Exercises II

All numbered references refer to the lecture notes of week II.

II:1 Prove (1). Also, write the probability as a function of the number of heterozygous genotypes $x = n_{Aa}$ (and the n_A and n) alone.

II:2 Suppose n genotypes are sampled at a locus and you want to test for HWE. How fast can you evaluate (a) Fisher's exact test, assuming there are two alleles at the locus, (b) χ^2 test, assuming there are m alleles at the locus? We assume that basic arithmetical operations with two numbers take time $O(1)$.

II:3 Bob has got data consisting of 10,000 genotypes over 6 biallelic loci. He wants to compute the contingency tables for each of the $\binom{6}{2}$ pairs of loci as fast as possible. How would you suggest Bob to process his data?

Solution to II:1

Because the samples are independent draws from the population, the distributions of the allele counts follow the multinomial (binomial) distributions:

$$\begin{aligned}\Pr(n_{AA}, n_{Aa}, n_{aa}) &= \binom{n}{n_{AA}\ n_{Aa}\ n_{aa}} p_{AA}^{n_{AA}} p_{Aa}^{n_{Aa}} p_{aa}^{n_{aa}}\,,\\ \Pr(n_A, n_a) &= \binom{2n}{n_A\ n_a} p_A^{n_A} p_a^{n_a}\,;\end{aligned}$$

note that while the number of samples (genotypes) is n, the number of alleles is $2n$.

Under HWE, that we assume to hold, we have $p_{AA} = p_A^2$, $p_{Aa} = 2p_Ap_a$, and $p_{aa} = p_a^2$. Furthermore, the genotype counts and the allele counts satisfy $n_A = 2n_{AA} + n_{Aa}$ and $n_a = 2n_{aa} + n_{Aa}$. Subtitution gives

$$\begin{aligned}\Pr(n_{AA}, n_{Aa}, n_{aa}) &= \binom{n}{n_{AA}\ n_{Aa}\ n_{aa}} p_A^{2n_{AA}} p_A^{n_{Aa}} p_a^{n_{Aa}} p_a^{2n_{aa}} 2^{n_{Aa}}\,,\\ \Pr(n_A, n_a) &= \binom{2n}{n_A\ n_a} p_A^{2n_{AA}+n_{Aa}} p_a^{2n_{aa}+n_{Aa}}\,.\end{aligned}$$

Now, by taking the ratio of these probabilities the factors involving the allele frequencies p_A and p_a cancel out, and by writing the multinomial coefficients in terms of factorials yields the claimed expression:

$$\Pr(n_{AA}, n_{Aa}, n_{aa}|n_A, n_a) = \frac{\Pr(n_{AA}, n_{Aa}, n_{aa})}{\Pr(n_A, n_a)} = \frac{n!n_A!n_a!2^{n_{Aa}}}{n_{AA}!n_{Aa}!n_{aa}!(2n)!}\,.$$

Write $x = n_{Aa}$. Since $n_a = 2n - n_A$, $n_{AA} = (n_A - x)/2$, and $n_{aa} = (n_a - x)/2$ we have

$$\Pr(n_{AA}, n_{Aa}, n_{aa}|n_A, n_a) = \Pr(n_{Aa}|n, n_A) = \frac{n!n_A!(2n-n_A)!2^x}{((n_A - x)/2)!x!((2n - n_A - x)/2)!(2n)!}\,.$$

Solution to II:2

Consider the following algorithm.

(a)

1. For $k = 1, 2, 3, \ldots, 2n$ precompute the factorials $k!$ iteratively to an array $f[k]$ (by $f[k] = kf[k-1]$ and $f[0] = 1$).
2. Compute $q = \Pr(n_{Aa}|n, n_A)$ using the expression derived in exercise II:1 and the array $f[.]$.
3. Let $x = \min\{n_A, 2n - n_A\}$. Let $s = 0$.
4. Repeat
   - Compute $r = \Pr(x|n, n_A)$ using the expression derived in exercise II:1 and the array $f[.]$.
   - If $r \leq q$ then let $s = s + r$.
   - Let $x = x - 2$.

   Until $x < 0$.
5. Return s.

Observe that the algorithm returns the p-value needed for Fisher's exact test. Steps 1 and 4 take time $O(n)$, while the other steps take time $O(1)$. Thus, the test can be computed in time $O(n)$.

(b) Suppose a list of n genotypes at a single locus with m alleles are given as input. Write n_{ij} for the number of genotypes consisting of the ith and the jth allele, $1 \leq i \leq j \leq m$, and $\hat{p}_i = (2n_{ii} + \sum_{i<j} n_{ij})/(2n)$ for the sample frequency of the ith allele. Observe that under HWE, that we assume to hold, the expected genotype counts are obtained as $e_{ii} = n\hat{p}_i^2$ and $e_{ij} = n2\hat{p}_i\hat{p}_j$ for $i < j$.

Now write the chi-squared test statistic as

$$X^2 = \sum_{i \leq j} (e_{ij} - n_{ij})^2 / e_{ij} \, .$$

For a while, suppose the numbers n_{ij} and e_{ij} have been counted and listed for every ij; this can be obviously done in time $O(n + m^2)$ by using an array of size $\binom{m}{2} + m$ (an entry for each ij with $i \leq j$). Then X^2 can be evaluated in a straightforward manner in time $O(m^2)$, thus in time $O(n + m^2)$ in total.

We can, however, reduce the running time to $O(n + m)$ by using a couple of tricks. The basic idea is to consider only genotypes ij that appear in the input, that is, $n_{ij} > 0$. To this end, write

$$\begin{aligned}
X^2 &= \sum_{i\le j: n_{ij}>0} (e_{ij} - n_{ij})^2/e_{ij} + \sum_{i\le j: n_{ij}=0} e_{ij} \\
&= \sum_{i\le j: n_{ij}>0} (e_{ij} - n_{ij})^2/e_{ij} + \sum_{i\le j} e_{ij} - \sum_{i\le j: n_{ij}>0} e_{ij} \\
&= n \sum_i \sum_j \hat{p}_i \hat{p}_j + \sum_{i\le j: n_{ij}>0} (e_{ij} - n_{ij})^2/e_{ij} - e_{ij} \\
&= n + \sum_{i\le j: n_{ij}>0} (n_{ij}^2 - 2e_{ij}n_{ij})/e_{ij} \\
&= n + \sum_{i\le j: n_{ij}>0} n_{ij}^2/e_{ij} - 2n_{ij}\,.
\end{aligned}$$

It remains to show that the numbers n_{ij} and e_{ij} can be computed and listed, exactly once for each ij with $n_{ij} > 0$, in time $O(n)$. We can do this by three passes through the input. Before the first pass we allocate memory for an array A with one entry for each ij. In the first pass, we set $A[ij] = 0$ for every genotype ij read from the input list (during this pass we may also count the alleles). In the second pass, we increment $A[ij]$ whenever the genotype ij is read from the input list. In the third pass, the first time ij is read—this is when $A[ij] > 0$—the term $n_{ij}^2/e_{ij} - 2n_{ij}$ is computed and added to a variable x (initialized to n), and $A[ij]$ is set to 0. Finally x is returned as the target value of X^2.

Solution to II:3

We consider two approaches to produce the contingency tables from the genotype list given as input. For the comparison of the time complexities of these approaches it is useful to treat the number of markers, m, and the number of sampled genotypes, n, as variables, eventhough the exercise concerns the particular case of $m = 6$ and $n =$10,000.

The first approach produces each of the $\binom{m}{2}$ contingency tables separately for each marker pair by one pass through the genotype list. The number of additions scales roughly as $\binom{m}{2}n$ and the total number of basic operatons is $O(\binom{m}{2}n)$.

The second approaches takes only one pass through the genotype list. An array A, with an entry for each possible genotype g over the m loci, is initialized to zeros, and then $A[g]$ is incremented by one every time g is read from the input list. This first phase takes roughly n additions and in a total of $O(3^m + n)$ basic operations. The contingency table for each pair st is obtained by simply adding up the entries $A[g]$ for fixed one-locus genotypes g_s and g_t; this takes roughly $\binom{m}{2}3^m$ additons. Thus, in total, the number of basic operatons needed in this approach is $O(\binom{m}{2}3^m + n)$.

The hidden constant factors being about equal, we find that the second approach is faster if $\alpha(m, n) := \binom{m}{2}n$ is larger than $\beta(m, n) := \binom{m}{2}3^m + n$. With the particular values $m = 6$ and $n =$10,000, we have $\alpha(6, 10000) = 150000$ and $\beta(6, 10000) = 20935$. Thus, the second approach is roughly 7.5 times faster than the first approach.

Exercises III

All numbered references refer to the lecture notes of week III.

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 shorter haplotypes.

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

Solution to III:1

```
Input: a set of genotypes G.
Output: a set of haplotypes H supposed to explain some genotypes in G.
1: H := {};
2: for each g in G do
  2.1: if g is heterozygous in at most one SNP, then
    2.1.1: add the two haplotypes of g to H;
3: G := clean(G, H, H);
4: H' := H;
5: while H' not empty do
  5.1: remove a h from H';
  5.2: for each g in G do
    5.2.1: h' := g-h;
    5.2.2: if h' is a valid haplotype, then
      5.2.2.1: add h' to H and H';
      5.2.2.2: G := clean(G, {h'}, H);
      5.2.2.3: break (5.2);
6: return H.

function clean(G, I, J):
1: G' := G;
2: for each (i, j) in I x J do
  2.1: G' := G' \ {i+j};
3: return G'.
```

The algorithm works as follows. First (1) the haplotype set is set to the empty set. Then (2) all haplotypes that can be directly inferred from some genotype in G are inserted to H. Next (3) every genotype that can be explained by some pair of haplotypes from H is removed from G. Then (4) H' is set to the inferred haplotypes that may be needed in explaining some remaining genotypes in G. Next in a loop each haplotype in H' is considered in turn and removed (5.1); the haplotype is compared to each genotype in H: if the difference $h' = g - h$ is a valid haplotype, then h' is inserted to both H' and H' and not only the g but also every other genotype that can be explained by the h' combined with an already inferred haplotype in H are removed from G (5.2.2). Such "cleaning" is executed to ensure that a genotype that can be explained with the just introduced h' will not generate unnecessary new haplotypes. Because no haplotype that is removed from H' will be reinserted to H', the time complexity is $O(|H|^2|G|m)$, where m is the number of markers. Since $|H|$ can be at most $2|G| = 2n$ the running time is $O(n^3m)$. Note that if the "cleaning" procedure is omitted, then the running time is only $O(n^2m)$.

Suppose the genotypes 0000, 2222, 1111, 0101, 0112, 1020 are given as input for the algorithm. Then the algorithm first infers that haplotypes 0000, 1111, 1010, and 0010, and the genotypes 0000, 2222, 1111, and 1020 are removed: the genotypes 0101 and 0112 remain to be explained. Next, say, haplotype 0000 is picked and genotype 0101 is explained and removed by introducing a new haplotype 0101. Because haplutype 0011 has not yet been inferred, the cleaning phase does not explain and remove the remining genotype 0012. However, eventually the just introduced haplotype 0101 is picked and genotype 0112 is explained by introducing a new haplotype 0011. Thus, the haplotype set $\{0000, 1111, 1010, 0010, 0101, 0011\}$ is returned at the end.

Solution to III:2

Recall *Gusfield's ILP model*:

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

We will prove the following. The haplotype set $H = \{h : x_h = 1\}$ is a valid solution to the pure parsimony problem, given that x_h is part of a solution to Gusfield's ILP model. First we show that H explains G. Then we show the other direction, namely that, if H explains G, then there is such a configuration of the x and y variables that satisfies the ILP model and yield H as the set $\{h : x_h = 1\}$. These together imply that $H = \{h : x_h = 1\}$ is the smallest possible set that explains G.

Lemma 1. *H explains G.*

Proof. Let $g \in G$. Because $\sum_{h+h'=g} y_{h,h'} \geq 1$ and $y_{h,h'} \in \{0,1\}$, there is at least one pair

(h, h') such that $y_{h,h'} = 1$ and $h + h' = g$. Because $y_{h,h'} \leq x_h, x_{h'}$, we have that $\{h, h'\}$ is a subset of H. Since g was arbitrary, we have that H explains G. □

Lemma 2. *Suppose H explains G. Then there exist a configuration of the x and y variables that satisfies the ILP model and $H = \{h : x_h = 1\}$.*

Proof. We construct such a configuration. First, set $x_h = 1$ if and only if $h \in H$. Then, set $y_{h,h'} = 1$ if and only if $\{h, h'\} \subseteq H$ and $h + h' \in G$. Note that $y_{h,h'} \leq x_h, x_{h'}$ clearly holds. Now, let $g \in G$. Because H explains G, we have a pair $\{h(g), h'(g)\} \subseteq H$ with $h(g) + h'(g) = g$. Thus, $\sum_{h+h'=g} y_{h,h'} \geq y_{h(g),h'(g)} \geq 1$, which completes the proof, □

Solution to III:3

One sees that the algorithms given in the lecture notes apply and that a haplotype in $\{0, 1\}^m$ need not be considered (but can be skipped) if its frequency is 0. As the frequencies are, by the partition ligation construction, nonzero for at most k haplotypes, the running time reduces from $O((2^n)^2 m)$ to $O(k^2 m)$. (Note that a factor of m was errorneously missing from the exercise statement.)

Exercises IV

IV:1 Does the following sample form a haplotype block under (a) the Patil et al. definition (b) the Wang et al. definition (four gamete test)?

```
11100
01011
00?11
1?100
01011
10101
1010?
010?1
?1001
11001
01011
```

IV:2 Suppose the missing data in the above exercise (IV:1) are replaced by 1s. (a) Which haplotypes are uniquely distinguished by the first two SNPs? (b) Find a tagging set for the haplotypes according to the Patil et al. definition with $\beta = 0.80$.

IV:3 Suppose the values $f(t, j)$ have been computed and stored into a respective array for all $1 \leq t \leq m$ and $1 \leq j \leq k$. Present a pseudo code for an algorithm that, given such an array as input, constructs and outputs an optimal disjoint set of blocks, that is, an I such that $K(I) \leq k$ and $L(I) = f(m, k)$. What is the asymptotic running time of your algorithm in terms of m, k, and the maximum block length M?

Solution to IV:1

(a) Observe first that there are four ambiguous haplotypes: `1?100` because it is compatible with both `11100` and `1010?`; `1010?` because it is compatible with both `10101` and `1?100`; `010?1` because it is compatible with both `01011` and `?1001`; `?1001` because it is compatible with both `11001` and `010?1`. Thus, we remove *all* these four haplotypes from the sample, despite the fact that already removing one of these would render some of the remaining three unambiguous. The remaining haplotypes, grouped, are

```
11100

01011
01011
01011

00?11

10101

11001
```

We see that only 3 out of the all 7 haplotypes occur at least twice in the sample. Because $3/7 < 0.80$, the sample is not a haplotype block under the Patil et al. definition: too large a fraction of the haplotypes are unique in the sample.

(b) The sample is not a haplotype block under the Wang et al. definition either, because all four gametes occur already at the first two SNPs (e.g., haplotypes 1, 2, 3, and 6).

Solution to IV:2

After replacing the missing data by 1s and grouping identical haplotypes, the sample reads

```
11100
11100

01011
01011
01011

00111

10101
10101

11001
11001
```

(a) Clearly the haplotypes that start with `11` are not uniquely distinguished by the first two SNPs, since there are two different such groups. On the contrary, the haplotypes starting with any of the three other combinations are seen to be uniquely distinguished.

(b) Because there are five haplotype groups, no pair of SNPs can uniquely distinguish all the haplotypes in the sample. That is, with any fixed pair of SNPs, at least two groups will be merged into one. Now because the smallest group sizes are 1 and 2, at least $1 + 2 = 3$ haplotypes will not be uniquely distinguished; thus two SNPs do not suffice for uniquely dinstinguishing $80\% > 8/11$ of the haplotypes. This means that at least three SNPs are needed. The first three SNPs is a tagging set, since they actually uniquely dinstinguish all the 11 haplotypes.

Solution to IV:3

The following pseudo code describes an algorithm that constructs a set of disjoint blocks I such that $L(I) = f(m, k)$ and $K(I) \leq k$.

```
Input: Functions f, B, K, and L, and integers m, k, and M.
Output: A set of disjoint blocks I with L(I) = f(m, k) and K(I) <= k.
t := m; j := k; I := {};
while t > 0 do
    if f(t-1, j) = f(t, j) then
        t := t - 1;
    else
        s := t;
        while f(s-1, j-K(s, t)) + L(s, t) != f(t, j) or B(s, t) = 0 do
            s := s - 1;
        t := s - 1; j := j - K(s, t); I := I U [s, t];
return I.
```

The idea is simple. The algorithm backtracks the dynamic programming recurrence; as soon as $f(t, j)$ matches (i.e., equals) either $f(t-1, j)$ or $f(s-1, j-K(s, t)) + L(s, t)$ the index t is decreased accordingly to $t-1$ or $s-1$, and the set I updated with the corresponding block, if any. Because every marker t is visited only once, associated with a constant number of basic operations, the running time of the algorithm is $O(m)$. Note that neither k nor M play a role in the running time bound.

Exercises V

V:1 (a) Extend the definition of the pure parsimony problem to allow for a deletion haplotype. (Hint: redefine $\{h, h'\}$ *explains* g.) (b) What modifications are needed in Gusfield's ILP model to make it valid for the extended pure parsimony problem?

V:2 Show that the values $I_i(\mathbf{g})$ for all $i \in H_1 = \{0, 1\}^m \cup \{\mathrm{d}\}$ can be computed in time $O(m2^m)$, given the trio genotype $\mathbf{g}$ and the model parameters (haplotype frequencies p_h, $h \in H_1$ and the error models τ_s, $s = 1, 2, \ldots, m$) as input.

V:3 Suppose you are given the haplotype frequencies p_h, $h \in H_1$, and the error models τ_s, $s = 1, 2, \ldots, m$. For a random observed genotype $g \in \{0, 1, 2, \mathrm{N}\}^m$, write (in terms of the p_h and τ_s) the conditional probability that the underlying true genotype is (a) a homozygous deletion, (b) a hemizygous deletion, given the observed g. (c) How fast can you compute these probabilities given g as input?

Solution to V:1

(a) We redefine the meaning of "$\{h, h'\}$ explains g" such that it extends to haplotypes $h, h' \in \{0, 1\}^m \cup \{\mathrm{d}\}$ and genotypes $g \in \{0, 1, 2, \mathrm{N}\}^m$, as follows. We say $\{h, h'\}$ *explains* g if $h_s \oplus h'_s = g_s$ for all $s = 1, 2, \ldots, m$, where the binary operation $\oplus$ satisfies $x \oplus y = x + y$ for $x, y \in \{0, 1\}$, $x \oplus \mathrm{d} = \mathrm{d} \oplus x = 2x$ for $x \in \{0, 1\}$, and $\mathrm{d} \oplus \mathrm{d} = \mathrm{N}$.

With this definition, the original formulation of the pure parsimony problem readily extends to the case of a deletion haplotype.

(b) In Gusfields ILP model it suffices to replace $h + h' = g$ by $h \oplus h' = g$ and extend the range of the haplotypes from $\{0, 1\}^m$ to $\{0, 1\}^m \cup \{\mathrm{d}\}$. However, note that if the input contains a genotype that has N at some but not all SNPs, then the genotype cannot be explained by any pair of haplotypes in $\{0, 1\}^m \cup \{\mathrm{d}\}$.

Solution to V:2

We observe that the four-step algorithm given in the lecture notes of Week II readily applies when the haplotype range $H_0 = \{0, 1\}^m$ is replaced by $H_1 = \{0, 1\}^m \cup \{\mathrm{d}\}$. This gives an algorithm to compute the values $I_i(\mathbf{g})$ in time $O(m(2^m + 1)^2) = O(m4^m)$.

To expedite the algorithm it suffices to show how to multiply a $(2^m + 1) \times 1$ vector by a $(2^m+1) \times (2^m+1)$ matrix in time $O(m2^m)$. We, of course, need to exploit the structure in the matrix. Unfortunately, the matrix is not a direct product of m small matrices, and so the results of Week III do not directly apply. However, we can easily break the computational task into three subproblems of which one is exactly the one we have considered in Week III. To see this, let (v_j) be the vector and (γ_{ij}) the matrix. The problem is to compute

$$c_i = \sum_{j \in H_1} \gamma_{ij} v_j$$

for all $i \in H_1$. Now, write

$$c_i = \gamma_{i\mathrm{d}} v_{\mathrm{d}} + \sum_{j \in H_0} \gamma_{ij} v_j \, .$$

Observe that the first term can be computed for all $i \in H_1$ in time $O(2^m)$. The second term can be computed for all $i \in H_0$ in time $O(m2^m)$, since this is exactly the expression studied in Week III. Finally, the second term can be computed for $i = \mathrm{d}$ in time $O(m2^m)$, since we only need to loop over the j. Thus the values c_i for all $i \in H_1$ can be computed in time $O(m2^m)$.

Solution to V:3

(a) The probability that the maternal haplotype h and the paternal haplotype h' underlying the given genotype g is a homozygous deletion is given by

$$\begin{aligned}
\Pr(h = h' = \mathrm{d}|g) &= \Pr(h = h' = \mathrm{d}, g) / \Pr(g) \\
&= \Pr(h = h' = \mathrm{d}) \Pr(g|h = h' = \mathrm{d}) \Big/ \sum_{h,h'} \Pr(h, h', g) \\
&= \Pr(h = h' = \mathrm{d}) \Pr(g|h = h' = \mathrm{d}) \Big/ \sum_{h,h'} \Pr(h, h') \Pr(g|h, h') \\
&= p_{\mathrm{d}}^2 \tau(g; \mathrm{d}, \mathrm{d}) \Big/ \sum_{h,h'} p_h p_{h'} \tau(g; h, h') \,,
\end{aligned}$$

where

$$\tau(g; h, h') = \prod_{s=1}^{m} \tau_s(g_s; h_s, h'_s) \,.$$

(b) Similarly, the probability that the haplotype pair (h, h') underlying the given genotype g is a heterozygous deletion is given by

$$\begin{aligned}
\Pr(h = \mathrm{d} \neq h' \text{ or } h \neq \mathrm{d} = h'|g) &= \Pr(h = \mathrm{d} \neq h' \text{ or } h \neq \mathrm{d} = h', g) / \Pr(g) \\
&= \big(\Pr(h = \mathrm{d} \neq h', g) + \Pr(h \neq \mathrm{d} = h', g)\big) / \Pr(g) \\
&= \big(\sum_{i \neq \mathrm{d}} \Pr(h = \mathrm{d}, h' = i) \Pr(g|\mathrm{d}, i) + \Pr(h = i, h' = \mathrm{d}) \Pr(g|i, \mathrm{d})\big) / \Pr(g) \\
&= \frac{\sum_{i \neq \mathrm{d}} p_{\mathrm{d}} p_i \big(\tau(g; \mathrm{d}, i) + \tau(g; i, \mathrm{d})\big)}{\sum_{h,h'} p_h p_{h'} \tau(g; h, h')} \,.
\end{aligned}$$

If we assume that τ is symmetric with respect to the two haplotypes, the numerator simplifies to $\sum_{i \neq \mathrm{d}} 2 p_{\mathrm{d}} p_i \tau(g; \mathrm{d}, i)$.

(c) By the previous exercise (V:2) we know that expressions of the form

$$e_i = \sum_{j \in H_1} p_j \tau(g; i, j)$$

can be computed for all $i \in H_1$ in time $O(m2^m)$. Thus, the demoninator, $\Pr(g)$, can be computed as

$$\sum_{h \in H_1} p_h e_h$$

in time $O(m2^m)$. The numerators can obviously be computed in a straightforward manner also in time $O(m2^m)$.