ suffices.

3.3 Evaluation of the complete and partial likelihoods

We next give a fast algorithm to compute the partial likelihood $I_i(\mathbf{g})$ for all $i \in \{0,1\}^m$ given a trio genotype $\mathbf{g}$ and fixed haplotype frequencies p_h, for $h \in \{0,1\}^m$. The techniques immediately apply to the computation of the three other partial likelihoods and the complete likeihood; so we omit the details. Also, we leave it as an easy exercise to apply to techniques to the simpler case of unrelated genotypes, that is, to compute the partial likelihoods $I_h(g)$ and the complete likelihood $L(g)$ given a genotype g and fixed haplotype frequencies.

A key observation is that the expression of the partial likelihood $I_i(\mathbf{g})$ is a sum of products where each factor in the product depends on at most two of the four haplotypes. For convenience we introduce the shorthands

$$\alpha_{ii'} = \tau(g^{\mathrm{m}}; i+i')\,, \qquad \beta_{jj'} = \tau(g^{\mathrm{f}}; j+j')\,, \qquad \gamma_{ij} = \tau(g^{\mathrm{c}}; i+j)\,,$$

for $i, i', j, j' \in \{0,1\}^m$. Now, we may write

$$I_i(\mathbf{g}) = \sum_{i'}\sum_{j}\sum_{j'} p_i p_{i'} p_j p_{j'} \alpha_{ii'}\beta_{jj'}\gamma_{ij} = p_i\Big(\sum_{i'} p_{i'}\alpha_{ii'}\Big)\Big(\sum_j p_j\gamma_{ij}\Big(\sum_{j'} p_{j'}\beta_{jj'}\Big)\Big)\,.$$

This immediately suggests the following algorithm for the evaluation of the expression.

1. Compute $a_i := \sum_{i'} p_{i'} \alpha_{ii'}$ for all $i \in \{0,1\}^m$.
2. Compute $b_j := \sum_{j'} p_{j'} \beta_{jj'}$ for all $j \in \{0,1\}^m$.
3. Compute $c_i := \sum_{j} p_{j} \gamma_{ij} b_j$ for all $i \in \{0,1\}^m$.
4. Return $p_i a_i c_i$ as the value of $I_i(\mathbf{g})$ for all $i \in \{0,1\}^m$.

The algorithm is obviously correct (that is, it computes the correct output). Steps 1–3 take time $O(4^m m)$ and step 4 in time $O(2^m m)$. We next show how the first three steps can be implemented much faster.

We proceed by viewing each of the steps 1–3 as a multiplication of a 1×2^m vector by a $2^m \times 2^m$ matrix. Since these multiplication tasks are essentially equivalent, we only consider step 3. Of course, a runtime of order 4^m would be unavoidable if the matrix (γ_{ij}) had no structure. However, we note that, by (2), the matrix is a direct (or Kronecker or tensor) product of m matrices, each of size 2×2; we will clarify this soon.

Definition 5. *Let $M^1, M^2, \ldots, M^r$ be matrices of size $t \times t$. Then the* direct product $M = M^1 \otimes M^2 \otimes \cdots \otimes M^r$ *is a $t^r \times t^r$ matrix, whose elements are given by*

$$M_{i_1+i_2t+\cdots+i_rt^{r-1}, j_1+j_2t+\cdots+j_rt^{r-1}} = M^1_{i_1,j_1} M^2_{i_2,j_2} \cdots M^r_{i_r,j_r} \, .$$

Note that parenthezation is not needed, since direct product is an associative operation.

Returning to the matrix γ, notice that, by (2), it is the direct product of matrices $\gamma^1, \gamma^2, \ldots, \gamma^m$, defined by $\gamma^s_{i_s,j_s} = \tau_s(g^{\mathrm{c}}_s; i_s + j_s)$ for $i_s, j_s \in \{0,1\}$.

The direct product structure enables efficient evaluation of the vector-by-matrix product—the algorithm is known as Yates's algorithm:

Lemma 3. *If M is a direct product of r matrices of size $t \times t$ and v is a $t^r \times 1$ vector, then the product Mv can be evaluated in time $O(rt^{r+1})$.*

Proof (sketch). Let $M = M^1 \otimes M^2 \otimes \cdots \otimes M^r$. For an index $i = i_1 + i_2 t + \cdots + i_r t^{r-1}$ we may write also $i_1 i_2 \cdots i_r$ for convenience. Write

$$\begin{aligned} (Mv)_i &= \sum_j M_{ij} v_j = \sum_{j_1} M^1_{i_1,j_1} \sum_{j_2} M^2_{i_2,j_2} \cdots \sum_{j_r} M^r_{i_r,j_r} v_j \\ &= \sum_{j_1} M^1_{i_1,j_1} \sum_{j_2} M^2_{i_2,j_2} \cdots \sum_{j_{r-1}} M^{r-1}_{i_{r-1},j_{r-1}} v'_{j_1 j_2 \cdots j_{r-1} i_r} \, , \end{aligned}$$

where

$$v'_{j_1 j_2 \cdots j_{r-1} i_r} = \sum_{j_r} M^r_{i_r,j_r} v_j \, .$$

We can "eliminate" j_r, that is, compute the vector v' in time $O(t^{r+1})$. Repeating analogous elimination steps $r-1$ more times will result in the the vector Mv in time $O(rt^{r+1})$. □

We have thus shown that the partial likelihoods $I_i(\mathbf{g})$, for all haplotypes i simultaneously, can be computed in time $O(m2^m m)$.

Exercises

III:1 Formulate Clark's algorithm as a pseudocode. What does your algorithm output if the following genotypes are given as the input: 0000, 2222, 1111, 0101, 0112, 1020? What is the worst-case time complexity of the algorithm in terms of the number of genotypes n and the number of markers m?

III:2 Prove Proposition 1.

III:3 When the number of (biallelic) markers m is large, the number of potential haplotypes, 2^m, may prohibit the application of the presented techniques for haplotype inference. There is a popular heuristic approach, called *partition ligation*, to address this issue. The idea is to infer the possible candidate haplotypes separately for the first and the last $m/2$ markers; usually the number of haplotypes with nonzero frequency estimates is much below $2^{m/2}$ for each of the two parts, say ℓ and r for the first and the last part, respectively. Then the haplotype frequencies over the complete set of m markers is inferred under the supposition that non-zero frequencies are hold only by haplotypes that belong to the $k := \ell r << 2^m$ possible combinations of the already inferred shorther haplotypes.

Show that each iteration in the EM algorithm for estimating the frequencies of the k possible haplotypes can be done in time $O(k^2 m)$ (per observed trio genotype).