PREMIER - PRobabilistic Error-correction using Markov Inference in Errored Reads

Xin Yin[*], Zhao Song[†], Karin Dorman[*‡] and Aditya Ramamoorthy[†]
[*]Dept. of Statistics, Iowa State University, Ames, IA 50011
[†]Dept. of Electrical & Computer Eng. Iowa State University, Ames, IA 50011
[‡]Dept. of Genetics, Development & Cell Biology Iowa State University, Ames, IA 50011
{xinyin, zhaosong, kdorman, adityar}@iastate.edu

***Abstract*—In this work we present a flexible, probabilistic and reference-free method of error correction for high throughput DNA sequencing data. The key is to exploit the high coverage of sequencing data and model short sequence outputs as independent realizations of a Hidden Markov Model (HMM). We pose the problem of error correction of reads as one of maximum likelihood sequence detection over this HMM. While time and memory considerations rule out an implementation of the optimal Baum-Welch algorithm (for parameter estimation) and the optimal Viterbi algorithm (for error correction), we propose low-complexity approximate versions of both. Specifically, we propose an approximate Viterbi and a sequential decoding based algorithm for the error correction. Our results show that when compared with Reptile, a state-of-the-art error correction method, our methods consistently achieve superior performances on both simulated and real data sets.**

I. Introduction

DNA sequencing is the process of finding the identity and order of nucleotides or bases, adenine (A), guanine (G), cytosine (C) and thymine (T), in DNA molecules. It is used widely in biological and medical research to determine the genomes of diverse organisms ranging from microbes to humans. In recent years the advent of low cost, high throughput DNA sequencing [1] has made it feasible to sequence multiple individuals, even entire populations of organisms. This technological advance may be the key to achieving truly personalized medicine, and several other grand goals in biology.

The summed length of the DNA molecules that need to be sequenced can vary from a few thousand bases to hundreds of gigabases. Sequencing operates by breaking the DNA double helix molecules at random locations and generating incomplete "reads" that start at one end of the resulting fragments and read contiguous bases along one of the two strands. Read lengths are quickly increasing, but vary from thirty base pairs (bp) in the past to over a thousand bp on some platforms [2].

A critical issue in DNA sequencing is the elevated error rate in reads from the current technology [3]. While the throughput rate is high, substitution errors, *e.g.*, base A called as C, and insertion/deletion errors where spurious bases are included or valid bases are left out, are frequent. Errors in reads pose a serious problem for downstream uses of sequence data, including sequence assembly [4], where the full-length sequence is inferred from the short reads, and variant identification [5], for detecting genetic heterogeneity in a population.

This work was supported in part by NSF awards DMS-1120597 and CCF-1149860.

The primary approach for dealing with errors is to capitalize on the high throughput of the sequencing technology. Such an excess of fragments are sequenced that each base in the DNA molecules is covered by multiple reads. However, because the starting location of each read is random, there is no alignment information to indicate which reads cover a particular base in reference-free sequencing scenario. This lack of alignment information makes the problem different from classical error correction [6]. Indeed if alignment information were available, the problem would roughly reduce to the decoding of a repetition code.

Error correction of noisy reads has received significant attention in the bioinformatics community in recent years [7]. We briefly review the methods most closely related to our proposed method. Many methods begin by counting the occurrence of all $k$mers in the reads where a $k$mer is defined as a substring of length k. Euler [8] corrects a read via the smallest set of corrections that ensures that all $k$mers in the read have high occurrences. Hammer [9] identifies cliques by linking similar $k$mers, then corrects all members to the clique's consensus $k$mer. FreClu [10] corrects full-length reads if it finds a significantly more frequent read differing at just one position. Quake [11] iteratively corrects bases by maximizing a posterior probability of the true sequence given the observed read until all $k$mers have high occurrences. Two other methods use a probability model to correct a read [12] or $k$mer [13] to the most likely true sequence. In all these methods the focus must turn to $k$mers when the read length is long to guarantee sufficient repetition to distinguish error and true bases. So as read lengths increase, even read-based methods must become $k$mer-based. All existing $k$mer-based methods ignore the fact that $k$mers are dependently read as contiguous substrings within reads. Moreover, while some allow arbitrarily complex error models, either all error parameters must be provided *a priori* or the parameter estimation procedure is *ad hoc*.

The work of [14], modeled a genome as the output of a discrete memoryless source and determined the coverage levels required to guarantee correct sequence assembly under a noiseless read process. Approaches based on statistical modeling of the sequencing process have been used [15], [16] for basecalling, but not for error correction of reads.

Main contributions. In this work we address the problem of correcting errors in noisy reads from a signal processing and error control coding viewpoint. We consider reads from the Illumina DNA sequencer that is known to exhibit substitution errors (but essentially no insertion/deletion errors) [17]. We demonstrate that Illumina reads can be modeled as symbols emitted from a Hidden Markov Model (HMM). To overcome the unmanageably large state space, we use constraints and penalties to estimate the HMM parameters. Given the parameters of this HMM, we pose the problem of error correction of reads as a maximum likelihood sequence detection problem. While time and memory considerations rule out an implementation of the optimal Baum-Welch algorithm (for parameter estimation) and the optimal Viterbi algorithm (for error correction), we propose low-complexity approximate versions of both. This approach is successful in identifying many errors. In addition we propose a sequential decoding [18] algorithm that achieves even better performance. Our results on *real*, publicly available sequencing data for the *E. coli* genome demonstrate a 10% relative improvement in error correction rates over a current state of the art technique.

II. Problem Formulation

Let $\mathcal{G}$ denote the genome that needs to be sequenced. It is a 4-ary sequence of length $|\mathcal{G}|$, where each letter is in $\Omega = \{A, C, G, T\}$; we call these letters bases or nucleotides. Sequencing operates by breaking the genome into fragments, from which length L reads are made. The sequencer has access to multiple copies of $\mathcal{G}$ and produces up to billions of reads. The starting point of the fragment within $\mathcal{G}$ is random and unknown. The sequencer processes the fragment and outputs a read $(\boldsymbol{x}, \boldsymbol{y})$, where $\boldsymbol{x}$ is the sequencer's best guess of L bases in the fragment and $\boldsymbol{y}$ are the corresponding quality scores; the quality scores are discrete measures of confidence in the base calls $\boldsymbol{x}$. A given base location will typically be covered by multiple reads. Let N denote the total number of reads obtained in this process. The coverage level is defined to be $NL/|\mathcal{G}|$.

A. HMM modeling

We model the sequencer as a Hidden Markov Model (HMM); each read is an independent realization of the HMM. Given $\boldsymbol{s}$, the unknown true read of length L, we define $\boldsymbol{s}[i]$ as the i-th character of $\boldsymbol{s}$ and $\boldsymbol{s}[i \dots j]$ as the substring from position i to j (both $\boldsymbol{s}[i]$ and $\boldsymbol{s}[j]$ are included). Let $\boldsymbol{s}_t = \boldsymbol{s}[t-k+1 \dots t]$ be the t-th *state*. In the discussion below we will also refer to the states as $k$mers. Similarly, $\boldsymbol{x}_t = \boldsymbol{x}[t-k+1 \dots t]$ will be referred to as the tth observed $k$mer and $\boldsymbol{y}_t = \boldsymbol{y}[t-k+1 \dots t]$ will denote the corresponding quality scores. We model the sequencer as transitioning between the states $\boldsymbol{s}_{t-1}$ to $\boldsymbol{s}_t$. On the tth transition it emits the output $(\boldsymbol{x}[t], \boldsymbol{y}[t])$.

To specify the model completely, we need to define:

- State space $\mathcal{K}$, where $|\mathcal{K}| \le 4^k$.
- Transition distribution $p(\boldsymbol{s}[t+1]|\boldsymbol{s}_t)$, where
$$\sum_{\beta \in \Omega} p(\beta|\boldsymbol{\alpha}) = 1, \quad \forall \boldsymbol{\alpha} \in \mathcal{K}.$$
- The *d-neighborhood* of observed $k$mer $\boldsymbol{x}_t$ as
$$\mathcal{N}^d(\boldsymbol{x}_t) = \{\boldsymbol{w} : \boldsymbol{w} \in \mathcal{K} \text{ and } D(\boldsymbol{x}_t, \boldsymbol{w}) \le d\}, \quad (1)$$
where $D(\cdot, \cdot)$ is the Hamming distance function.
- Emission distribution $f_t(\boldsymbol{x}[t], \boldsymbol{y}[t] \mid \boldsymbol{s}_t)$ with
$$f_t(\boldsymbol{x}[t], \boldsymbol{y}[t] \mid \boldsymbol{s}_t) = q_t(\boldsymbol{y}[t] \mid \boldsymbol{x}[t], \boldsymbol{s}_t) g_t(\boldsymbol{x}[t] \mid \boldsymbol{s}_t), \quad (2)$$
where we assume the following simple forms.
$$q_t(\boldsymbol{y}[t] \mid \boldsymbol{x}[t], \boldsymbol{s}_t) = \begin{cases} q_{t0}(\boldsymbol{y}[t]) & \boldsymbol{x}[t] = \boldsymbol{s}[t], \\ q_{t1}(\boldsymbol{y}[t]) & \boldsymbol{x}[t] \ne \boldsymbol{s}[t], \text{ and} \end{cases}$$
$$g_t(\boldsymbol{x}[t] \mid \boldsymbol{s}_t) = \mathbb{1}\{\boldsymbol{s}_t \in \mathcal{N}^d(\boldsymbol{x}_t)\} g_t(\boldsymbol{x}[t] \mid \boldsymbol{s}[t]), \quad (3)$$
$$\text{with} \sum_{\beta \in \Omega} g_t(\beta \mid \beta') = 1, \quad \forall \beta' \in \Omega.$$

Our modeling philosophy is guided by the following considerations. It is well recognized that nucleotides in genomes display strong local dependence, and Markov models, like the one we use for $\boldsymbol{s}$, have long been used to model this dependence [19]. Like most error correction methods, we start with a simple error model. Both $g_t(\beta \mid \beta')$, the probability of (mis)reading base β' as β, and $q_{tj}(q), j = 0, 1$, the quality score probability mass functions, depend on position t. We expect that $q_{t0}(q)$ is shifted right of $q_{t1}(q)$, for all t, because the sequencer should assign higher quality scores to error free bases.

The choice of $k$mer length k and state space $\mathcal{K}$ are guided by several considerations. In principle, given k, one could choose all possible 4^k states as $\mathcal{K}$, but even for moderate k (around 15), such $\mathcal{K}$ is too big. Thus, for a given set of reads, we restrict $\mathcal{K}$ to contain only observed $k$mers. Even though $\mathcal{K}$ includes erroneous $k$mers, we hope to identify them during estimation of the HMM. The choice of k depends on two conflicting requirements. On the one hand, we want an accurate model. If k is too small, say $k = 3$, then each $k$mer, say ATC, will exist in several locations in the original $\mathcal{G}$. Our model will tend to "correct" uncommon downstream bases to common downstream bases. For example, if $ATCG$ occurs twice and $ATCT$ occurs once, then true read $ATCT$ may be erroneously corrected to $ATCG$. On the other hand, very large k (though it cannot exceed the read length L), may lead to decreased $k$mer coverage and eventually an overparameterized, inestimable model. Thus, there is an ideal value of k that achieves uniqueness and an estimable model. In our experiments, we choose k to optimize performance.

To reduce computational complexity, we further constrain the emission distribution. As errors are relatively rare, we only allow $k$mers within a small Hamming distance of an observed $k$mer $\boldsymbol{x}_t$ to have non-zero emission distributions. Notice (3) reduces the overall number of parameters since $g_t(\boldsymbol{x}[t] \mid \boldsymbol{s}_t) \equiv 0$ if $\boldsymbol{s}_t \notin \mathcal{N}^d(\boldsymbol{x}_t)$.

The HMM is fit to the read data using the iterative Expectation-Maximization (EM) algorithm (Baum-Welch). Subsume all model parameters into vector $\boldsymbol{\theta}$. The EM locally maximizes the likelihood and produces parameter estimate $\hat{\boldsymbol{\theta}}$. The initial parameters $\boldsymbol{\theta}^{(0)}$ for the EM are computed as follows. For transition probabilities, we count the occurrence, $n(\boldsymbol{\alpha}, \beta)$, of $\boldsymbol{\alpha}$ followed by β in all reads and set,
$$p^{(0)}(\beta|\boldsymbol{\alpha}) = \frac{n(\boldsymbol{\alpha}, \beta)}{\sum_{\beta' \in \Omega} n(\boldsymbol{\alpha}, \beta')}, \boldsymbol{\alpha} \in \mathcal{K}, \beta \in \Omega. \quad (4)$$

As for the emission part, we initialize

$$\begin{aligned} q_{tj}^{(0)}(q) &= 1/Q_{max}, \quad q \in \{1, \ldots, Q_{\max}\} \\ g_t^{(0)}(\beta|\beta') &= 1/4, \qquad \beta, \beta' \in \Omega, \end{aligned} \tag{5}$$

where $Q_{\max}$ is the maximum quality score the sequencer can produce, $j \in \{0, 1\}$ indicates presence of an error, and $t \in \{k, k+1, \ldots, L\}$ is read position. We tried multiple random initializations and found the EM insensitive to choice of $\boldsymbol{\theta}^{(0)}$.

B. Penalized Estimation

While the HMM reasonably captures the local dependence present in genomic sequences and the error characteristics of modern sequencers, it fails to recapitulate the finite genome length. To impose our certainty that the vast majority of $k$mers are unique, we use the approximate ℓ_0 penalty proposed in [20],

$$J(\boldsymbol{\theta}) = \sum_{\boldsymbol{\alpha} \in \mathcal{K}, \beta \in \Omega} \frac{\ln(1 + p(\beta|\boldsymbol{\alpha})/\gamma)}{\ln(1 + 1/\gamma)} \tag{6}$$

that penalizes small transition probabilities and drives them to zero. Given the set of observed reads $\mathcal{R}$, the EM can be adapted to maximize the penalized log-likelihood $[l(\mathcal{R}|\boldsymbol{\theta}) - \lambda J(\boldsymbol{\theta})]$ where λ and γ are user specified tuning parameters. In general, increasing λ or decreasing γ strengthens the penalty. Desirable values for λ and γ could be determined by imposing a level of sparsity consistent with the prior knowledge of the genome length. In the experiments reported here, we seek a strong penalty to push small transition probabilities to zero and eventually eliminate $k$mers with suspiciously low coverage. Therefore, we choose $\gamma = 10^{-4}$ and vary λ over the rough grid $\{100, 150, 200, 250, 300\}$ to optimize performance.

III. Error Correction Algorithm

Given a fitted HMM for the data, we now discuss the actual error correction algorithm. One naturally turns to the Viterbi algorithm to estimate the maximum likelihood "true read" $\boldsymbol{s}$, given the pair $(\boldsymbol{x}, \boldsymbol{y})$. However, the size of the state space $\mathcal{K}$, even for modest k and relatively small genome size $|\mathcal{G}|$, is formidable and prevents exact Baum-Welch and Viterbi algorithms. Accordingly, we use an approximate Viterbi-like decoding and sequential decoding as discussed below.

A. An approximate Viterbi Algorithm

For computational reasons, the Viterbi is limited, like the HMM, to only consider true sequences, $\boldsymbol{s}$, constrained by our assumptions on the emission distribution. While it propagates likelihoods of survivor paths, if a survivor path contains a state $\boldsymbol{s}_t$ differing at more than d locations from $\boldsymbol{x}_t$, then state $\boldsymbol{s}_t$ is deemed implausible and the survivor path is not extended. The same restriction is applied during the Baum-Welch parameter estimation described in Section II-A. We call this decoding method A-Viterbi.

B. Sequential Decoding on HMM with Fano Algorithm

Sequential decoding was proposed as a way to decode convolutional codes (prior to the optimal Viterbi algorithm) [18], [21]. For codes with high constraint lengths, it serves as a good low-complexity alternative to the Viterbi algorithm. In our work we adapt the Fano algorithm for determining the maximum likelihood state sequence in the HMM. Our discussion here is based on the description in [22] (see Fig. 1). In the Fano algorithm at any given stage there is only one active path, where a path is defined to be a sequence of states $\boldsymbol{s}_1, \ldots, \boldsymbol{s}_t$ that have a non-zero probability of occurrence. Let the path labels of the predecessor path, the current path, and the successor path be $\boldsymbol{\nu}_p$, $\boldsymbol{\nu}_c$, and $\boldsymbol{\nu}_s$. The corresponding Fano metrics are denoted as M_p, M_c, and M_s.

Fig. 1 Sequential Decoding with Fano Algorithm on HMM

```
Input: kmers obtained from the read sequence, step size $\Delta$, bias $B$, parameters for HMM
Output: corrected read (or path $\boldsymbol{\nu}^*$) and Fano metric $M^*$
  Initialization: threshold $T = 0$, $\boldsymbol{\nu}_p = dummy$, $M_p = -\infty$, $\boldsymbol{\nu}_c =$ kmer in the first stage, $M_c = 0$, stage $t = 1$.
 1: Choose the successor path which has the largest Fano metric based on the transition probability of $\boldsymbol{\nu}_c$ and emission probability of the $(t+k)$th base. Denote this path label as $\boldsymbol{\nu}_s$ and corresponding Fano metric as $M_s$.
 2: if $M_s \geq T$ then
 3:     Move one base forward and update
        $\boldsymbol{\nu}_p = \boldsymbol{\nu}_c$, $M_p = M_c$; $\boldsymbol{\nu}_c = \boldsymbol{\nu}_s$, $M_c = M_s$; set $t \leftarrow t+1$
 4:     if $t = L - k + 1$ then
 5:         Stop algorithm and output $\boldsymbol{\nu}^*$, $M^* = M_c$
 6:     else
 7:         if $M_p < T + \Delta$ then
 8:             Tighten threshold $T$ such that $T \leq M_c < T + \Delta$. Go to step 1.
 9:         end if
10:         Go to step 1.
11:     end if
12: else
13:     if $M_p \geq T$ then
14:         Move one base back and update
            $\boldsymbol{\nu}_s = \boldsymbol{\nu}_c$, $M_s = M_c$; $\boldsymbol{\nu}_c = \boldsymbol{\nu}_p$, $M_c = M_p$; $t \leftarrow t - 1$ and re-compute $M_p$ and $\boldsymbol{\nu}_p$.
15:         Attempt to find the non-visited successor path of $\boldsymbol{\nu}_c$ which has the largest Fano metric. Denote this path as $\boldsymbol{\nu}_t$ and its Fano metric as $M_t$.
16:         if $\boldsymbol{\nu}_t$ is empty then
17:             Go to Step 13.
18:         else
19:             Update $\boldsymbol{\nu}_s = \boldsymbol{\nu}_t$, $M_s = M_t$ and go to Step 2.
20:         end if
21:     else
22:         Lower threshold as $T \leftarrow T - \Delta$ and go to Step 1.
23:     end if
24: end if
```

A given candidate successor path $\boldsymbol{\nu}_s$, corresponds to appending a new state to $\boldsymbol{\nu}_c$ such that the new state has a positive probability of being reached from the last state of $\boldsymbol{\nu}_c$. The probability of choosing $\boldsymbol{\nu}_s$ as the successor path can be computed from the transition distribution of the HMM; we denote it as $a(\boldsymbol{\nu}_c, \boldsymbol{\nu}_s)$ below. Likewise the emission distribution specifies the probability of the emitted base and quality score corresponding to this transition; this is denoted by ξ_s below (to avoid complicated notation). The Fano metrics are updated as follows.

$$M_s = M_c + \log_2 \left[a(\boldsymbol{\nu}_c, \boldsymbol{\nu}_s)\right] + \log_2(\xi_s) + B. \tag{7}$$

Here B represents the bias whose value is chosen with the purpose that the Fano metric will keep increasing as long as we are on the correct path [22].

IV. Experimental Results

We compared the performance of the A-Viterbi and Fano decoding algorithms to Reptile on one simulated and one real dataset (Table I). Reptile is a top-performing method in a recent survey of error correction methods [7]. All the results we present here assume we know the first true $k$mer. Since we use simulation or resequencing experiments of known genomes, we can reliably infer this information. In practice, a known primer sequence is often attached to both ends of the DNA fragment being sequenced, so the assumption is not restrictive. All methods include model complexity parameters, such as k and d, that can be difficult to choose when the true sequence is unknown. In our experiments, with the true sequence available, we can tune these parameters for optimal performance. We emphasize that we have tuned *all* methods to achieve their respective best performance.

For each dataset, maximum likelihood estimates of the HMM parameters were estimated using the penalized likelihood ($\gamma = 0.0001, \lambda = 250$) for various $k$mer lengths, $k = 13, 14, 15$, and maximum Hamming distance, $d = 4$. Then, A-Viterbi and Fano were used to perform the decoding at each chosen k. The A-Viterbi algorithm was run using the same d used to estimate the HMM parameters. For the Fano algorithm, parameter B in Eq. (7) was set to be 2 and 10 for the simulated and real dataset, respectively; we tried $\Delta = \{0.5, 1.0\}$ and fix $\Delta = 0.5$ in Tables II and III since the error correction performances are close to each other with respect to Δ. Note, the Fano algorithm places no constraints on the $k$mer Hamming distances.

Reptile uses a "tile" formed by concatenating two $k$mers of length k with overlap of length $k - step$. Corrections are made if the observed tile is uncommon and there is a substantially more common tile in the neighborhood of the observed tile. The tile neighborhood is formed by allowing up to d errors in each $k$mer. Tile counts are computed from high quality reads only. We chose the best parameters $(k, step)$ using a grid-search over $7 \leq step \leq k \leq 12$. Given $(k, step)$, thresholds for error correction decisions were automatically selected, following the instructions in the software manual, except taking half the recommended *T_expGoodCnt* and leaving *MaxBadQPerKmer* at the default 4. The maximum errors allowed per $k$mer was set to $d = 4$. All remaining parameters were left at their defaults.

Let e be the total number of ground truth errors in the sequencing reads excluding those in the first $k$mer (or tile for Reptile), ce be the number of correctly recovered bases with the set of ground truth errors and fa be the number of originally error-free bases that have been incorrectly changed by the algorithm. The probability of error correction is defined as $\zeta \triangleq ce/e$ and gain is defined as $\eta \triangleq (ce - fa)/e$, measuring the effective number of errors removed from the dataset [23]. The runtime of each method is denoted ω, measured in seconds. Timing for A-Viterbi and Fano is reported as seconds to run the EM plus seconds to run A-Viterbi or Fano.

TABLE I
Benchmark sequencing datasets

Dataset	Genome length	Read length (bp)	Number of reads	Coverage	Error rate (%)
D1	250000	36	1000000	144.0x	1.23
D2	500000	36	2132026	153.5x	0.53

TABLE II
Error Correction Results for D1

	$k^\dagger$	ce	ζ	fa	η
Fano	13	430391	0.9981	100	0.9979
	14	428324	0.9993	21	0.9993
	15	**425442**	**0.9996**	**6**	**0.9996**
A-Viterbi	13	430384	0.9967	227	0.9962
	14	427839	0.9978	102	0.9975
	15	**424881**	**0.998**	**75**	**0.9979**
Reptile	(8, 8)	355395	0.8423	3900	0.8331
	(9, 9)	**405971**	**0.9848**	**678**	**0.9832**
	(10, 10)	314306	0.7867	708	0.7849

† : For Reptile, this column is reported as $(k, step)$.

A. D1: Simulated Dataset

To create the simulated dataset, one million reads were generated by randomly sampling 36bp sequences from a 250Kbp region (1000Kbp — 1250Kbp) of the *E. coli* genome (Accession NC.000913). From the real dataset (see IV-B), we estimated empirical distributions of quality scores given read positions, and used these to generate quality scores for every position of each simulated read. We assumed the simulated quality scores indicated the true error probabilities and replaced the true base β_t at position t with error base $\beta'_t \neq \beta_t$ with probability $\left(10^{-q_t/10}/3\right)$ where q_t is the quality score.

Table II shows the error correction results for various choices of $k$mer length, k, or $(k, step)$ for Reptile. In bold, we show the best performance for each method, as measured by the gain metric η, along with the results for a few additional, nearby settings. Both Fano and A-Viterbi outperform Reptile by achieving higher error correction probabilities while having a lower false alarm probability. Overall, the Fano algorithm is the best performer at $k = 15$ and $\Delta = 0.5$.

The HMM-based approaches and Reptile exhibit best performance at different values of k, but Reptile is much more sensitive to this choice. In a reference-free error correction setup, when no explicit ground truth is available to guide choice of k, the HMM-based approaches have the advantage of yielding robust performance over a wider range of k. Quake [11] recommends choosing k such that $2|\mathcal{G}|/4^k \approx 0.01$, which suggests $k = 13$ in this case. While Fano and A-Viterbi are not at their peak performance for this choice of k, their performance is near-optimal. In contrast, the Reptile authors recommend $k = \log_4 |\mathcal{G}| \approx 9$, which indeed works well.

B. D2: Real Experimental Dataset

To test the performance of our model on a real Illumina dataset, we used the data of an *E. coli* resequencing experiment (Accession SRX000429). Knowing the reference genome allows us to identify the "ground truth" errors as long as we can identify the position of the read in the reference genome. To align the reads to the reference genome, we used the Burrows-Wheeler Aligner [24] with default parameters. We selected all reads with a unique match to a 500Kbp region (1000Kbp —

TABLE III
ERROR CORRECTION RESULTS FOR D2

	k	ce	ζ	fa	η	ω
Fano	13	366201	0.9666	10776	0.9381	624+85
	14	364530	0.9724	6213	0.9559	438+63
	15	**360162**	**0.9726**	**4768**	**0.9597**	**331+75**
A-Viterbi	13	356358	0.9406	6116	0.9245	624+40
	14	**351649**	**0.9172**	**3862**	**0.9278**	**438+33**
	15	346867	0.9367	3329	0.9277	331+39
Reptile	(8, 8)	283286	0.7771	22320	0.7159	159
	(9, 9)	**322838**	**0.9123**	**12755**	**0.8763**	**257**
	(10, 10)	241216	0.7076	5263	0.6921	390

1500Kbp) on the reference genome and tallied the true errors as mismatches between the selected reads and the reference sequence. There are cases of erasure where the nucleotide is recorded as "N" for "not determined" in the reads. For Reptile, all N bases were replaced by A, but were left intact for A-Viterbi and Fano, both of which can directly correct base N.

The error correction results on dataset D2 are summarized in Table III. The superiority of the Fano algorithm is accentuated in this real data. We presume that Fano outperforms A-Viterbi by exploring parts of the 4-ary tree ruled out by A-Viterbi due to the neighborhood constraint (cf. Section III-A). The Quake-recommended k for this dataset is 14; Reptile recommends 9.

C. Parameter Selection and Sensitivity

While not as sensitive as Reptile to choice of k, the HMM-based methods are sensitive to other parameter choices. We mention only the strongest effects here as we have not yet completed a thorough analysis. Most strikingly, A-Viterbi achieved gain of only 0.89 on the real dataset D2 with $\gamma = 0.001$ and $\lambda = 500$. The Fano algorithm is most sensitive to the bias B. For the real data, Fano required large B to avoid excessive backtracking.

Since we had roughly optimized the A-Viterbi and Fano algorithms over λ, γ, Δ, and B, we attempted to optimize Reptile over two of its parameters. Through much experimentation, we could improve Reptile performance to $\eta = 0.9577$ when *T_expGoodCnt* = 28 and *T_card* = 13 on the real dataset, which is very close yet marginally inferior, to the best Fano performance. However, neither HMM-based method was allowed quite the same diligent exploration of its parameter space. It is clearly imperative that we develop methods to choose λ, γ, Δ, and B without reference to a test genome, so all methods can be compared on equal footing. Our current set of experiments leads us to conclude that HMM-based methods are substantially easier to tune, less sensitive to parameter settings, and more flexible when it comes to adopting more realistic emission distributions.

V. CONCLUSION

We establish the HMM for the problem of noisy DNA read correction and develop the approximate Viterbi algorithm and Fano algorithm to execute the error correction. Based on our test results for both simulated and real data, the proposed algorithms often outperform another state of the art method in this field. Our future work will include development of systematic procedures for choosing the parameters of the algorithms, an investigation of more complex, context dependent error emission distributions and an exhaustive comparison with respect to competing methods.

REFERENCES

[1] M. L. Metzker, "Sequencing technologies - the next generation." *Nat Rev Genet*, vol. 11, pp. 31–46, 2010.

[2] M. A. Quail, M. Smith, P. Coupland, T. D. Otto, S. R. Harris, T. R. Connor, A. Bertoni, H. P. Swerdlow, and Y. Gu, "A tale of three next generation sequencing platforms: comparison of ion torrent, pacific biosciences and illumina miseq sequencers." *BMC Genomics*, vol. 13, p. 341, 2012.

[3] E. Y. Chan, "Next-generation sequencing methods: impact of sequencing accuracy on snp discovery." *Methods Mol Biol*, vol. 578, pp. 95–111, 2009.

[4] S. L. Salzberg, A. M. Phillippy, A. Zimin, D. Puiu, T. Magoc, S. Koren, T. J. Treangen, M. C. Schatz, A. L. Delcher, M. Roberts, G. Marais, M. Pop, and J. A. Yorke, "Gage: A critical evaluation of genome assemblies and assembly algorithms." *Genome Res*, vol. 22, pp. 557–567, 2012.

[5] A. Wilm, P. P. K. Aw, D. Bertrand, G. H. T. Yeo, S. H. Ong, C. H. Wong, C. C. Khor, R. Petric, M. L. Hibberd, and N. Nagarajan, "Lofreq: a sequence-quality aware, ultra-sensitive variant caller for uncovering cell-population heterogeneity from high-throughput sequencing datasets." *Nucleic Acids Res*, vol. 40, pp. 11 189–11 201, 2012.

[6] S. Lin and D. J. Costello, *Error Control Coding, 2nd Ed.* Prentice Hall, 2004.

[7] X. Yang, S. P. Chockalingam, and S. Aluru, "A survey of error-correction methods for next-generation sequencing." *Brief Bioinform*, 2012.

[8] M. J. Chaisson, D. Brinza, and P. A. Pevzner, "De novo fragment assembly with short mate-paired reads: Does the read length matter?" *Genome Res*, vol. 19, pp. 336–346, 2009.

[9] P. Medvedev, E. Scott, B. Kakaradov, and P. Pevzner, "Error correction of high-throughput sequencing datasets with non-uniform coverage." *Bioinformatics*, vol. 27, pp. i137–i141, 2011.

[10] W. Qu, S.-I. Hashimoto, and S. Morishita, "Efficient frequency-based de novo short-read clustering for error trimming in next-generation sequencing." *Genome Res*, vol. 19, pp. 1309–1315, 2009.

[11] D. R. Kelley, M. C. Schatz, and S. L. Salzberg, "Quake: quality-aware detection and correction of sequencing errors." *Genome Biol*, vol. 11, p. R116, 2010.

[12] E. Wijaya, M. C. Frith, Y. Suzuki, and P. Horton, "Recount: expectation maximization based error correction tool for next generation sequencing data." *Genome Informatics*, vol. 23, pp. 189–201, 2009.

[13] X. Yang, S. Aluru, and K. S. Dorman, "Repeat-aware modeling and correction of short read errors." *BMC Bioinformatics*, vol. 12 Suppl 1, p. S52, 2011.

[14] A. Motahari, G. Bresler, and D. Tse, "Information theory for dna sequencing: Part i: A basic model," in *IEEE International Symposium on Information Theory (ISIT)*, July 2012, pp. 2741 –2745.

[15] S. Das and H. Vikalo, "Onlinecall: fast online parameter estimation and base calling for illumina's next-generation sequencing," *Bioinformatics*, vol. 28, pp. 1677–1683, 2012.

[16] T. Wu and H. Vikalo, "Joint parameter estimation and base-calling for pyrosequencing systems," *IEEE Transactions on Signal Processing*, vol. 60, pp. 4376 –4386, Aug. 2012.

[17] C. Ledergerber and C. Dessimoz, "Base-calling for next-generation sequencing platforms." *Brief Bioinform*, vol. 12, pp. 489–497, 2011.

[18] R. M. Fano, "A heuristic discussion of probabilistic decoding," *IEEE Trans. Inf. Theory*, vol. 9, pp. 64–74, 1963.

[19] E. Picardi and G. Pesole, "Computational methods for ab initio and comparative gene finding," vol. 609, 2010.

[20] D. H. Alexander and K. Lange, "Enhancements to the ADMIXTURE algorithm for individual ancestry estimation," *BMC Bioinformatics*, vol. 12, p. 246, 2011.

[21] J. M. Wozencraft and B. Reiffen, *Sequential Decoding.* MIT Press, 1961.

[22] Y. Han and P. N. Chen, "Sequential decoding of convolutional codes," *Encyclopedia of Telecommunications (Editor: John Proakis), Wiley*, pp. 2140–2164, 2002.

[23] X. Yang, K. S. Dorman, and S. Aluru, "Reptile: Representative tiling for short read error correction," *Bioinformatics*, vol. 26, pp. 2526–2533, 2010.

[24] H. Li and R. Durbin, "Fast and accurate long-read alignment with burrows–wheeler transform," *Bioinformatics*, vol. 26, pp. 589–595, 2010.