# A hidden Markov model with binned duration algorithm

S. Winters-Hilt* and Z. Jiang**
Department of Computer Science
University of New Orleans
2000 Lakeshore Drive
New Orleans, LA 70148

* corresponding author: SWH: email: winters@cs.uno.edu; office: 504-280-2407; fax: 504-280-7228
** ZJ is an equally contributing author

**Abstract**

The hidden Markov model with duration (HMMD) is critically important when the distributions on state intervals deviate significantly from the geometric distribution, such as for multimodal distributions and heavy-tailed distributions. Heavy-tailed distributions, in particular, are widespread in describing phenomena across the sciences, where the log-normal, students-T, and Pareto distributions are heavy-tailed distributions that are almost as common as the normal and geometric distributions in descriptions of physical phenomena or man-made phenomena.

The standard hidden Markov model (HMM) constrains state occupancy durations to be geometrically distributed, while HMMD avoids this limitation, but at significant computational expense. We propose a new algorithm - Hidden Markov Model with Binned Duration (HMMBD), whose result shows no loss of accuracy compared to the HMMD decoding performance and a computational expense that only differs from the much simpler and faster HMM decoding by a constant factor.

## I. INTRODUCTION

The Viterbi and Baum-Welch algorithms are the underlying communication, error-coding, and structure-identification algorithms used in cell-phone communications, deep-space satellite communications, voice recognition, and in gene-structure identification [5,6], with growing applications in areas such as image processing

and channel current cheminformatics [3]. The HMMD generalization is important because the standard, HMM-based, Viterbi and Baum-Welch algorithms are critically constrained in their modeling ability to distributions on state intervals that are geometric. This can lead to a significant decoding failure in noisy environments when the state-interval distributions are not geometric (or approximately geometric), as will be shown in what follows. The starkest contrast occurs for multimodal distributions and heavy-tailed distributions.

Heavy-tailed distributions are widespread in describing phenomena across the sciences [7]. The log-normal and Pareto distributions are heavy-tailed distributions that are almost as common as the normal and geometric distributions in descriptions of physical phenomena or man-made phenomena (such as internet packet size). The length distribution for introns, in particular, has very strong support in an extended heavy-tail region, likewise for the length distribution on open reading frames (ORFs) in genomic DNA [3]. In fact, the anomalously long-tailed aspect of the ORF distributions is the key distinguishing feature of this distribution, and has been the key attribute used by biologists to identify likely protein-coding regions in genomic DNA since the early days of (manual) gene structure identification. The length distributions on blockade states in channel current analysis can likewise be strongly skewed (engineered) towards having a heavy-tail, especially if the channel current environment is modulated to obtain an inverted population. Thus, in many signal processing applications, in Bioinformatics and Cheminformatics in particular, use of HMMDs over HMMs is expected to offer significant benefit. Non-geometric duration distributions also occur in many familiar areas, such as the length of spoken words in phone conversation [8], as well as other areas in voice recognition. (For voice recognition, however, the approximations implicit in use of an HMM over an HMMD are manageable, so HMMDs are not used in such application areas.) For the most part, however, the use of HMMs over HMMDs has typically been justified on the grounds that the HMMD was more difficult to implement, and would take far longer to process the same data – neither limitation is true with the latest results we present here.

The original description of an explicit HMMD required computation of order $O(TNN+TND^2)$ [9], where $T$ is the period of observations, $N$ is the number of states, and $D$ is the maximum duration of state transitions to self (where $D$ is typically $> 500$ in gene-structure identification and channel current analysis [3]). This was generally too prohibitive (computationally expensive) in practical operations, and introduced a severe maximum-interval constraint on the interval-distribution model. Improvements via hidden semi-Markov models to computations of order $O(TNN+TND)$ were described in [1,2], where the Viterbi and Baum-Welch (EM) algorithms were implemented, the latter improvement only obtained as of 2003. In these derivations, however, the maximum-interval constraint is still present (comparisons of these methods were subsequently detailed in [10]). Other HMM generalizations include Factorial HMMs [11] and hierarchical HMMs [12]. For the latter, inference computations scaled as $O(T^3)$ in the original description, and have since been improved to $O(T)$ by [13]. Our HMMBD has its computation complexity of $O(TNN+TND^*)$, where $D^*$ (typically $< 50$, and can

be as low as 4 or 5) is the number "bins" used to group up consecutive durations. We begin the description of our method (Sec. II) with a brief description of the hidden semi-Markov model formulation followed by the description of the modifications needed to obtain our HMMBD algorithm in Sec. III. We present experimental results in Section IV, and Discussion and Conclusions in Section V.

## II. The HSMM

In this section we introduce our hidden semi-Markov model (HSMM) formalism, where we use a similar notation to that introduced by Rabiner [5]. An equivalent formulation of the HSMM was introduced in [1] for the Viterbi algorithm and in [2] for Baum-Welch. The formalism introduced here, however, is directly amenable to the HMMBD that will be described in Section III. We begin in Subsection II.A with a description of the Baum-Welch algorithm in the hidden semi-Markov model (HSMM) formalism. This is followed by Subsection II.B with a description of the Viterbi algorithm in the HSMM formalism. See [14] for further details on our HSMM formalism (carrying forward the notation of Rabiner [5], etc.).

### *A. The Baum-Welch algorithm in the explicit HMMD formalism*

Forward and backward formulas:

$$\alpha_t(i,d) = \begin{cases} \check{\alpha}_{t-1}(i)\; b_i(O_t) & \text{if } d = 1 \\ \alpha_{t-1}(i,d-1)\; b_i(O_t) & \text{if } 2 \le d \le D \end{cases} \quad \beta_t(i,d) = \begin{cases} b_i(O_t)\; \check{\beta}_{t+1}(i) & \text{if } d = 1 \\ b_i(O_t)\; \beta_{t+1}(i,d-1) & \text{if } 2 \le d \le D \end{cases} \tag{1}$$

where

$$\hat{\alpha}_t(i) = \sum_{d=1}^{D} \alpha_t(i,d) p_i(d) \quad \check{\alpha}_t(i) = \sum_{j=1,\, j\neq i}^{N} \hat{\alpha}_t(j) a_{ji} \quad \hat{\beta}_t(i) = \sum_{d=1}^{D} \beta_t(i,d) p_i(d) \quad \check{\beta}_t(i) = \sum_{j=1,\, j\neq i}^{N} a_{ij} \hat{\beta}_t(j) \tag{2}$$

And

$$\alpha_t(i,d) p_i(d) = P(O_1 O_2 \cdots O_t, S_i \text{ ends at } t \text{ with duration of } d \,|\lambda)$$

$$\beta_t(i,d) p_i(d) = P(O_t O_{t+1} \cdots O_T, S_i \text{ has a remaining duration of } d \text{ from } t \,|\lambda)$$

Re-estimation formulas:

$$\pi_i^{new} = \frac{\nu_1(i)}{\sum_{j=1}^{N} \nu_1(j)} \quad a_{ij}^{new} = \frac{\sum_{t=1}^{T-1} \mu_t(i,j)}{\sum_{j=1}^{N} \sum_{t=1}^{T-1} \mu_t(i,j)} \quad p_i^{new}(d) = \frac{\sum_{t=1}^{T} \omega(t,i,d)}{\sum_{d=1}^{D} \sum_{t=1}^{T} \omega(t,i,d)} \quad b_i^{new}(k) = \frac{\sum_{t=1 \text{ s.t. } O_t = k}^{T} \nu_t(i)}{\sum_{t=1}^{T} \nu_t(i)} \tag{3}$$

where

$$\omega(t,i,d) = \check{\alpha}_{t-1}(i)\; \beta_t(i,d)\; p(d) \quad \mu_t(i,j) = \hat{\alpha}_t(i) a_{ij} \hat{\beta}_{t+1}(j) \tag{4}$$

$$\nu_t(i) = \begin{cases} \hat{\alpha}_T(i) & \text{if } t = T \\ \nu_{t+1} + \sum_{j\neq i}^{N} (\mu_t(i,j) - \mu_t(j,i)) & \text{if } 1 \le t \le T-1 \end{cases} \tag{5}$$

To avoid underflow problem, we apply a dynamic scaling process to keep the forward and backward variables within a manageable numerical interval. Basically, we keeps two set of emissions $b_i(k)$: one is scaled, the other is not. At every time index, if the numerical values become too small (test $\hat{\alpha}_t(i)$, for example), we make the emission pointer point to the scaled version, otherwise use the unscaled version.

### *B. The Viterbi algorithm in the explicit HMMD formalism*

$$\delta_t(i,d) = \begin{cases} \log b_i(O_t) + \max\limits_{j=1,\, j\neq i}^{N} \{\hat{\delta}_{t-1}(j) + \log a_{ji}\} & \text{if } d = 1 \\ \delta_{t-1}(i, d-1) + \log b_i(O_t) & \text{if } 2 \leq d \leq D \end{cases} \tag{6}$$

where $V_t(i) = \max_{d=1}^{D} \{\delta_t(i,d) + \log p_i(d)\}$

## III. The HMMBD algorithm

The duration distribution of state $i$ consists of rapidly changing probability regions (with small change in duration) and slowly changing probability regions. In the last section all regions share an equal computation resource (represented as $D$ substates of a given state) $-$ this can be very inefficient in practice. In this section, we describe a way to recover computational resources, during the training process, from the slowly changing probability regions. As a result, the computation complexity can be reduced to $O(TNN + TND^*)$, where $D^*$ is the number of "bins" used to represent the final, coarse-grained, probability distribution. A "bin" of a state is a group of substates with consecutive durations. For example, $\alpha(i,d), \alpha(i,d+1), ...\alpha(i,d+\Delta d)$ can be grouped into one bin. The bin size is a measure of the granularity of the evolving length distribution approximation. A fine-granularity is retained in the active regions, perhaps with only one length state per bin, while a coarse-granularity is adopted in weakly changing regions, with possibly hundreds of length states per bin. Starting from this binning idea, for substates in the same bin, a reasonable approximation is applied:

$$\sum_{d=d'}^{d'+\Delta d} \alpha_t(i,d)\; p_i(d) = p_i(\bar{d}) \sum_{d=d'}^{d'+\Delta d} \alpha_t(i,d)$$

where $\bar{d}$ is the duration representative for all substates in that bin. For different bins, the values of $\bar{d}$ and $\Delta d$ may be differnt. From now on whenever they are mentioned, we mean those values associated with its bins.

We begin in Subsection III.A that follows with a description of the Baum-Welch using the HMMBD algorithm This is followed by Subsection III.B with a description of the Viterbi using the HMMBD algorithm.

*A. Baum-Welch using HMMBD algorithm*

First define a variable associated with each bin (which can be calculated recursively):

$$\hat{b}_t(i,n) = \prod_{t-\Delta d}^{t} b_i(O_t)$$

In an actual implementation the scaling procedure is applied on $b_t(O_t)$, so the underflow problem can be avoided. Based on the above approximation, the forward equations in (1) and (2) can be replaced by:

$$\alpha_t(i,n) = \begin{cases} [\alpha_{t-1}(i,n) - \rho_t(i,n) + \check{\alpha}_{t-1}(i)]\ b_i(O_t) & \text{if } n = 1 \\ [\alpha_{t-1}(i,n) - \rho_t(i,n) + \rho_t(i,n-1)]\ b_i(O_t) & \text{if } 1 < n \leq D^* \end{cases} \tag{7}$$

where

$$\hat{\alpha}_t(i) = \sum_{n=1}^{D^*} \alpha_t(i,n)\ p_i(\bar{d}) \qquad\qquad \check{\alpha}_t(i) = \sum_{j=1,\, j\neq i}^{N} \hat{\alpha}_t(j) a_{ji} \tag{8}$$

$$\rho_t(i,n) = Q(i,n).removelast() * \hat{b}_{t-1}(i,n) \qquad Q(i,n).addfirst(\rho_t(i,n-1)) \tag{9}$$

The explanation begins with associating every bin with a queue $Q(i,n)$. The queue's size is equal to the number of substates grouped by this bin. At every time index, the oldest substate: $\alpha(i, d+\Delta d)$ will be shifted out of its current bin and pushed into its next bin, as shown in right part of (9), where $Q(i,n)$ stores the original probability of each substate in that bin when they were pushed in. So when one substate becomes old enough to move to the next bin, its current probability can be recovered by: first move out its original probability, then multiply it by $\hat{b}_{t-1}(i,n)$, as shown in left part of (9). On the other hand, as long as substates stay inside a bin, their invidual computations can be grouped into only one in the $\alpha$ term, as shown in (7).

Similarly, the backward equations in (1) and (2) can be replaced by:

$$\beta_t(i,n) = \begin{cases} b_t(O_t)\ [\beta_{t+1}(i,n) - \rho_t(i,n) + \check{\beta}_{t+1}(i)] & \text{if } n = 1 \\ b_t(O_t)\ [\beta_{t+1}(i,n) - \rho_t(i,n) + \rho_t(i,n-1)] & \text{if } 1 < n \leq D^* \end{cases} \tag{10}$$

where

$$\hat{\beta}_t(i) = \sum_{n=1}^{D^*} \beta_t(i,n)\ p_i(\bar{d}) \qquad\qquad \check{\beta}_t(i) = \sum_{j=1,\, j\neq i}^{N} a_{ij} \hat{\beta}_t(j) \tag{11}$$

$$\rho_t(t,n) = Q(i,n).removelast() * \hat{b}_{t+\Delta d+1}(i,n) \qquad Q(i,n).addfirst(\rho_t(i,n-1)) \tag{12}$$

For re-estimation, $\omega$ term (in Section II) is now computed by:

$$\omega(t,i,\bar{d}) = \check{\alpha}_{t-1}(i)\ \beta_t(i,\bar{d})\ p(\bar{d}) \tag{13}$$

while the other re-estimating formulas remain unchanged.

*B. Viterbi using the HMMBD algorithm*

Similarly the process of Viterbi using the HMMBD algorithm (with computation complexity also $O(TNN + TND^*)$) is:

$$\hat{b}_t(i) = \begin{cases} 0 & \text{if } t = 1 \\ \hat{b}_{t-1}(i) + \log b_i(O_t) & \text{if } 1 < t \leq T \end{cases} \tag{14}$$

$$\zeta_t(i) = \max_{j=1, j\neq i}^{N} \{m_{t-1}(j) + \hat{b}_{t-1}(j) + \log a_{ji}\} \tag{15}$$

$$\zeta_t(i) = \zeta_t(i) - \hat{b}_{t-1}(i) \tag{16}$$

$$\text{if } (B(i,n).getlast() \text{ is too old}) \ \{B(i,n).removelast();\} \tag{17}$$

$$\text{while } (B(i,n).getfirst() < \zeta_t(i)) \ \{B(i,n).removefirst();\} \tag{18}$$

$$B(i,n).addfirst(\zeta_t(i)) \quad 1 \leq n \leq D^* \tag{19}$$

$$t' = t + d(i,n) - 1 \quad 1 \leq n \leq D^* \tag{20}$$

$$\text{if } (B(i,n).getlast() + log(p_i(\bar{d})) > m_{t'}(i)) \ \{m_{t'}(i) = B(i,n).getlast() + log(p_i(\bar{d}));\} \tag{21}$$

The explanation and usage for the above relations are as follows:

- First define the term $\hat{b}_t(i)$ as shown in (14). When a new substate transits (from one of other states) to state $i$ at time $t$, we store the difference between this substate's current path probability and the term $\hat{b}_{t-1}(i)$, as shown in (16) and (19). Then if this substate's probability is needed at any future time $t' > t$, its path probability at time $t'$ can be computed as "the originally stored difference" plus $\hat{b}_{t'}(i)$, as shown in the term $m_{t-1}(j) + \hat{b}_{t-1}(j)$ in (15). The benefit is that the add-computations on $\log b_i(O_t))$ in (6) are saved (in (14)).
- If a substate at $t-1$ has a duration $d-1$, then at time $t$, its duration increased to $d$. So at every time index we check to see if there is a substate too old to stay in the bin-$B(i,n)$, if so, it is be removed from the bin, as shown in (17).
- Before a new substate is pushed into the bin, we first scan the bin top-down and delete all substates whose path probability is less than the new substate. Because those deleted substates will never have a chance to be chosen as the "max" viterbi path, as shown in (18) and (19). As a result, paths inside a bin are always kept sorted, and the "max" one is always the last element of the bin. Thus, in (21) we can directly apply $getlast()$ on each bin to get its "max". The advantage of (18) is that the complexity of the "while" loop is only $O(1)$. (Suppose if more than one comparison is needed on average for every time index, then the bins will soon shrink to empty, a contradiction.)

- The max probability path of $B_t(i,n)$ can be pre-determined at time $t - d(i,n) + 1$, where $d(i,n)$ is the duration length of first substate of $B_t(i,n)$. This is because all substates of the same state $i$ get the same increment of $b_i(O_t)$ at very time index. Detailed steps are shown in (19), (20) and (21).

### *C. Training heuristics on length distribution representation*

A variety of heuristics have been explored for the HMMBD training process. In initial efforts, for simplicity, we used a heuristic where the first $1/1000$ of training on the fine-granularity D-states was performed and used to estimate the length distribution crudely, but enough to estimate some very slowly changing regions (where counts are sufficiently high for this to be trusted), we then conduct the next round of training on another $1/1000$ of the training data, but now with slightly fewer states because a very small amount of granularity in states is introduced (where not all bins simply have a single length state). We then iterate this granularization process, such that by the time 1/100 of the data has been examined the granularity reduction is accomplished, and the remaining 99% of training is performed at the optimized (dynamic mesh) granularity. All sums, recursions, max operations, etc., shift to expressions according to the granularity. In this way we get the best of all possibilities, an unconstrained-D HMMD that dynamically shifts to a strong representation of the length distribution information using only $D^* \ll D$ bins.

The next heuristic considered (and used in the Results below) is based on evaluations of the mean-probability differences between adjacent length-distribution bins, i.e., a '$min\_gap$' cutoff is introduced between the average probabilities of adjacent bins. If adjacent bins have mean bin probabilities that differ by less than '$min\_gap$', then those bins are combined. For monotonically decreasing distributions, with no bin-gaps less than $min\_gap = 0.05$, this results in less than $20$ bins in a solution, and for single-peaked distributions would give rise to less than $2 * (1/0.05) = 40$ bins in a solution. (In the Results we find that we typically obtain reductions to $4$ or $5$ bins, however, well below the bin-limit artifacts indicated.) The '$min\_gap$' heuristic begins with the full length distribution and adiabatically imposes the '$min\_gap$' constraint by incrementally imposing the '$min\_gap$' cutoff. In the experiments described in the Results we start from $min\_gap*(1/10)$ and increment by tenths until the full $min\_gap * (10/10)$ cutoff is imposed. The original formulation of the heuristic followed the structured learning process indicated above, by alternating incremental '$min\_gap$' coarsening of the length distributions with each round of the HMMBD learning process (with possible EM re-estimation of the coarse-grained length distributions). This incremental approach in the batch learning process is then trivially extendable to on-line learning (part of the motivation for its formulation), for use in situations where where slow drift in experimental conditions may result in a slowly changing set of the emission, transition, and length distribution variables (as occurs in channel current analysis [15]). Details about on-line HMMBD learning and its applications will not be discussed further in this paper.

## IV. EXPERIMENTAL RESULTS

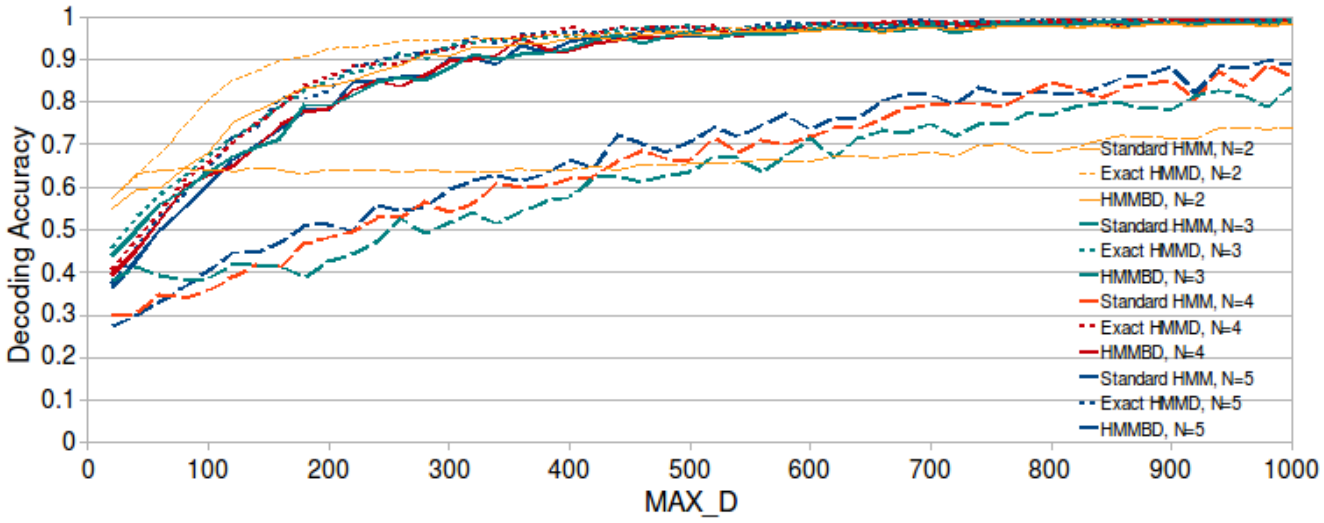


Fig. 1. Decoding Accuracy of Exact HMMD, HMMBD, and Standard HMM. The $N$=2 case is a special degenerate case (when you leave one state, there is only one other state to go to), so its behavior is notably different (the light orange lines). For $N$=3, 4, or 5, the clear decoding improvements of HMMD/HMMBD over HMMs is readily apparent (where long-tailed distributions are used in the data generation described in Sec. III.C). The behavior for $N$=10, 25, and 50 was also examined (not shown) and was found to be very similar to that shown above where the comparisons on the now small test-sets described in III.C could be trusted – i.e., there was good comparison for the lower $MAX\_D$ values, while the higher $N$ higher $MAX\_D$ experiments require larger datasets for true comparative studies and are not pursued further in this study.

For the experiments described here we consider scenarios that occur in nanopore detector channel current analysis [3,4,14,15]. In preliminary explorations three parameterized distributions were examined: geometric, Gaussian, and Poisson. Distributions both segmented and "messy" were also examined. In all cases HMMD and HMMBD performed comparably. In many tests HMMBD, with $min\_gap = 0.05$ coarsening on length distributions, arrived at very few states, often just $D^* = 3$ or 4, with performance still comparable to the exact HMMD (96.6% correct decoding for the HMMBD with $D^* = 3$ compared to 96.8% correct decoding for the exact HMMD). This remarkably improved performance compared to the HMM performance (61% correct) is attributed to the benefit of having a better fit to the "tail" of the distribution (i.e., the tail bin described earlier).

The preliminary results described above motivated the set of tests shown in Figures 1-3. In this set of experiments a much larger dataset is considered: Sequences are generated that are 100,000 samples long, with 100 sequences generated for training, 5 for testing. This data generation is repeated for each specification of $MAX\_D$ or of state number, etc. The number of states considered ranges from 2 to 50 (with only 2-5 shown in the Figures). For the three-state system, the states are described as occupying blockade levels with means at 11, 12, and 13 pA. All states have Gaussian emission with standard deviation 6 pA. All states have as length

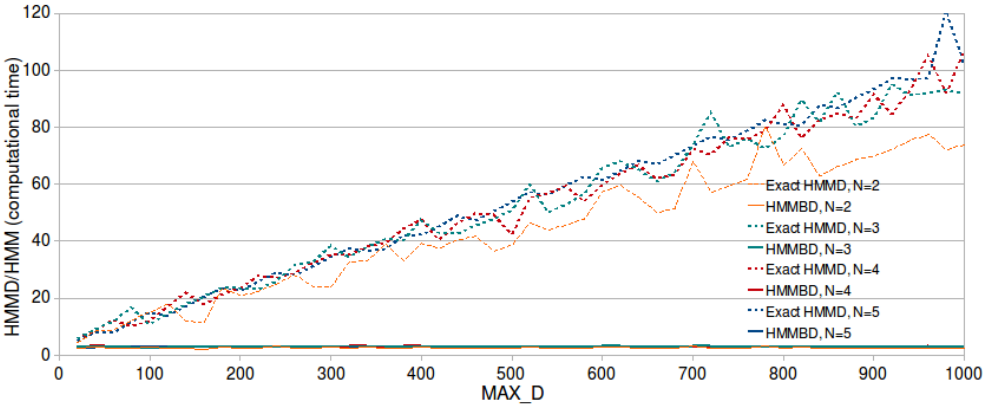


Fig. 2. Viterbi Processing Speed. The computational time required to complete the Viterbi algorithm for the Exact HMMD and HMMBD algorithms, in terms of comparison to the HMM decoding time on the same data. The HMMBD algorithm is shown to not scale with $MAX\_D$, and has asymptote at 2.5 times the computational time of the standard HMM.

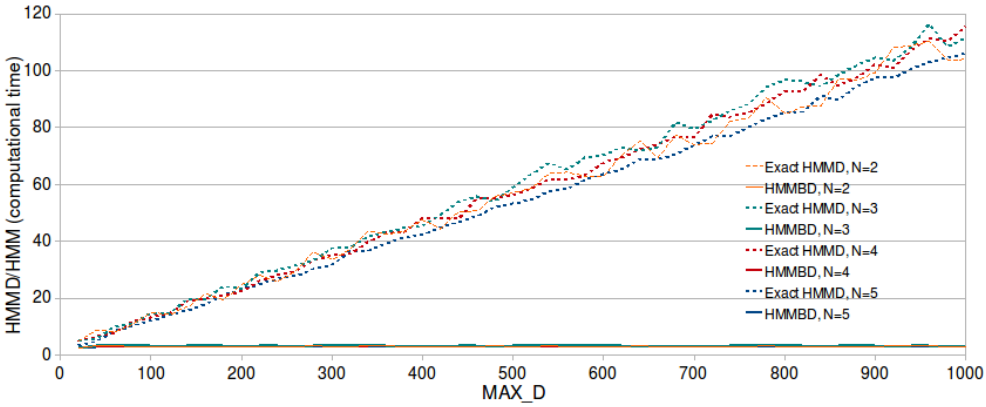


Fig. 3. Baum-Welch (EM) Processing Speed. The computational time required to complete the Baum-Welch algorithm for the Exact HMMD and HMMBD algorithms, in terms of comparison to the HMM decoding time on the same data. The HMMBD algorithm is shown to not scale with $MAX\_D$, and has asymptote at 2.4 times the computational time of the standard HMM.

distribution the student-t distribution (a heavy-tail distribution) with mean=$MAX\_D/10$. For the four-state system we have means set at 11, 12, 13, and 14, generalizing similarly for the other multi-state experiments.

In Fig. 1 we show that the decoding accuracy of the HMMBD algorithm, that is using the simple $min\_gap$ heuristic, is comparable to the much more computationally expensive exact HMMD. Both HMMD and HMMBD methods improve from the HMM's 60% decoding accuracy to approximately 95%, and this holds true for the 2, 3, 4, and 5-state systems shown, and over a large range of $MAX\_D$ (100 to 500). Overall, a 15% to 35% improvement is observed when switching from HMM modeling to HMMD. This vast improvement is critically needed in applications such as channel current experiments, where the decoding directly tracks molecular state and, thereby, directly confers kinetic information [16]. The need for an HMMD is also important in gene

structure efforts as well. For our new method it is critical that the HMMD method be fast, comparable to HMM processing speeds, and this is what is demonstrated for the HMMBD algorithm in Fig.s 2 and 3.

## V. CONCLUSION

In this paper we present the hidden markov model with binned duration algorithm (HMMBD), whose result showes no loss of accuracy compared to the HMMD decoding performance, but with computational expense that only differs from the much simpler and faster HMM decoding by a constant factor. What results (shown in the Sec. IV experiments), is an approximately 100-fold speedup over the fastest known HMMD approaches to Viterbi processing [1] and Baum-Welch processing [2]. Our approach also avoids the need to impose a maximum same-state duration, so improves upon the implementations described in [1,2] in that manner as well. The HMMBD algorithm described here is, thus, an HMM-based signal processing improvement that makes HMMD processing only marginally more computationally expensive than standard HMM processing in a variety of situations where HMMD decoding accuracy can be 95% while HMM decoding accuracy is barely 60%. The situations of greatest benefit involve heavy-tailed duration distributions (since the standard HMM imposes a 'light-tailed' distribution – a geometric distribution). A great variety of heavy-tailed distributions can be found in bioinformatics (e.g., gene structure identification), biology, channel current cheminformatics (e.g., nanopore detection), network packet modeling, and numerous other areas of science or engineering.

## VI. ACKNOWLEDGEMENTS

Funding for this research was provided by an NIH K-22 grant ($5K22LM008794$, SWH PI).

## REFERENCES

[1] P. Ramesh and J.G. Wilpon, "Modeling state durations in hidden Markov models for automatics speech recognition," in Proc. Int. Conf. Acoust., Spech, Signal Process., 1992, pp. 381-384.

[2] S.-Z. Yu and H. Kobayashi, "An efficient forward-backward algorithm for an explicit-duration hidden Markov model," IEEE Signal Process. Lett., vol. 10, no. 1, pp.11-14, Jan. 2003.

[3] S. Winters-Hilt, " Hidden Markov Model Variants and their Application," BMC Bioinformatics 2006, 7 S2: S14.

[4] A. Churbanov and S. Winters-Hilt. Implementing EM and Viterbi algorithms for Hidden Markov Model in linear memory. BMC Bioinformatics 2008, 9:228.

[5] L.R. Rabiner, "A tutorial on hidden Markov models and selected application in speech recognition,"Proc. IEEE, vol. 77, pp. 257-286, Feb. 1989.

[6] A. Krogh, S. Mian, D. Haussler. "A hidden Markov model that finds genes in E. coli DNA," Nucleic Acids Research. 1994;22( 22):4768-78.

[7] E. Limpert, W. A. Stahel, M. Abbt. "Log-normal Distributions across the Sciences: Keys and clues, " BioScience, Vol. 51, No. 5, pg 341-352.

[8] G. Herdan. “The relation between the dictionary distribution and the occurrence distribution of word length and its importance for the study of quantitative linguistics, ” Biometrika 45: 222-228.

[9] J.D. Ferguson, “Variable duration models for speech,” in Symp. Application of Hidden Markov models to Text and Speech, Oct. 1980, pp. 143-179.

[10] M.T. Johnson, “Capacity and complexity of HMM duration Modeling Techniques,” IEEE Sig. Process. Lett, Vol 12, No. 5, pp. 407-410, May 2005.

[11] Z. Ghahramani and M. Jordan. “Factorial hidden Markov models, ” Machine Learning, 29:245-273, 1997.

[12] S. Fine, Y. Singer, and N. Tishby. “The hierarchical Hidden Markov model: Analysis and applications, ” Machine Learning, 32:41, 1998.

[13] K. Murphy and M. Paskin. “Linear time inference in hierarchical HMMs, ” In NIPS, 2001.

[14] Winters-Hilt, S. Nanopore transduction analysis of biotin-streptavidin binding. Submitted to BMC Biotechnology. (preprint at http://www.cs.uno.edu/ winters/)

[15] Winters-Hilt, S., W. Vercoutere, V. S. DeGuzman, D. Deamer, M. Akeson, and D. Haussler, ”Highly Accurate Classification of Watson-Crick Base-Pairs on Termini of Single DNA Molecules,” Biophys. J. Vol. 84, pg 967, 2003.

[16] Winters-Hilt S. The alpha-Hemolysin Nanopore Transduction Detector – single-molecule binding studies and immunological screening of antibodies and aptamers. BMC Bioinf. 8 S7: S12 (2007).

Stephen Winters-Hilt. Ph.D. Computer Science, UCSC, 2003. Ph.D. Physics, U. Wisconsin, 1997. M.S. Applied Physics, California Institute of Technology, 1990. B.S. Electrical Engineering & Physics, California Institute of Technology, 1987. He is currently an Assistant Professor with the Computer Science Department at the University of New Orleans, New Orleans, LA, 70148 and is a Principal Investigator at Children’s Hospital, New Orleans, LA, 70118, where he is Director of the Nanopore Biophysics Lab.

Zuliang Jiang. B.S. Computer Science, Xiamen University, China, 2006. He is currently a Ph.D. Candidate with the Computer Science Department at the University of New Orleans, New Orleans, LA, 70148, USA.