

Exact Computation of Coalescent Likelihood Under the Infinite Sites Model

Yufeng Wu

Department of Computer Science and Engineering
University of Connecticut
Storrs, CT 06269, U.S.A.
ywu@engr.uconn.edu

Abstract. Coalescent likelihood is the probability of observing the given population sequences under the coalescent model. Computation of coalescent likelihood under the infinite sites model is a classic problem in coalescent theory. Existing methods are based on either importance sampling or Markov chain Monte Carlo. In this paper, we develop a simple method that can compute the *exact* coalescent likelihood for many datasets of moderate size, including a real biological data whose likelihood was previously thought to be difficult to compute exactly. Simulations demonstrate that the practical range of exact coalescent likelihood computation is significantly larger than what was previously believed.

1 Introduction

Originally developed by Kingman [11], coalescent theory has been a major research subject in population genetics. Coalescent theory provides stochastic genealogical models, and can be applied to answer many biological questions. Coalescent theory has become even more important now because it is one of the major analytical tools for population variation data analysis. Such data is currently being generated rapidly by high throughput genotyping and resequencing technologies.

There are different kinds of genetic data that the coalescent theory can be used to analyze. An important type of genetic marker is single nucleotide polymorphism (SNP). SNP data is the focus of this paper, although the method presented herein can be applied to other types of genetic data. SNP is a single nucleotide site where exactly two (of four) different nucleotides occur in a large percentage of the population. Thus, we simply use 0/1 to represent the nucleotide at a SNP site. A haplotype (also called a SNP sequence in this paper) is a binary vector of m SNPs (listed as s_1, \dots, s_m from left to right). The set of haplotypes sampled from a population is denoted by D , where D has n haplotypes (rows) and m sites (columns). Throughout the paper, we assume at most one mutation occurred at any SNP site during the evolution of the haplotypes, which is supported by the standard “infinite sites model” in population genetics [19]. This is particularly justified for populations that originated not very long

ago where the time-scale of interest is short enough that two mutations at any single site are unlikely.

Coalescent theory has been shown to be very useful in simulating population sequences (e.g. [10]). However, applying coalescent theory to *inference* problems can be challenging. Answering inference problems requires the development of efficient and accurate computational methods based on coalescent theory. Unfortunately, many problems on coalescent theory are computationally difficult. This article focuses on one such problem: compute the likelihood $P(D)$ of population haplotypes D for a given mutation rate.

Assuming that recombination is absent, the only parameter of the coalescent likelihood is the mutation rate. The commonly used scaled population mutation rate is $\theta = 4N\mu$, where N is the effective population size and μ is the mutation rate per haplotype per generation. We assume the entire genomic region under study has the same scaled mutation rate θ . Under the coalescent model, $P(D)$ can be viewed as a *summation* of probability over *all* possible coalescent genealogies. Intuitively, there are many possible gene genealogies that can generate D . Some genealogies are more likely and others are less likely. $P(D)$ is the sum of probability of all the genealogies that derive D . For a given coalescent genealogy (including both topology and timing), it is straightforward to compute $P(D)$ for a given mutation rate. See [17] for some concrete examples. The main difficulty of computing $P(D)$ is that the number of possible genealogies that derive D is often very large. Thus, computing $P(D)$ can be challenging under many genetic models.

Computing $P(D)$ under the coalescent model has been actively studied. Here we assume that only site mutations are considered. That is, other genetic processes such as recombination, migration and selection are neglected. A milestone in population genetics is the discovery by Ewens [3] that $P(D)$ has a closed-form formula under the infinite *alleles* model. No closed-form formula of $P(D)$ is known for the infinite sites model. When there is no recombination, a perfect phylogeny can be constructed using the efficient algorithm by Gusfield [7], assuming the infinite sites model of mutation. The inferred perfect phylogeny is useful because *every* coalescent genealogy must be consistent with the phylogeny. Moreover, the perfect phylogeny is unique up to some flexibility in arranging the site mutations along a single tree branch. However, the perfect phylogeny misses important aspects of the genealogy. First, the perfect phylogeny may contain some internal nodes with more than two descendants. Second and more importantly, the timing information is missing from the perfect phylogeny. The timing information is critical in computing the coalescent likelihood. Thus, although the perfect phylogeny is unique, the number of possible genealogy can still be very large. See Figure 1 for an illustration.

In a seminal paper, Griffiths and Tavarè [4] presented an importance sampling method to compute the coalescent likelihood under the infinite sites model. In 2000, Stephens and Donnelly [15] made significant progress in the importance sampling approach, which leads to significant reduction of variance in the likelihood computation. Alternatively, Kuhner, et al. [12] applied Markov chain Monte

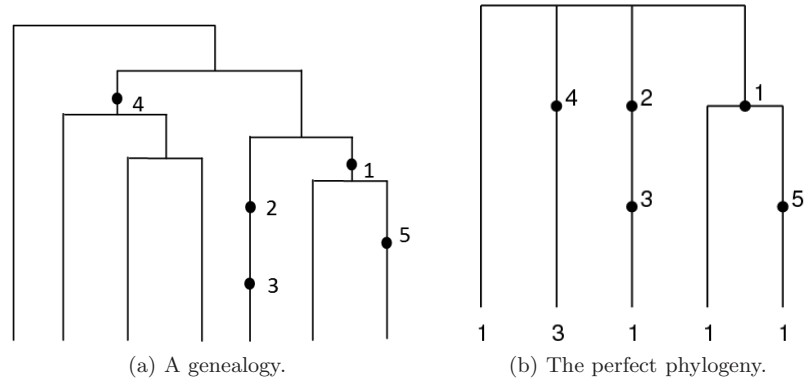


Fig. 1. An example of coalescent genealogy and inferred perfect phylogeny for seven haplotypes and five sites. The filled dots represent the mutation events (and the numbers to the right are the sites). Figure 1(a) is an example of coalescent genealogy. Figure 1(b) is the inferred perfect phylogeny from input haplotypes. The numbers below the tips are the multiplicities of the lineages. The lineage can be represented by the list of site mutations up to the root, with an extra 0 appended to the end. For example, the middle lineage can be represented as [3 2 0]. Note that there are multiple consistent genealogies for this phylogeny.

Carlo to compute the coalescent likelihood. New methods are still being developed for this likelihood computation problem. A recent paper [9] proposed a new importance sampling method to compute $P(D)$ under infinite sites model.

One should note all these methods are stochastic and none is ensured to compute the *exact* $P(D)$. It is well known that $P(D)$ can be computed exactly by solving a set of recursions, which were developed by Ethier, Griffiths and Tavarè in several papers between 1987 to 1995 [2, 4]. This recursion (which we call Ethier-Griffiths-Tavarè recursion or simply EGT recursion) is central to our method. In fact, Griffiths developed a program called *ptree* before the release of program *Genetree* [1, 4]. Program *ptree* can solve the EGT recursion exactly. However, program *ptree* was known to work for relatively small data (say with 15-20 sequences), while importance sampling methods can handle much larger data. This may be one reason that program *ptree* has largely been replaced by program *Genetree*. The common belief is that solving the EGT recursion exactly is usually hard. For example, it was stated in [8] that it may not be practical to find exact solution for EGT recursion when the summation of the number of sites and the number of sequences exceeds 30. The computational difficulty of solving the EGT recursion exactly is often listed as the motivation to develop stochastic methods (see, e.g. [9]).

In this paper, we present an *exact* method for computing the coalescent likelihood $P(D)$ for a given set of population haplotypes D under the infinite sites model of mutations. Only mutation and coalescent events are considered. Other

genetic processes such as recombination are not considered. Different from existing statistical methods [4, 12, 15, 9], our method is *exact* and fully deterministic. Our method is based on the well-known EGT recursion, which is also the foundation of the importance sampling method implemented in program Genetree [1, 4]. Our contribution is to provide a *time* and *memory* efficient implementation that solves the EGT recursion exactly. Although the practical range of our method is still limited, we show that our method can be used to analyze relatively large data including one dataset that was originally analyzed by Griffiths and Tavarè [5] using importance sampling. Likelihood of this data was once thought to be difficult to compute exactly [14]. We show in this paper that computing exact coalescent likelihood is still challenging *but* feasible for a wide range of data D . We show through extensive simulation the practical range of our method is significantly larger than what is previously believed. We also give a brief performance comparison of existing importance sampling methods using the exact method.

2 Method

We now describe our method for the exact likelihood computation under the infinite sites model. The basic idea is very simple: we solve the Ethier-Griffiths-Tavarè (EGT) recursion exactly using a dynamic programming approach. First, we briefly review some basic concepts of coalescent theory that are important to the current work. Then, we present our method for solving the EGT recursion exactly.

2.1 Ethier-Griffiths-Tavarè Recursion

For ease of exposition, we assume that the ancestral state of the genealogy is known to be *all zero* (i.e. the genealogy is rooted and the root is all-zero). It is easy to extend to the unrooted case (as explained in Section 3) and our implementation works with both rooted and unrooted cases. At any given time going *backwards* in time, there are two types of possible events in the genealogy:

1. Coalescent. Two identical sequences coalesce into a single sequence.
2. Mutation. A sequence changes the state at a site (which has not mutated before) from the derived state to the ancestral state.

An important concept in the EGT recursion is ancestral configuration (AC). An AC is essentially a multiset of sequences that exist in the genealogy at a particular point of time. At the present time, the AC is composed of all the input sequences in D . As we trace backwards in time, we may observe different ACs due to the coalescent or mutation events. Following the notations in Wakeley [17], we use the pair (X, v) to represent an AC. Here, X is a *set* of haplotypes and v is the multiplicity (i.e. haplotype count) vector for the haplotypes in X . Each haplotype in X is written as an ordered list of sites that appear on the path

from the haplotype to the root of the genealogy. At the end of each haplotype in X , we append a special site 0. Site 0 can be thought to label a dummy edge out of the root. See Figure 1 for an illustration. We let $p(X, v)$ be the likelihood of the AC for a given θ . At the root of the genealogy, we have $X = [0]$ and $v = [1]$ and $p([0], [1]) = 1.0$. Moreover, we let S_k be an operator that deletes the first element of the k th haplotype of X , and $S_k X$ as the new haplotype set X' after S_k is applied on X . This operator is used to represent the effect of mutations on X . We let R_k be an operator that deletes the k th item in a vector. For example, $R_k X$ as the new haplotype set X' after R_k is applied on X . Finally, we let e_k to be the vector whose k th bit is 1 and the rest is all 0.

We are now ready to give the EGT recursion [2]. This recursion is defined on a finite number of ACs and makes the exact computation of coalescent likelihood feasible. The recursion is based on three operators on the ACs, which correspond to the coalescent event and two types of mutation events in the genealogy. We say a haplotype is *mutable* in (X, v) if its haplotype count is 1 and its first site is unique in haplotypes in X . v_k is the multiplicity of the n -th haplotype in X . We let set A contain the indices of mutable haplotypes which remain distinct after deleting their first site. We let set B contain the indices of all the mutable haplotypes. Set C_k contains the indices of haplotypes that match the new haplotype X'_k after X_k is mutated at its first site. We now have the EGT recursion (See [16, 8, 17] for more information):

$$\begin{aligned} p(X, v) &= \frac{n-1}{\theta+n-1} \sum_{k:v_k \geq 2} \frac{v_k(v_k-1)}{n(n-1)} p(X, v - e_k) \\ &+ \frac{\theta}{\theta+n-1} \sum_{k \in A} \frac{1}{n} p(S_k X, v) \\ &+ \frac{\theta}{\theta+n-1} \sum_{k \in B} \sum_{j \in C_k} \frac{1}{n} p(R_k X, R_k(v + e_j)) \end{aligned}$$

2.2 Solving Ethier-Griffiths-Tavarè Recursion Efficiently

The main difficulty in calculating the *exact* P(D) is that the number of ancestral configurations can be very large for data of even moderate size [14]. Therefore, any exact method based on EGT recursion will eventually become impractical as the data size grows, unless there are some clever ways of reducing the size of the recursion (e.g. merging ACs to get a much smaller recursion). However, as shown in this paper, this infeasibility issue may not be as bad as what is previously believed: a well designed and implemented method can solve the exact EGT recursion on a modern desktop computer for simulated or real biological data that are thought to be difficult before. The key idea of our method is to solve the EGT recursion using *dynamic programming*. That is, we look at the genealogical history *forward* in time. This is different from most coalescent computation approaches (e.g. [4, 15]) which look backwards in time. As we show below, when

solving the EGT recursion under the infinite sites model, looking forward in time is more efficient in both running time and memory.

Recall that the genealogy is tightly constrained by the perfect phylogeny constructed from the input sequences. The perfect phylogeny actually provides all the information we need about what genealogical events (forward in time) are possible for a given AC. Thus we base our computation on the perfect phylogeny, and proceed in *stages*. For n input haplotypes and m sites, there are $n+m$ stages: $0, 1, \dots, n+m-1$. We start our computation for stage 0 from the root of the phylogeny. At stage 0, there is only a single $AC_0 = ([0], [1])$. An AC is said to be in stage i if the summation of the number of coalescent events and mutation events from the AC backwards to AC_0 is equal to i . It is straightforward to find which stage a given AC is: the stage number is equal to the number of haplotypes plus the number of segregating sites minus one. Note that grouping the ACs by the total number of events is natural and is known previously (see e.g. [8]). Such grouping helps to develop a practical and efficient method. At stage i , we maintain a list of all the ACs at stage i . For each $AC_{i,k} = (X, v)$ in this list, we keep the following information:

- $p(X, v)$: the probability of $AC_{i,k}$.
- A list of genealogical events (called active events) that can occur in order to move to stage $i+1$ from this AC.

The set of active events can be thought as a snapshot of the genealogical events specified by the perfect phylogeny. For every internal node of the phylogeny whose out-degree is k , there are $k-1$ coalescent events associated with the node. For each mutation along a phylogeny branch, there is one mutation event associated with the mutation. We assign a unique label (say e_1, e_2, \dots) for each of these genealogical events. At the root, the AC has a single active event: the coalescent event e_1 . A new AC is created for the next stage when one of the active events in the current AC occurs. For an AC, it is straightforward to find the ACs that can be reached from this AC together with their lists of active events. To see this, examine each active event in the current AC. The active events of a new AC after event e occurs are the union of the active events of the current AC and the set of the newly feasible events $E_n(e)$ (and minus e). If e is a coalescent event, $E_n(e)$ contains mutation events at the outgoing branches from the internal node in the phylogeny where the coalescent events occurs, and a coalescent event next to e at the same node in the phylogeny (if there are more coalescent events possible besides e). If e is a mutation event, $E_n(e)$ contains a single event: either (a) a mutation event right below e if e is not the last mutation event along a branch in the phylogeny, or (b) a coalescent event at the internal node right below where e occurs if e is the last mutation event along a branch. Note that we can reduce the number of ACs by fixing the order of mutations along the same branch in the perfect phylogeny. This has been used in program Genetree [1, 4]. It helps to provide an example on how the ACs are maintained in the computation. In Figure 1, consider $AC_1 = ([0], [2])$, which is the single AC at stage 1. The set of active events of AC_1 contains three mutation events

at sites 1, 2 and 4 respectively and a coalescent event at the root of phylogeny. Now suppose we create $AC_{2,1}$ for stage 2 by performing the mutation event at site 2. In this new AC, mutation event at site 3 becomes feasible. The active events of $AC_{2,1}$ are three mutation events at sites 1, 3 and 4 respectively and a coalescent event at the root of phylogeny.

The computation proceeds in stages. Suppose we have generated the complete list of ancestral configurations $AC_{i,k}(X, v)$ at stage i . We then generate the list of $AC_{i+1,k'}(X', v')$ for stage $i + 1$ from the list of $AC_{i,k}(X, v)$. We also set $p(X', v')$ to be the product of $p(X, v)$ and the transition probability. The transition probability depends on the type of the event, which is given in the EGT recursion (see Section 2.1). When the generated AC is already in the list, we merge the two by simply adding the probability of the new AC to the one already in the list, and then discard the newly generated AC.

Since the number of ACs to consider is very large, our implementation is designed to scan through the ACs as quickly as possible. A more serious problem is in the memory since the memory needed to store such a large number of ACs can be very large. Looking forward in time allows us to process each AC just once. A main benefit of this dynamic programming approach is the savings of memory. When an AC is processed and new ACs for the next stage from it have been generated, we can *discard* this AC. This simple technique is similar to what has been previously used in memory efficient pairwise sequence alignment. Moreover, we only keep the minimum amount of information in each AC. Note that such savings of time and space is *not* easily achievable for a method searching backwards in time.

At last, our method can handle the likelihood computation on a grid of θ values in a very simple way. We just keep a list of probability values for each AC, each for a different θ . This allows a much faster computation when maximum likelihood estimate of θ is desired.

3 Results

We implement the exact likelihood computation method in C++. To test the effectiveness of this method, we use it to compute the exact likelihood for both simulated and real biological data. We use our methods to obtain maximum likelihood estimation (MLE) of θ . The experiment was performed on a 3192 MHz Intel Xeon workstation with 16 GB memory.

3.1 Simulated data

We first test our method on simulated data. We use Hudson's program MS to generate datasets with several different settings for haplotypes and scaled mutation rate θ . We simulate 20, 30, 40 or 50 haplotypes, and $\theta = 1, 3$ or 5. For each settings, we generate 100 datasets. We assume the root haplotype is fixed to be all-0 haplotype (which is what program MS uses). The simulation results

Table 1. Exact likelihood computation of simulated data. Average MLE of θ , running time (in seconds), number of ACs are listed (for the datasets where MLE computation is feasible). Some datasets are too large to compute the exact likelihood. So we also report percentage of data where exact computation is feasible.

#rows	θ	%res.	Ave. MLE	Ave. time	Ave. #ACs
20	1	100	1.12	< 0.1 s	906
20	3	100	2.94	0.7 s	10,039
20	5	100	5.04	7.7 s	64,566
30	1	100	1.01	0.2 s	3571
30	3	100	2.92	34.5 s	231,252
30	5	99	4.93	319 s	1,580,873
40	1	100	1.06	1.6 s	15,543
40	3	99	2.88	280 s	1,277,251
40	5	94	4.78	2886 s	11,229,671
50	1	100	1.01	30.3 s	147,789
50	3	95	2.85	2425 s	8,044,357
50	5	74	4.43	9816 s	26,395,143

are listed in Table 1. When θ is small, sometimes there is segregating sites. We treat such cases as computable.

The results in Table 1 indicate that exact likelihood computation is feasible for many moderate sized data. As expected, the running time and the number of ACs increase when the number of rows and θ increase. Exact computation starts to become more difficult when the number of haplotypes is over 50 and θ is over 5.0. Nevertheless, our result shows that exact computation can be practical for the range of data that was deemed to be infeasible before. For example, in Hein, et al. [8], it was stated that exact computation is feasible for moderately sized data, where it was stated “moderately sized would be the number of segregating sites plus the number of haplotypes is less than thirty”. Our experimental results suggest significantly larger data allows exact computation of $P(D)$. Also, Hobolth, et al. [9] tested their importance sampling method for simulated data with 50, 75 and 100 rows and $\theta = 1, 3$ and 5. As shown in Table 1, many of these data allows exact computation.

3.2 A real biological data

We now demonstrate the capability of our method on a real biological data. This is a Mitochondrial data, originally collected in [18] and analyzed by Griffiths and Tavarè in [5]. The original data contains 63 mtDNA sequences. To apply their importance sampling method based on the infinite sites model, Griffiths and Tavarè chose a subset of mtDNA sequences by removing sequences in which mu-

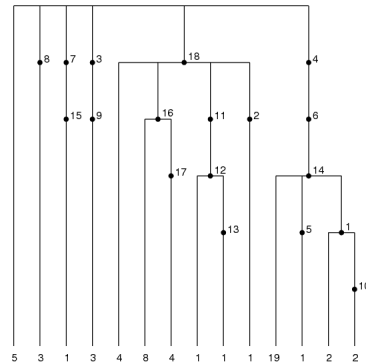


Fig. 2. Perfect phylogeny for Ward, et al. mtDNA data [18]. The numbers at the tree tips are the multiplicities of the lineages.

tations seem to have occurred more than once at some sites. The remaining 55 mtDNA sequences (with 18 segregating sites) are compatible with the infinite sites model. The perfect phylogeny constructed from this data is shown in Figure 2, which is drawn by Griffiths' program Genetree. Song, et al. [14] alluded that exact computation of $P(D)$ for this data may be impractical. We now show that, although the running time is a little long and the amount of memory needed is large, exact computation of the likelihood of this data can be practically computed with moderate amount of computing resource.

Since the ancestral sequence of this mtDNA data was not known, Griffiths and Tavarè treated it as a root-unknown problem. They simply compute the likelihood for each possible rooted phylogeny that is compatible with the unrooted phylogeny, and then sum over the likelihood of all the rooted phylogeny. We take the same approach here. See Table 2 for the likelihood and the running time for each of the 18 possible rooted phylogeny at $\theta^* = 4.8$, which is the MLE of θ .

Table 2. Detailed analysis of the mtDNA data in [18]. For each rooted tree, the number of ACs, the likelihood for $\theta = 4.8$ and running time are listed. Also, if $\theta = 4.8$ does not maximize the probability for the rooted tree, we also give the θ' that does. We use '-' if 4.8 does indeed maximize the probability. The root is listed as the sites with different values from the root in Figure 2. Time is in hours (h) and minutes (m). The likelihood value is scaled by multiplying a factor of 10^{20} .

Root	#AC	Likelihood	Time	Best θ	Root	#AC	Likelihood	Time	Best θ
-	218,486,135	8.99	21 h, 13 m	-	8	156,785,599	0.0254	16 h, 48 m	-
15	137,267,493	0.150	17 h, 7 m	4.7	7,15	68,633,748	0.0231	8 h, 12 m	4.6
14	217,678,497	4.75	25 h, 40 m	4.7	14,4	216,989,956	2.86	25 h, 35 m	4.7
14,4,6	216,301,428	0.799	25 h, 52 m	-	14,4,6,5	105,395,284	0.000321	12 h, 6 m	-
14,4,6,1	188,058,258	0.00392	22 h, 44 m	-	14,4,6,1,10	107,461,865	0.0000118	12 h, 37 m	-
18	218,359,460	1.353	25 h, 7 m	4.9	18,2	97,623,432	0.00210	11 h, 20 m	4.9
18,11	167,291,541	0.0702	21 h, 7 m	-	18,11,12	125,468,660	0.01380	16 h, 11 m	4.7
18,11,12,13	41,822,889	0.0000242	4 h, 34 m	4.7	18,16	212,086,076	0.0272	25 h, 16 m	4.9
18,16,17	165,530,601	0.000132	17 h, 15 m	4.9	3	168,748,161	0.255	19 h, 27 m	4.7
3,9	126,561,124	0.0429	15 h, 32 m	4.6					

As shown in Table 2, both the running time and the likelihood value for different root sequences can vary. However, if we take the MLE of θ by only picking just one of the roots, the result is not very different: the range of such estimate is within [4.6, 4.9]. Also, the running time for each root is well correlated with the total number of ACs. Thus, it may be useful to first compute the total number of ACs in order to see whether exact likelihood computation is feasible. There are 2.96×10^9 ACs with all the roots combined, and it takes about 360 hours to compute the full likelihood of the data. This indicates that exact likelihood computation under infinite sites model is still challenging but feasible for moderate sized data. Note that the total number of ACs is slightly smaller than that in [14]. This is due to a different way of handling multiple mutations

along a single tree branch: our method fixes the order of these mutations, while all possible order is considered in [14].

The likelihood curve for the entire data (i.e. the sum of the likelihood for all the 19 roots) is shown in Figure 3. Figure 3 confirms $\theta^* = 4.8$ is the MLE of θ as found by Griffiths and Tavarè [5].

Identifying the ancestral lineage. We can validate one of ancestral inference solutions in [5]. Griffiths and Tavarè infer the likely ancestral lineage by picking the rooted tree with the largest likelihood and choosing the corresponding root as the inferred ancestral lineage. They then concluded that the all-zero lineage (i.e. the root in Figure 2) is the most likely ancestral lineage. From Table 2, we can draw the *same* conclusion, although there are a few other ancestral lineages (e.g. the lineage with exactly one 1 value at site 14) that give smaller but still comparable likelihood values.

Importance sampling methods. Hobolth, et al. [9] compares the likelihood curves generated by three importance sampling methods: Griffiths and Tavarè [4] (GT), Stephens and Donnelly [15] (SD) and their own method (HUW). It appears that they only consider the single phylogeny shown in Figure 2, instead of the unrooted case. The plot of the three likelihood curves is given in Figure 4 in [9]. To give a rough comparison of the three likelihood curves with the exact likelihood curve in Figure 3, we simply compare the maximum likelihood values. the maximum likelihood values GT, SD, and HUW are 7.85, 9.23, and 8.75 respectively, while the exact likelihood is 8.99. The likelihood value is scaled by multiplying a factor of 10^{20} . It can be seen that both SD and HUW appear to be more accurate than GT. Also it is shown in [9] that HUW has smaller variance than SD and SD has smaller variance than GT. Note that more validations are needed to further compare these importance sampling methods.

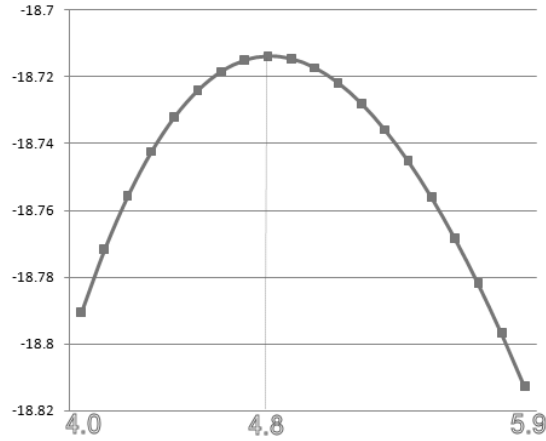


Fig. 3. MLE of θ of the mtDNA data in [18] is 4.8, which matches the result in [5]. The plotted value is the log-likelihood for θ from 4.0 to 5.9, with increment of 0.1.

4 Discussions

In this paper, we show that exact likelihood for haplotypes evolving under the infinite sites model can be computed for many data, while such computation for many such data is previously considered to be infeasible. Our method is *exact* and fully deterministic. Statistical methods, e.g. the importance sampling method implemented in Genetree, on the other hand, usually have (sometimes large) variance in their estimate. Some importance sampling methods may have smaller variance, but the accuracy of the estimated likelihood remains an issue.

Nevertheless, exact computation becomes increasingly difficult when the data size grows. Currently, no known methods can compute exact likelihood for large data (with 100 haplotypes with $\theta = 10$, for example). Statistical methods (e.g. [4, 12, 15, 9]) are much faster and can handle larger datasets than the exact method. Moreover, our results in Section 3 suggest that faster statistical methods can be quite accurate. We also note that program Genetree has more functionalities such as allowing input sequences from subdivided population. So what is the use of the new exact method?

We believe that there are two applications of the exact method.

1. Exact likelihood computation of moderately sized data can be useful. Although slower than statistical methods, exact likelihood can be computed for many datasets on a modern computer. We expect the type of computer on which exact method can run for these data will be accessible to many users, and situation can improve with the continuing fast development of computing technology. Biological data is often noisy, which may not be fully consistent with the model assumed by analysis methods. Moreover, the model itself can be controversial. Stochastic methods, although powerful, can introduce another source of errors; Thus, exact computation eradicates this additional uncertainty introduced by analysis methods. This can ease the concerns on analysis methods and help the study of underlying biological processes.

2. Exact methods can also help to compare and justify existing and new statistical methods. Comparing with exact likelihood is natural and was used in [4] when developing new statistical methods. We believe with more powerful exact methods (such as the method developed here) as the reference, one can compare the different methods and develop new methods by testing on larger simulated and real biological data. Initial comparison of several statistical methods with the exact method is given in Section 3. Our exact method can be useful in a more thorough comparison of these methods. Lack of good exact methods often poses problems when one wants to compare different statistical methods. As an example, there is no known good exact method that can compute the exact coalescent likelihood with recombination unless the data is tiny [13]. This makes it difficult to compare the relative performance of different methods for the coalescent with recombination problem [6]. Thus, it is valuable to develop effective exact methods for the purpose of comparison and validation.

Funding and Acknowledgment This work is supported by National Science Foundation [IIS-0803440]. I am also supported by the Research Foundation of University of Connecticut. I thank Yun S. Song for useful discussions.

References

1. M. Bahlo and R. C. Griffiths: Inference from Gene Trees in a Subdivided Population, *Theoretical Population Biology*, v.57, pages 79-95, 2000.
2. S.N. Ethier and R.C. Griffiths: The Infinitely-Many-Sites Model as a Measure Valued Diffusion, *Annals of Probability*, v.15, pages 515-545, 1987.
3. W. J. Ewens: The sampling theory of selectively neutral alleles, *Theor. Popul. Biol.*, v. 3, pages 87-112, 1972.
4. R. C. Griffiths and S. Tavarè: Simulating Probability Distributions in the Coalescent, *Theor. Popul. Biol.*, v. 46, pages 131-159, 1994.
5. R. C. Griffiths and S. Tavarè: Ancestral inference in population genetics *Statistical Science*, v. 9, pages 307-319, 1994.
6. R. C. Griffiths, P. A. Jenkins and Y. S. Song: Importance Sampling and Two-Locus Model with Subdivided Population Structure, *Adv. Appl. Prob.*, v. 40, pages 473-500, 2008.
7. D. Gusfield: Efficient algorithms for inferring evolutionary history, *Networks*, v. 21, pages 19-28, 1991.
8. J. Hein, M. Schierup and C. Wiuf: *Gene Genealogies, Variation and Evolution: A primer in coalescent theory*, Oxford University Press, 2005.
9. A. Hobolth, M. K. Uyenoyama and C. Wiuf: Importance Sampling for the Infinite Sites Model, *Stat. Appl. Genet. and Mol. Biol.*, v.7, Article 32, 2008.
10. R. Hudson: Generating Samples under the Wright-Fisher neutral model of genetic variation, *Bioinformatics*, v.18, no.2, p.337-338, 2002.
11. J. F. C. Kingman: The coalescent, *Stochast. Process. Appl.*, v. 13, pages 235-248, 1982.
12. M. K. Kuhner, J. Yamato and J. Felsenstein: Estimating effective population size and mutation rate from sequence data using Metropolis-Hastings sampling, *Genetics*, v. 140, pages 1421-1430, 1995.
13. R. Lyngso, Y. S. Song, and J. Hein: Accurate Computation of Likelihoods in the Coalescent with Recombination Via Parsimony, in *Proceedings of Research in Computational Molecular Biology (RECOMB) 2008*, v. 4955, pages 463-477, Springer-Verlag LNCS, Berlin, Germany, 2008.
14. Y.S. Song, R. Lyngsoe and J. Hein: Counting all possible ancestral configurations of sample sequences in population genetics, *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, v.3, pages 239-251, 2006.
15. M. Stephens and P. Donnelly: Inference in molecular population genetics, *J. R. Stat. Soc.*, v. 62, pages 605-655, 2000.
16. S. Tavarè: Ancestral Inference in Population Genetics, in *Lectures on Probability Theory and Statistics*, Springer, pages 1931, 2004.
17. J. Wakeley: *Coalescent Theory: An Introduction*, Roberts and Company Publishers, Greenwood Village, CO., 2008.
18. R.H. Ward, B.L. Frazier, K. Dew and S. Paabo: Extensive Mitochondria Diversity within a Single Amerindian Tribe, *Proc. of the Nat. Academy of Science*, v. 88, pages 8720-8724, 1991.
19. G. A. Watterson: On the number of segregating sites in genetical models without recombination, *Theoretical Population Biology*, v. 7, pages 256-276, 1975.