



ECHO: A reference-free short-read error correction algorithm

Wei-Chun Kao, Andrew H. Chan and Yun S. Song

Genome Res. published online April 11, 2011

Access the most recent version at doi:[10.1101/gr.111351.110](https://doi.org/10.1101/gr.111351.110)

P<P Published online April 11, 2011 in advance of the print journal.

Accepted Manuscript Peer-reviewed and accepted for publication but not copyedited or typeset; accepted manuscript is likely to differ from the final, published version.

License

Email Alerting Service Receive free email alerts when new articles cite this article - sign up in the box at the top right corner of the article or [click here](#).



To subscribe to *Genome Research* go to:
<https://genome.cshlp.org/subscriptions>

Copyright © 2011, Cold Spring Harbor Laboratory Press

ECHO: A reference-free short-read error correction algorithm

Wei-Chun Kao¹, Andrew H. Chan¹, and Yun S. Song^{1,2}

¹ Computer Science Division, University of California, Berkeley, CA, USA

² Department of Statistics, University of California, Berkeley, CA, USA

April 1, 2011

Corresponding Author:

Yun S. Song
University of California, Berkeley
Department of EECS
683 Soda Hall # 1776
Berkeley, CA 94720-1776
U.S.A.
Tel: +1 510 642 2211
Fax: +1 510 642 5775
E-mail: yss@cs.berkeley.edu

Running Title:

Reference-free short-read error correction

Abstract

Developing accurate, scalable algorithms to improve data quality is an important computational challenge associated with recent advances in high-throughput sequencing technology. In this paper, a novel error correction algorithm, called ECHO, is introduced for correcting base-call errors in short-reads, without the need of a reference genome. Unlike most previous methods, ECHO does not require the user to specify parameters of which optimal values are typically unknown *a priori*. ECHO automatically sets the parameters in the assumed model and estimates error characteristics specific to each sequencing run, while maintaining a running time that is within the range of practical use. ECHO is based on a probabilistic model and is able to assign a quality score to each corrected base. Furthermore, it explicitly models heterozygosity in diploid genomes and provides a reference-free method for detecting bases that originated from heterozygous sites. On both real and simulated data, ECHO is able to improve the accuracy of previous error correction methods by several folds to an order of magnitude, depending on the sequence coverage depth and the position in the read. The improvement is most pronounced toward the end of the read, where previous methods become noticeably less effective. Using a whole-genome yeast data set, it is demonstrated here that ECHO is capable of coping with non-uniform coverage. Also, it is shown that using ECHO to perform error correction as a preprocessing step considerably facilitates *de novo* assembly, particularly in the case of low to moderate sequence coverage depth.

Software: ECHO is publicly available at <http://uc-echo.sourceforge.net> under the Berkeley Software Distribution License.

Introduction

Over the past few years, next-generation sequencing (NGS) technologies have introduced a rapidly growing wave of information in biological sciences; see Metzker (2010) for a recent review of NGS platforms and their applications. Exploiting massive parallelization, NGS platforms generate high-throughput data at very low cost per base. An important computational challenge associated with this rapid technological advancement is to develop efficient algorithms to extract accurate sequence information. In comparison with traditional Sanger sequencing (Sanger et al. 1977), NGS data have shorter read lengths and higher error rates, and these characteristics create many challenges for computation, especially when a reference genome is not available. Reducing the error rate of base-calls and improving the accuracy of base-specific quality scores have important practical implications for assembly (Peng et al. 2010; Simpson et al. 2009; Chaisson et al. 2009; Medvedev and Brudno 2008; Zerbino and Birney 2008; Sundquist et al. 2007; Li et al. 2008; Langmead et al. 2009; Butler et al. 2008); polymorphism detection, especially rare ones (Li et al. 2008; Brockman et al. 2008); and downstream population genomics analysis of NGS data (Jiang et al. 2009; Hellmann et al. 2008).

To improve data quality, it is well recognized that development of accurate, scalable computational tools must parallel the rapid advancement in sequencing technology. There are two main approaches to addressing this challenge: *i*) One approach is to develop improved image analysis and base-calling algorithms. This line of work has been pursued by several researchers in the past, including ourselves [see Whiteford et al. (2009); Erlich et al. (2008); Rougemont et al. (2008); Kao et al. (2009); Kao and Song (2010); Kircher et al. (2009)]. Indeed, by employing more sophisticated statistical methods, it has been demonstrated that it is possible to deliver significant improvements over the tools developed by the manufacturers of the sequencing platforms. *ii*) An alternative approach is to correct for potential errors after base-calling has been performed by leveraging on the fact that each position in the genome on average is sequenced multiple times. Several methods have been developed for this approach; e.g., see Gajer et al. (2004); Chaisson et al. (2009); Schröder et al. (2009); Shi et al. (2009); Chin et al. (2009); Salmela (2010); Qu et al. (2009); Wijaya et al. (2009); Yang et al. (2010).

The goal of the present paper is to introduce a novel, efficient computational method for the latter approach. Our error correction algorithm is called ECHO, and it has the following notable features, most of which are unique to our method:

1. Whereas previous reference-free error correction algorithms (Chaisson et al. 2009; Schröder et al. 2009; Shi et al. 2009; Salmela 2010) typically rely on the user to specify some of the key parameters, of which optimal values are typically unknown *a priori*, ECHO automatically finds the optimal parameters and estimates error characteristics specific to each sequencing run.
2. ECHO is based on a probabilistic framework and can assign a quality score to each corrected base. Quality scores are useful and sometimes necessary in downstream analysis.
3. Previous reference-free error correction algorithms (Chaisson et al. 2009; Schröder et al. 2009; Shi et al. 2009; Salmela 2010) for short-reads are based on *k*-mer or substring frequencies. ECHO instead relies on finding overlaps between reads. Despite being more computationally intensive, this approach retains more of the information contained in the reads without aggregating away potentially useful relationships. Although error correction based on read overlap has been tried previously with Sanger reads (Batzoglou et al. 2002), ECHO differs from previous works in that it is specialized for high-throughput short-reads.

4. ECHO explicitly models heterozygosity in diploid genomes and allows one to process diploid data in a novel way. Specifically, for each length- l haplotype (or short-read) in the input data, ECHO can infer a likely length- l genotype sequence from which the haplotype may have originated, and position-specific quality scores can be assigned to each genotype. For each input short-read, this approach provides a way, in the absence of a reference genome, to detect the bases that originated from heterozygous sites in the sequenced diploid genome and to infer the corresponding genotypes.

ECHO performs error correction by first finding overlaps between reads. This procedure requires not only computational considerations to handle the enormous number of reads typically generated by NGS platforms, but also requires an effective method to filter out false overlaps. ECHO achieves this by imposing a maximum error tolerance in the overlaps and a minimum overlap length. These stringent requirements typically filter out many of the potential false overlaps, provided that the genome is not repeat-rich. However, the optimal error tolerance and overlap requirement are usually unknown *a priori*. Whereas previous error correction algorithms require the user to specify the key parameters, which may greatly affect the performance of the algorithm, ECHO automatically determines the optimal values for the error tolerance and minimum overlap length by utilizing assumptions on the coverage distribution of the reads.

Once ECHO determines the relevant parameters and finds the overlaps between reads, it estimates the error characteristics of the data using an expectation-maximization (EM) procedure. Our statistical approach attempts to accurately model the error behavior of the sequencing platform. This is in contrast to other error correction algorithms that typically use an error threshold or coverage cutoff, which may not be statistically motivated. Furthermore, in ECHO, error modeling is performed for each input data set to characterize more effectively the unique properties of different sequencing runs. Utilizing the read overlaps and the error characterization, ECHO performs error correction on the reads using a maximum *a posteriori* procedure. Because the sequencing behavior is modeled statistically, meaningful quality scores can be assigned to the bases of the corrected reads.

Repeats in the genome are challenging to handle since they may lead to false overlaps. In general, ECHO is less sensitive to repeats than are previous error correction methods that rely on k -mer or substring frequencies. By retaining the relationships among reads more explicitly, rather than computing aggregate statistics on substring frequencies, ECHO is better able to handle repeat patterns in the genome. However, the obstacles imposed by repeat regions are still significant, and more sophisticated methods that incorporate additional information will be necessary to overcome these challenges adequately.

To test the performance of our algorithm, we consider real short-read data generated by Illumina's Genome Analyzer (GA) I and II, as well as synthetic short-read data simulated to mimic GA's error characteristics. For GA, most of the sequencing errors are miscall errors (as opposed to indels), with the property that the error rate generally increases toward the end of the read. We show that ECHO is able to improve the accuracy of previous error correction methods by several folds to an order of magnitude, depending on the sequence coverage depth and the position in the read. In particular, provided that the sequence coverage depth is moderate to high (roughly, 15 or higher), ECHO remains effective throughout the entire read length, typically reducing the error rate at the end of the read from over 5% to under 1%.

ECHO can process relatively small genomes such as that of yeasts or fungi on a desktop computer. To demonstrate this point and to show that ECHO is capable of coping with non-uniform coverage that may arise in real data, we apply our method on a real whole-genome yeast data set.

In addition to the improvement in data quality, we also examine here the effects of error correc-

tion on *de novo* assembly. Assembly is particularly challenging when the sequence coverage depth is low to moderate, and it may benefit considerably from the improved data quality delivered by error correction. In this paper, we show that performing error correction as a preprocessing step facilitates *de novo* assembly significantly.

A software implementation of the algorithm described in this paper is publicly available at <http://uc-echo.sourceforge.net> under the Berkeley Software Distribution License.

Methods

In this section, we describe our error correction algorithm ECHO, which is divided into two stages: 1) **neighbor finding** and 2) **maximum *a posteriori* error correction**. Since we assume that the reference genome is not available, we first devise a clustering algorithm to find “neighboring” reads that presumably cover the same region of the genome. Upon finding the putative neighbors of each read, we then correct for sequencing errors by employing a maximum *a posteriori* estimation procedure for each base of the read.

Neighbor finding

In a sequencing run with a moderate to high coverage depth, each position in the genome on average is sequenced multiple times. The basic idea behind error correction is to leverage on this redundant information. When there is a reference genome, identifying the reads that cover the same region can be done via “read mapping” (Li et al. 2008; Langmead et al. 2009; Li and Durbin 2010). However, in the absence of a reference genome, the task of identifying neighbors becomes more challenging. Below, we devise a neighbor-finding algorithm based on hashing, rather than an explicit search of all pairs of reads, which would be computationally prohibitive. The approach is similar to the one in Batzoglou et al. (2002) to find overlaps between reads. However, their method is geared toward Sanger reads, whereas ECHO is specialized for short-read data, which requires different computational considerations.

In a preprocessing step, we first augment the set of reads with their reverse complements. For a given keyword length k , we iterate through the reads and their complements, and create a hash table mapping each k -mer to the reads that contain it, along with the position in the read where the k -mer occurs. If a k -mer occurs multiple times in a read, for computational efficiency we record only the position that is closest to the beginning of the read; this choice is motivated by the fact that, for the Illumina platform, early cycles of a sequencing run generally have higher quality than do later cycles. (If the read is instead a reverse complement that was added in the preprocessing step, we keep the position that is closest to the end of the read, which would have higher quality than the beginning.) Assuming k to be a constant, this hashing stage requires expected $O(Nl)$ time, where N is the number of input reads and l is the read length.

Once the hash table mapping k -mers to the reads is constructed, we consider more thorough alignments to construct an adjacency list connecting neighboring reads together. We run through the k -mers, and for each one, consider its associated set of reads. For every pair of reads, we do a simple alignment to assess the quality of the overlap between the two reads. Figure 1 illustrates how the two reads are aligned given a common k -mer. We use ϵ to denote the fraction of errors in the overlap, i.e.,

$$\epsilon = \frac{m}{\omega},$$

where m is the number of mismatches in the overlap and ω is the length of the overlap. The “score” of an overlap is defined as $1/\epsilon$, and it is used to select the best alignment from among

several possible alignments between two reads. Further, we require that the overlap length be greater than the *minimum overlap parameter* ω^* and that the fraction of errors in the overlap ϵ be less than ϵ^* , the *error tolerance*:

1. $\omega \geq \omega^*$, the overlap length must be at least the minimum required overlap length ω^* .
2. $\epsilon \leq \epsilon^*$, the fraction of mismatched bases is less than the error tolerance ϵ^* .

The two threshold parameters ω^* and ϵ^* are selected automatically in ECHO, as described below in **Parameter selection**.

In practice, the best alignment for the overlap between two reads involves aligning together their common k -mer, since it already provides k matching bases. In this case the alignment is given immediately by the position of the k -mer itself. Using this method, an alignment between a pair of reads is found in $O(1)$ time. When two reads share multiple common k -mers, the scoring method described above is used to select the best alignment, and only the best alignment is kept in the adjacency list. In the case of ties, the alignment with the greatest overlap length is chosen.

Looking only at common k -mers between reads is typically sufficient to find overlaps, because for a small enough k , the chance that two truly overlapping reads do not have any common k -mer in the overlap is small, provided that the sequencing error rate is low. Both reads only need an error-free stretch long enough to cover the k -mer in the overlapping region for the reads to have a common k -mer. Given a relatively low error rate, this occurs with high probability. A similar idea is also utilized in (Rasmussen et al. 2005) and (Jokinen and Ukkonen 1991). Although the current implementation does not allow for indels in the overlaps, it would be straightforward to generalize this alignment step to accommodate for indels at the expense of greater computational cost.

With $B_{\max}$ denoting the maximum number of reads associated with a single k -mer (i.e., the size of the largest bin in the hash table) and $\mathcal{K}$ denoting the total number of k -mers, the neighbor finding stage requires $O(Nl + \mathcal{K}lB_{\max}^2)$ time. Here, $O(Nl)$ is the time required for the hashing stage, and $O(\mathcal{K}lB_{\max}^2)$ is the time needed to process the reads associated with every k -mer, where counting the number of mismatches in an overlap takes $O(l)$ time and there are at most $B_{\max}^2$ pairs of reads to consider for each k -mer. There are times when the number of reads that contain a given k -mer is abnormally large, e.g., when the k -mer is a substring of a repetitive region or the coverage of the genome is highly non-uniform. In our algorithm, we put a cap on the k -mer bin size in the hash table such that the reads associated with a k -mer with an extreme number of overlaps are not corrected. This approach allows us to avoid correcting reads that belong to highly repetitive regions, for which the assumptions on overlaps become more tenuous.

Parameter selection

In ECHO, three parameters are required for finding the overlaps between reads: k , the keyword length; ω^* , the minimum overlap length; and ϵ^* , the threshold on the fraction of allowed errors in an overlap. The keyword length k must be chosen to balance two effects:

1. If k is too small, the number of reads associated with each k -mer will be very large. This exacerbates the running time during the neighbor finding stage because it is quadratic in the number of reads associated with each k -mer.
2. If k is too large, we might miss a significant portion of the true overlaps between reads, since it is only when two reads share a common k -mer that we even consider computing an overlap between them.

The selection of k depends on the expected error rate of the reads; e.g., k needs to be small enough to ensure that two overlapping reads have a good chance of possessing a common error-free stretch of length k . We find that choosing $k = \lfloor l/6 \rfloor$, where l is the read length, retains a high probability of two overlapping reads sharing a common k -mer, while not being too small as to significantly increase the running time.

Note that k is used only in a heuristic to find overlaps quickly. We do *not* consider k -mer distributions as in other error correction techniques, and hence an accurate choice of k is not essential as long as it errs on the side of caution. We show in **Results** that the performance of the algorithm is robust with respect to the choice of k .

In contrast, selecting the thresholds ω^* and ϵ^* appropriately has a more significant impact on the performance of the algorithm. If ω^* and ϵ^* are too strict, then neighboring reads that should be linked together are missed. If they are too lenient, then reads that do not truly overlap will be mistakenly linked together. Because the performance of the error correction algorithm depends greatly on the accuracy and correctness of the read overlaps, it is important to select ω^* and ϵ^* carefully.

The idea behind finding appropriate ω^* and ϵ^* relies on the assumption that the coverage at a given position of the genome approximately follows a Poisson distribution, given that the reads are sampled uniformly at random from the genome. Given the set of overlaps between reads, which we infer without the use of a reference genome, we can compute the empirical coverage distribution for every position, assuming that reads overlap because they come from the same region of the genome. Note that this assumption on the overlaps and the coverage distribution becomes less accurate as the genome structure becomes more repetitive, which may cause false overlaps to be inferred more frequently.

We adopt the following approach to find ω^* and ϵ^* that produce an empirical coverage distribution most closely resembling a Poisson distribution: Given a fixed overlap requirement ω_0^* and error tolerance ϵ_0^* , we find the Poisson distribution that minimizes the total variation distance between the empirical distribution and the Poisson distribution. Figure 2 illustrates the total variation distance between the Poisson distribution and the empirical distribution. By running the neighbor finding algorithm on a grid search, we select the parameters ω^* and ϵ^* that have the minimum total variation distance. As a grid search is typically computationally intensive, for efficiency, we perform the neighbor finding algorithm only on a subset of the data. Although the empirical coverage distribution may not exactly resemble a Poisson distribution, we find that using it as a guide selects ω^* and ϵ^* coherent with the assumptions on the read coverage distribution.

Maximum *a posteriori* error correction for haploid genomes

The neighbor finding stage produces the putative overlap relationships between reads. Here we formulate a maximum *a posteriori* estimation procedure for error-correction. In the Illumina platform, most of the sequencing errors are miscall errors (as opposed to indels), with the property that the error rate generally increases towards the end of the read. For ease of exposition, we consider only the case for the original reads; the procedure for the reverse complements is analogous. We assume that miscall errors are made independently within each read and also across different reads. To be more precise, we assume that each base in a given read is distributed as a multinomial distribution with parameters that depend on the position of the base within the read and the true corresponding base in the original genome. If the true base at position m of a given read is b , the probability that it is called as b' is denoted by $\Phi_{b,b'}^{(m)}$. Following Kao et al. (2009); Kao and Song (2010), we refer to $\Phi^{(m)} = (\Phi_{b,b'}^{(m)})_{b,b' \in \{A,C,G,T\}}$ as the *confusion matrix* for position $1 \leq m \leq l$, where l is the read length. It gives the probability that the true base is miscalled as another base given its position in

the read. We later provide an EM procedure for estimating $\Phi^{(m)}$ for $1 \leq m \leq l$.

Let r denote a read, and suppose that the base at position m of r , which we denote r_m , originated from position i of the genome S . In the absence of any sequencing error, read r will be identical to the substring $H \equiv S_{i-m+1:i+l-m}$, where l is the length of the read, and $S_{a:b}$ denotes the substring of S from position a to position b , inclusive. In other words, H is the “perfect” read that r would be if there were no sequencing errors. The entry $\Phi_{b,b'}^{(m)}$ in the confusion matrix $\Phi^{(m)}$ is defined as

$$\Phi_{b,b'}^{(m)} = \mathbb{P}(r_m = b' \mid H_m = b), \quad (1)$$

the probability that the m th position of read r is called as b' given that the true base is b . Note that $\Phi^{(m)}$ is indexed by m , the called-base’s position in the read. This is critical because the Illumina platform suffers from progressively worse accuracy toward the ends of its reads, and hence it is essential to model this behavior statistically.

We use the following model for the distribution on observed reads:

1. Bases are distributed uniformly at random in the genome:

$$S_i \sim \text{Unif}\{A, C, G, T\}. \quad (2)$$

2. Every base is called independently, and the distribution on the called base depends on its position in the read and the true base:

$$r_m \mid (H_m = b) \sim \text{Mult}(1; \Phi_{b,A}^{(m)}, \Phi_{b,C}^{(m)}, \Phi_{b,G}^{(m)}, \Phi_{b,T}^{(m)}). \quad (3)$$

Based on this model, we obtain the posterior distribution

$$\mathbb{P}(H_m = b \mid r_m = b') = \frac{\frac{1}{4} \Phi_{b,b'}^{(m)}}{\frac{1}{4} \sum_{a \in \{A, C, G, T\}} \Phi_{a,b'}^{(m)}}, \quad (4)$$

which gives the probability that, given we observe base b' in a read, the true base is in fact b (which may or may not be the same as b').

Now we use the assumption that overlaps of neighboring reads cover the same positions in the genome. For every position belonging to a particular overlap, we consider every read that contributes to the overlap. Each read provides its own base-call for that position, which is used to “sharpen” the posterior distribution described in (4).

Let Ω be the multiset of base-calls from reads that correspond to (overlap with) position m in read r . The posterior of H_m to be base b is then given by

$$f_{r,m}(b) = \frac{\frac{1}{4} \prod_{b' \in \Omega} \Phi_{b,b'}^{(m)}}{\sum_{a \in \{A, C, G, T\}} \left[\frac{1}{4} \prod_{b' \in \Omega} \Phi_{a,b'}^{(m)} \right]}. \quad (5)$$

The maximum *a posteriori* (MAP) estimate of H_m is then

$$H_m^{\text{MAP}} = \underset{b \in \{A, C, G, T\}}{\text{argmax}} f_{r,m}(b). \quad (6)$$

This MAP estimate of H_m is taken as the base at position m in read r . We perform this maximum *a posteriori* correction for all the reads in the data set. The procedure for handling a reverse complement read is analogous to the aforementioned steps.

Coverage checks

An abnormally large set of overlapping reads usually indicates non-uniform coverage or that the reads are sampled from a repetitive region, and suggests that our assumptions on overlapping reads most likely do not hold. From the overlaps found in the neighbor finding stage, we have an estimate of the expected coverage μ , as described in **Parameter selection**. If the coverage at a position is greater than $\mu + \alpha\sqrt{\mu}$ (α standard deviations greater than the average coverage), we do *not* perform a correction. Our empirical study suggests that the precise value of α does not generally affect the accuracy of the algorithm; $4 \leq \alpha \leq 6$ is typically effective. This “filtering” allows ECHO to be robust against parts of the genome that have repetitive structure or have highly non-uniform coverage. We refer to this filtering as a “coverage check,” because it ensures that the estimated coverage for a read looks reasonable before correction is attempted.

Quality scores

Providing an accurate measure of per-base quality has practical importance. For instance, MAQ (Li et al. 2008), a widely-used read mapping algorithm, utilizes base-specific quality scores to produce mapping quality scores and to call variants. A widely adopted definition of quality score is that used in *Phred* (Ewing and Green 1998); it is a transformed error probability of a base, defined more precisely as follows: Let $e_{r,m}$ denote the probability that r_m , the m th base of read r , is erroneously called. Then the Phred quality score $q_{r,m}$ for r_m is given by

$$q_{r,m} = -10 \log_{10} e_{r,m}. \quad (7)$$

Since ECHO provides an estimate of the posterior probability of each base, the error probability $e_{r,m}$ of base r_m^{ECHO} , where r^{ECHO} denotes the ECHO-corrected read, can be estimated by

$$e_{r,m} = 1 - f_{r,m}(r_m^{\text{ECHO}}), \quad (8)$$

where $f_{r,m}(\cdot)$ is defined as in (5). The Phred quality score for r_m^{ECHO} is then obtained by using (8) in (7).

Generalization to diploid genomes

In this section, we show how our approach can be generalized to handle short-reads from diploid genomes, in which case the main challenge lies in distinguishing heterozygotes from sequencing errors. We use the same neighbor finding algorithm as in the haploid case, but modify the maximum *a posteriori* error correction step as described below. Again, our exposition refers to the original reads; the case for reverse complements is analogous.

The unordered genotype at a given position of a diploid genome is either a homozygote or a heterozygote. We use $\text{Hom} = \{\{A, A\}, \{C, C\}, \{G, G\}, \{T, T\}\}$ to denote the set of homozygous genotypes and $\text{Het} = \{\{A, C\}, \{A, G\}, \{A, T\}, \{C, G\}, \{C, T\}, \{G, T\}\}$ to denote the set of heterozygous genotypes. The set of all possible genotypes is denoted by $\mathcal{G} = \text{Hom} \cup \text{Het}$.

Given a length- l haplotype sequence $r = (r_1, \dots, r_l) \in \{A, C, G, T\}^l$, let $D = (D_1, \dots, D_l) \in \mathcal{G}^l$ denote the true unphased genotype sequence in the diploid genome from which r originated. Our goal is to assign a length- l unphased genotype sequence $\mathcal{D}(r)$ to each read r , such that $\mathcal{D}(r)$ is as close to D as possible. For each position m in r , we assume that the two alleles in D are equally likely to be the source of the m th nucleotide r_m . Hence, sequencing errors at that position are characterized by the following generalized confusion matrix $\Psi^{(m)} = (\Psi_{g,b}^{(m)})_{g \in \mathcal{G}, b \in \{A, C, G, T\}}$:

$$\Psi_{\{a,a'\},b}^{(m)} = \mathbb{P}(r_m = b \mid D_m = \{a, a'\}) = \frac{1}{2} \left(\Phi_{a,b}^{(m)} + \Phi_{a',b}^{(m)} \right),$$

where $\{a, a'\}$ is the true genotype, b is the base at position m in read r , and $\Phi_{(\cdot, \cdot)}^{(m)}$ is defined as in (1).

We use h to denote the prior probability that a given site in the diploid genome is heterozygous. Then for a particular genotype $g \in \mathcal{G}$, the prior is

$$\mathbb{P}(g) = \begin{cases} \frac{1-h}{4}, & \text{if } g \in \text{Hom}, \\ \frac{h}{6}, & \text{if } g \in \text{Het}. \end{cases}$$

That is, given the zygosity, we assume the uniform distribution over all possible genotypes. Analogous to the haploid case, the posterior distribution is given by

$$\mathbb{P}(D_m = g \mid r_m = b) = \frac{\Psi_{g,b} \mathbb{P}(g)}{\sum_{g' \in \mathcal{G}} \Psi_{g',b} \mathbb{P}(g')}.$$

Let Ω be the set of corresponding base-calls from reads that overlap with position m in read r . Then the posterior of D_m being genotype g is

$$f_{r,m}^d(g) = \frac{\mathbb{P}(g) \prod_{b' \in \Omega} \Psi_{g,b'}}{\sum_{g' \in \mathcal{G}} [\mathbb{P}(g') \prod_{b' \in \Omega} \Psi_{g',b'}]},$$

and the maximum *a posteriori* estimate of D_m is

$$D_m^{\text{MAP}} = \underset{g \in \mathcal{G}}{\operatorname{argmax}} f_{r,m}^d(g).$$

In ECHO, this MAP estimate is taken to be the inferred genotype corresponding to position m of read r . As in the haploid case, if the estimated coverage at a particular position is much greater than the expected coverage, the maximum *a posteriori* estimate is not used and instead the algorithm outputs the genotype $\{r_m, r_m\}$, analogous to not correcting the base in the haploid case.

Let $\mathcal{D}(r) = (\mathcal{D}_1(r), \dots, \mathcal{D}_l(r))$ be the sequence of genotypes estimated by ECHO for read r . The error probability for $\mathcal{D}_m(r)$, the genotype at position m of $\mathcal{D}(r)$, is

$$e_{r,m} = 1 - f_{r,m}^d(\mathcal{D}_m(r)),$$

and (7) can be used to compute the quality score for $\mathcal{D}_m(r)$. Incidentally, note that the above algorithm for the diploid case with $h = 0$ reduces to the algorithm described earlier for the haploid case.

Estimating confusion matrices

If a reference genome is available, the confusion matrices $\Phi^{(m)}$, for $1 \leq m \leq l$, may be estimated by first aligning reads to the reference genome, treating mismatches as errors. However, this estimation can be biased, since mismatches may also originate from single nucleotide polymorphism or structural variations. Another method of estimating the confusion matrices is to use a control lane that sequences a known genome. This method avoids the potential biases caused by variations between the sampled genome and the reference genome, but this solution is costly and error characteristics can still differ between different lanes or runs.

Based on our probabilistic model, in which the true bases in the genome are unobserved latent variables, we adopt the EM algorithm to estimate confusion matrices. The advantage of the EM

algorithm is that estimation can be performed without a reference genome. To start the estimation of $\Phi^{(m)}$, for $1 \leq m \leq l$, we first obtain the overlapping relationships between reads with the parameters k , ω^* , and ϵ^* as described in **Neighbor finding** and **Parameter selection**. Then, we fix the neighbor relationships and apply the EM algorithm, with the initial confusion matrix having 0.99 in the diagonal and 0.01/3 for all off-diagonal entries. The EM algorithm typically terminates after a few iterations.

Results

In this section, we evaluate the performance of our algorithm ECHO and compare it to previous error correction methods. In particular, we consider the Spectral Alignment (SA) algorithm used in the preprocessing step of the *de novo* assembler EULER-USR (Chaisson et al. 2009). This error correction algorithm is based on examining the k -mer spectrum of the input reads (Pevzner et al. 2001; Chaisson et al. 2004). Because SA is sensitive to the choice of k , for each experiment we ran SA over a range of k and selected the k that generated the optimal results (this was typically around $k = 15$).

SHREC (Schröder et al. 2009) is a recent error correction algorithm based on a generalized suffix trie data structure; it essentially considers the substring spectrum of the input reads, i.e. a range of k -mers where k is over a given interval. It requires the user to specify the size of the genome, and we provided it with the true genome size for each data set we considered. Further, SHREC has a parameter called “strictness,” with the default value being 7. For coverage depths $c < 40$, SHREC failed to run with the default parameter value, so we used the largest value less than 7 that the program accepted. As discussed in **Methods**, ECHO automatically chooses its parameters k , ω^* , and ϵ^* .

In addition to the improvement in data quality, we also examine the effects of error correction on *de novo* assembly.

Data and experiment setup

We tested the performance of our method on both real and simulated data over a range of coverage depths. Specifically, the following data sets were considered:

- D1 (PhiX174): We used standard resequencing data of the PhiX174 virus. 76 bp reads were obtained from the Illumina GA-II platform, with the viral sample in a single lane of the flow cell. The length of the PhiX174 genome is 5386 bp, and the short-read data had a coverage depth of around 210,000. In our experiments, various coverage depths were obtained by sampling without replacement from the actual data set. We used Illumina’s standard base-calling software, Bustard, to produce base-calls. To create labeled test data, we aligned the reads against the reference genome and regarded every mismatch as a sequencing error.
- D2 (*D. mel.*): Another real data set we used is that of *D. melanogaster* inbred line RAL-399, sequenced and assembled by the Drosophila Population Genomics Project (DPGP, <http://www.dpgp.org/>). The short-reads were downloaded from the NCBI Sequence Read Archive (<http://www.ncbi.nlm.nih.gov/Traces/sra/sra.cgi>). The data consisted of 36 bp reads sequenced on the Illumina GA I platform and base-called by Bustard. The overall coverage depth of the data set was roughly 13. To create a labeled test set, we aligned the reads against the genome assembly (release 1.0) produced by DPGP. Then we selected the reads that mapped to chromosome 2L and regarded mismatches as sequencing errors.

- D3 (Simulated *D. mel.*): In the test described above in D2, errors in the assembled genome (i.e., differences between the assembled genome and the true genome) may confound the assessment of the accuracy of error correction algorithms. To avoid this potential problem, we generated 76 bp simulated reads from a 100 kbp or a 5 Mbp region of chromosome 2L of the RAL-399 inbred line mentioned above. To simulate each 76 bp read, we chose a starting position uniformly at random from the chosen region or its reverse complement, and introduced errors by selecting a random “error template” as follows: First, we selected a read randomly from the PhiX174 data set D1 and aligned it to the PhiX174 reference genome to determine the positions of errors. Then, we took that error template and added errors to the simulated *Drosophila* read at the corresponding locations. The miscalled base was determined according to the position-specific empirical confusion matrices estimated from the PhiX174 data, because the “error template” from the sampled PhiX174 read only provides the *positions* of errors. This approach allowed us to simulate more realistic error patterns; e.g., the correlation of errors in a read and the clustering of errors on a subset of the reads. Our simulation method permitted miscall errors but no indels.
- D4 (Simulated human data with repeats and duplicated regions): To evaluate the performance of ECHO on sequences with repeats and duplicated regions, we generated 76 bp reads from a 250 kbp region of human chromosome 16 known to have a duplicated gene (Martin et al. 2004). More specifically, the region spans chromosomal coordinates 20240000 – 20490000 of the GRCh37 reference assembly of chromosome 16, downloaded from the NCBI Genome database. We followed the same procedure as in D3 to introduce errors into reads randomly selected from the 250 kbp region.
- D5 (Simulated diploid data): To generate a known diploid genome, we took the 100 kbp region mentioned in D3 and mutated each base with probability 0.001, with the new base drawn uniformly at random from the alternative bases. To simulate each 76 bp read, we followed the same procedure as in D3 to add errors to a randomly sampled read from the diploid strands and their reverse complements. For this diploid data set, the sequence coverage depth is given by $\frac{NI}{L}$, where N denotes the number of reads, $l = 76$, and $L = 100,000$.
- D6 (*Saccharomyces cerevisiae*): This real data set consisted of 7.1 million reads from the whole-genome sequencing of a laboratory-evolved yeast strain derived from DBY11331 (Gresham et al. 2008), Accession Number SRR031259 in the NCBI Sequence Read Archive. Because the yeast strain differed slightly from the *S. cerevisiae* reference genome, we created a new assembly using MAQ (Li et al. 2008) and the standard reference sequence. We used this newly assembled genome to evaluate error correction performance in our experiments. The data set contained single-end 36 bp reads from the Illumina GA II platform, covering all 16 chromosomes and the mitochondrial DNA of *S. cerevisiae*. The overall coverage depth of the data set was roughly 21.

Error correction accuracy for haploid genomes

We compared the error correction accuracy of SA, SHREC, and ECHO on the data sets described in the previous section. Because the output of SHREC is partitioned into “corrected” and “discarded” reads, we measured the performance of SHREC by considering only the corrected reads.

In what follows, we use the term *by-base* error rate to refer to the ratio of the total number of incorrect bases in the reads to the total combined length of the reads. We also consider the *by-read* error rate, which corresponds to the ratio of the number of reads with *at least* one miscalled

base to the total number of reads. Table 1 shows the by-base and by-read error rates for haploid data sets D1–D4 before and after running the three error correction algorithms. On all data sets, ECHO was more effective than SA and SHREC at reducing the error rates. In general, SA did not seem effective at reducing the by-base error rate. SHREC was more effective than SA, whereas our algorithm improved on SHREC by several folds on both real and simulated data. Table 2 shows the numbers of corrected errors and introduced errors, as well as the *gain*, for the same data sets after error correction. The gain is defined as the number of corrected errors minus the number of introduced errors, divided by the number of actual errors, in the same manner as in Yang et al. (2010). As the table shows, the gain for ECHO was substantially higher than that for SA or SHREC.

Observe that all three algorithms generally improved the by-read error rates noticeably, with ECHO being more effective than both SA and SHREC. It is interesting that while SA only moderately improved the by-base error rate, it was effective at improving the by-read error rate.

To compare the performance of the three error correction algorithms in more detail, we examined the *position-specific* by-base error rates; i.e., the by-base error rate at position i is the ratio of the number of reads with an incorrect base at position i to the total number of reads. For PhiX174 data D1 with sequence coverage depth 30, Figure 3(a) illustrates the position-specific by-base error rates before and after applying the error correction algorithms. It is well-known that the error rate of the Illumina platform generally increases toward the end of the read, reaching as high as 5% in the example shown. Although SA and SHREC were able to reduce this effect to a certain degree, they both became less effective for later positions. In contrast, ECHO remained quite effective throughout the entire read length. In particular, the error rate at the end of the read was reduced from about 5% to less than 1%.

We considered the effect of coverage depth on the performance of our algorithm. Figure 3(b) shows the position-specific by-base error rates before and after applying ECHO on PhiX174 data with varying sequence coverage depths. At a coverage depth of 15, the error rate could be controlled throughout the entire read length, resulting in only a slight increase in the error rate toward the end of the read.

Although ECHO improved on the other error correction methods on the simulated human data set D4 (see Tables 1 and 2), the improvement was not as pronounced as for the *Drosophila* and PhiX174 data sets. This demonstrates that ECHO is most effective on genomes with limited repetitive structure. Although the read overlap approach might be less susceptible to repeat regions than a k -mer spectrum method, it does not adequately overcome this obstacle. Further work is required to handle repeat regions, which are common in more complex genomes, such as mammalian genomes.

Whole-genome error correction

To evaluate the performance of ECHO on whole genome data, we ran an experiment on the yeast data D6. Results from the experiment are shown in Tables 3 and 4. The performance of all evaluated error correction algorithms was noticeably worse for this data set than for the others. Closer examination of the data revealed that the mitochondrial DNA (mtDNA) had much higher coverage than the other chromosomes, 210X compared to 19.5X. Because ECHO compares a read's estimated coverage to the estimated coverage over the *entire* genome, nearly every read from the mtDNA was not corrected. This contrasts with SHREC, which does not account for such drastic differences in coverage across the genome. This suggests that although ECHO assumes uniform coverage of the genome when determining its parameters, when it encounters data with highly non-uniform coverage, it avoids confusion caused by the non-uniform coverage by requiring the reads to

pass the coverage checks described earlier. It should be noted that repetitive structure would also cause a read not to pass the coverage check, which would cause ECHO to be more conservative around repeat regions.

To better quantify the performance of the error correction algorithms, we divided the data set into two groups: reads that passed the coverage checks and those that did not. Restricted to those reads that passed the coverage checks (i.e., the reads on which error correction was actually attempted), ECHO was able to reduce the by-base error rate from 1.28% to 0.83% (which corresponds to about 35% improvement), while reducing the by-read error rate from 17.3% to 8.4%. Hence, ECHO's performance on this restricted read set was almost as good as that for the other real data sets (e.g., D1 and D2). In comparison, for the reads on which SHREC attempted error correction, it was able to reduce the by-base error rate from 0.89% to 0.79% (which corresponds to about 11% improvement), while reducing the by-read error rate from 14.8% to 10.6%. SA did not make any error correction on this data set.

Figure 4 shows the gain for ECHO and the coverage along chromosome 1. As mentioned before, the gain corresponds to the number of corrected errors minus the number of introduced errors, divided by the number of actual errors. The plots illustrate that when the coverage increases significantly, the gain tends to drop. This is because the reads from these positions generally do not pass the coverage checks ECHO imposes.

Table 4 illustrates that SHREC corrected about 1.5 times more bases than did ECHO, but the number of errors introduced by SHREC was 41 times greater than that introduced by ECHO. As a result, the gain for ECHO was higher than that for SHREC. Because of the coverage checks it performs, ECHO was more conservative in correcting reads coming from highly covered regions of the genome. We expect this to be an advantageous feature of ECHO, as a high *estimated* coverage is often indicative of repeat structure.

Robustness with respect to the choice of the keyword length k

In **Methods**, under **Parameter selection**, we provided a rough guideline for choosing the keyword length k in the neighbor finding algorithm. Here we show empirically that the performance of ECHO with the suggested setting $k = \lfloor l/6 \rfloor$, with l being the read length, is comparable to the performance of the algorithm with the optimal k that maximizes the error correction performance.

For this study, we subsampled reads from data D3 to create a data set consisting of 25,000 reads of length 76 bp. This corresponds to a sequence coverage depth of 19. The average by-base error rate in the simulated reads was around 1%. For each given k , we used the method described in **Parameter selection** to determine ω^* and ϵ^* . These parameters, k , ω^* and ϵ^* , were then used to perform error correction on the reads. For the simulated data we considered, the value of k that led to the largest improvement in the error rate was $k = 7$, in which case ECHO reduced the error rate from 1% to 0.107%. In comparison, the associated error rate for $k = \lfloor l/6 \rfloor = 12$ was 0.109%, which is comparable to the optimal case.

Error correction accuracy for diploid genomes

As described in **Methods**, ECHO can handle diploid data as well as haploid data. Recall that ECHO explicitly models heterozygosity in diploid genomes and has a parameter, denoted h , corresponding to the probability that a given site is heterozygous. Setting $h = 0$ reduces the model to the haploid case. Neither SA nor SHREC explicitly models diploidy, so they may be compared with the case of $h = 0$ in our model.

We tested the methods on simulated diploid data D5. In simulating the diploid genome, the

probability of heterozygosity was set to 10^{-3} for each site. Table 5 shows a summary of error correction results on diploid data D5. As in the haploid case, SA was not effective in reducing the by-base error rate, though it was able to reduce the by-read error rate reasonably well. SHREC significantly improved on SA in terms of the by-base error rate, whereas their by-read error rates were comparable. ECHO with $h = 0$ improved on both SA and SHREC by several folds for coverages 30 and higher. For these coverage depths, ECHO reduced the by-read error rate from around 27% to about 2%. The improvement in the by-base error rate was equally significant in proportion. For $h > 0$, ECHO displayed accuracy greater than that for $h = 0$, demonstrating the utility of explicitly modeling the diploidy of the genome.

Table 6 shows the detection accuracy of heterozygous sites in the diploid data. The *precision* and *recall* are shown for varying levels of coverage and for both $h = 10^{-3}$ and $h = 10^{-4}$. Precision and recall are commonly-used performance measurements in information retrieval tasks. In this context, precision is the number of correctly called heterozygous sites divided by the number of all called heterozygous sites (including incorrectly called heterozygous sites), and recall is the number of called heterozygous sites divided by all true heterozygous sites.

Every genotype called by ECHO is given a quality score. As illustrated in Table 6, we can trade off recall for higher precision by filtering based on quality. These results demonstrate that, with high enough coverage, ECHO provides a reliable reference-free method of detecting bases that originated from heterozygous sites and inferring the corresponding genotypes.

Effects on de novo assembly

Intuitively, improved data quality should facilitate assembly. To assess the effects of various error correction methods on *de novo* assembly, we used Velvet (Zerbino and Birney 2008) to carry out the assembly. Like many other *de novo* assembly algorithms, Velvet is based on k -mers and the associated de Bruijn graph representation. In each experiment, we used $k = 39$ and automatic coverage cutoff, and provided Velvet with the true expected coverage.

The data sets we considered were real PhiX174 data D1 and simulated data D3 from a 5 Mbp region of *D. melanogaster*; the former experiment was repeated 20 times, while the latter was repeated twice.

For the PhiX174 data set, Velvet was run on the following four sets of short-reads:

- Uncorrected reads, corresponding to the original simulated reads.
- Error-corrected reads processed by SA, SHREC, or ECHO.

For the simulated *D. melanogaster* data sets, Velvet was also run on the following set of short-reads, in addition to the above four sets:

- Perfect reads, corresponding to the error-free reads from which the uncorrected reads were simulated.

The results for PhiX174 are shown in Table 7, where “Max” denotes the maximum contig length, and “N50,” defined in the caption, is a statistic commonly used to assess the quality of *de novo* assembly. Larger values of Max and N50 indicate better assembly quality. For nearly all sequence coverage depths, note that using ECHO-corrected reads led to the largest Max and N50 values.

The assembly results for simulated 5 Mbp *D. melanogaster* data are summarized in Table 8. There was a clear advantage to performing error correction prior to assembly, and error correction produced an appreciable improvement over uncorrected reads. In addition to Max and N50, we assessed the assembly quality using the following two new measures:

- SeqCov1000 is the percentage of the genome covered by the contigs of length ≥ 1000 bp.
- Error1000 is the percentage of the total number of assembly errors (measured by edit distance from the true genome, with a penalty of 1 for mismatch, insertion, or deletion) in those contigs with length ≥ 1000 bp, with respect to the total length of those contigs.

For coverage depths ≥ 20 , SeqCov1000 values for all three error correction methods were comparable to that for perfect reads.

As Tables 7 and 8 illustrate, the key advantage of ECHO is its consistency; i.e., the performance of ECHO is consistently good for all coverage depths and all data sets, while the other error correction methods seem more erratic (e.g., see the performance of SHREC in Table 7, and the performance of SA in Table 8 for coverages ≥ 20).

Running times

We ran the three error correction algorithms on simulated data from the *D. melanogaster* reference sequence to compare their running times. We fixed the sequence coverage depth at 20 and varied the genome length to adjust for different quantities of reads. Errors were added to the reads in the same way as in D3. All experiments were done on a Mac Pro with two quad-core 3.0 GHz Intel Xeon processors and 14 GB of RAM. SA and ECHO each used only a single core, whereas SHREC was allowed to use all 8 cores.

Shown in Table 9 is a summary of running times. Since our algorithm needs to determine the parameters ω^* and ϵ^* , its running time is divided into two parts: parameter selection time, shown in parentheses, and the running time for neighbor finding and error correction. Although our algorithm is slower, note that its running time is within the range of practical use. On the whole-genome yeast data D6 discussed earlier, SA took 6 minutes and SHREC took 37 minutes, while ECHO took 260 minutes. (Recall that SA did not make any error correction on this data set.)

If the entire adjacency list for the overlaps between reads can be stored in memory, the running time of ECHO scales linearly in the number of reads, provided that k is chosen large enough such that the number of reads that contain a given k -mer scales linearly in the coverage depth. In our experiments, ECHO was given $k = \lfloor l/6 \rfloor = 12$.

On the other hand, if the available amount of memory is not sufficient, our implementation first partitions the read set into smaller subsets. Then successively for the partitioned reads, the adjacency lists for the read overlaps are constructed and then merged in a tree-like fashion. Error correction is then performed on the reads while reading in only a small portion of the adjacency list at a time, reducing memory usage significantly. Using this approach and assuming the number of repeat regions in the genome is not too high, the running time of ECHO scales approximately as $O(N \log N)$, where N is the number of reads.

Discussion

In this paper, we introduced a new reference-free error correction algorithm, ECHO, for correcting sequencing errors in short-reads. It has several important novel features. One advantage of ECHO over previous methods is that it does not rely on the user to specify many of the key parameters; this is an important feature since the optimal parameter values for given input data are typically unknown to the user *a priori*. Without the need of a reference genome, ECHO efficiently clusters those reads that may have originated from the same region of the genome, and automatically sets the parameters in the assumed model. Moreover, being based on a probabilistic framework, ECHO

can assign a quality score to each corrected base; we are not aware of any other reference-free error correction algorithm that also does this.

Another novel feature of ECHO is that it explicitly models heterozygosity in diploid genomes. This allows for the detection of bases that originated from heterozygous sites in the sequenced diploid genome and to infer the corresponding genotypes, without using a reference genome. We leave as future research utilizing this information in downstream computational problems, including polymorphism detection and assembly.

The improvement in data quality delivered by our algorithm is most pronounced toward the end of the read, where previous methods become considerably less effective. In the Illumina platform, the error rate generally increases toward the end of the read. However, provided that the sequence coverage depth is sufficiently high, ECHO is able to control the error rate to stay at less than 1% throughout the entire read length (see Figures 3(a) and 3(b)). This suggests that one possible way to obtain longer reads for a given error tolerance is to allow the sequencing machine to run beyond that tolerance level and then apply an error correction algorithm afterward to reduce the error rate. This kind of strategy should help to relax the limitation on the read length imposed by the imperfectness of the chemical process.

Similar to previous error correction algorithms, ECHO is typically most effective on genomes without complex repeat structure, such as bacterial and other non-mammalian genomes. Although ECHO has several mechanisms to avoid correcting repetitive regions, such as ignoring reads with an abnormally large number of overlaps and setting the minimum overlap high enough such that the coverage distribution is Poisson distributed, repetitive regions still lead to a greater number of false overlaps, thus confounding the error correction process.

As an application to *de novo* assembly, we demonstrated that error correction can provide significant improvements in the quality of the resulting contigs. In particular, we showed that, for low to moderate coverage depths, performing error correction prior to assembly may significantly improve the quality of *de novo* assembly. It will be interesting to explore whether reference-free error correction may facilitate *de novo* transcriptome assembly (Birol et al. 2009).

The method described in this paper may be improved in a few ways. 1) In parameter selection, we assumed that the number of neighbors that overlap with any particular position in a given read is Poisson distributed. In practice, this may not be a very good assumption, and using a different distribution for fitting might lead to better choices of ω^* and ϵ^* . 2) Instead of using a subset of reads randomly chosen from the input set of reads, it may improve parameter selection if the subset is chosen more judiciously; e.g., by filtering out the reads with repeats. 3) The short-reads generated by most NGS platforms come with base-specific quality scores, but ECHO currently does not utilize that information. It is possible to incorporate per-base quality scores into ECHO, thereby further improving its accuracy. 4) Our current implementation of the neighbor-finding algorithm assumes that indel errors are rare, as is the case for the Illumina platform. For the sequencing platforms with high indel error rates (e.g., Roche's pyrosequencing technology), the alignment step in the neighbor-finding algorithm needs to be generalized to include indels. However, the overall idea underlying ECHO is still applicable.

Acknowledgments

We thank Satish Rao, Aileen Chen, Nanheng Wu, and Jerry Hong for useful discussions. This research is supported in part by an NSF CAREER Grant DBI-0846015, an Alfred P. Sloan Research Fellowship, and a Packard Fellowship for Science and Engineering to YSS; and by a National Defense Science & Engineering Graduate (NDSEG) Fellowship to AHC.

Figure Legends

Figure 1. Illustration of the alignment of two reads by identifying a common k -mer in the overlap. The boxes labeled by K denote a common k -mer between the two reads. The symbol $\times$ denotes a sequencing error that would preclude a common k -mer from occurring in the immediate area. However, as long as the error rate is low enough and the overlap is sufficiently long, a common k -mer will exist with high probability.

Figure 2. Illustration of the empirical coverage distribution compared to the Poisson distribution. The empirical distribution is drawn with a dashed-line and the Poisson distribution is drawn with a solid line. The total variation distance is the shaded area between the two distributions divided by 2. ECHO finds the Poisson distribution minimizing the total variation distance to the empirical distribution to find the optimal set of parameters ω^* and ϵ^* .

Figure 3. Position-specific by-base error rates for 76 bp PhiX174 data D1. (a) Error rates before and after applying the three error correction algorithms for sequence coverage depth 30. Spectral Alignment (or SA; see Chaisson et al. 2009) and SHREC (Schröder et al. 2009) are able to improve the error rate in intermediate positions, but they both become less effective for later positions. In contrast, ECHO remains effective throughout the entire read length, reducing the error rate at the end of the read from about 5% to under 1%. (b) Error rates before and after running ECHO with varying coverage depths. ECHO's ability to correct sequencing errors improves as the sequence coverage depth increases. A coverage depth of 15 seems sufficient to control the error rate throughout the entire read length.

Figure 4. The gain of ECHO and the position-specific coverage for Chromosome 1 of the yeast data D6. Each plot uses bins of 1000 bp. The top plot shows the gain of ECHO, defined as the number of corrected errors minus the number of introduced errors, divided by the number of actual errors. The bottom plot shows the position-specific coverage.

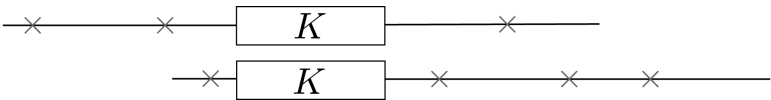


Figure 1

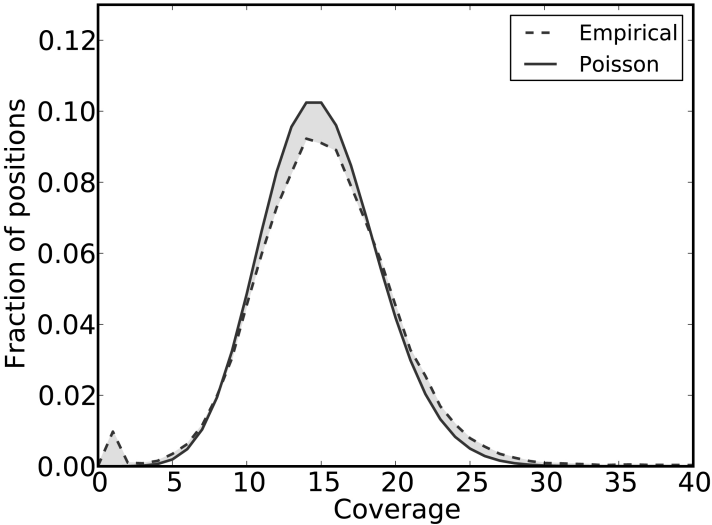
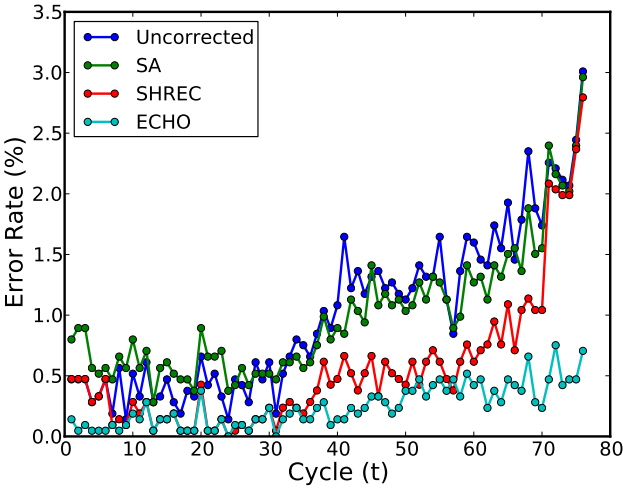
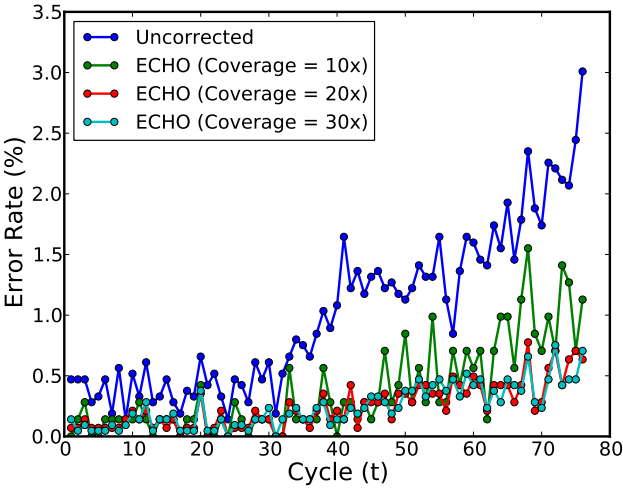


Figure 2



(a)



(b)

Figure 3

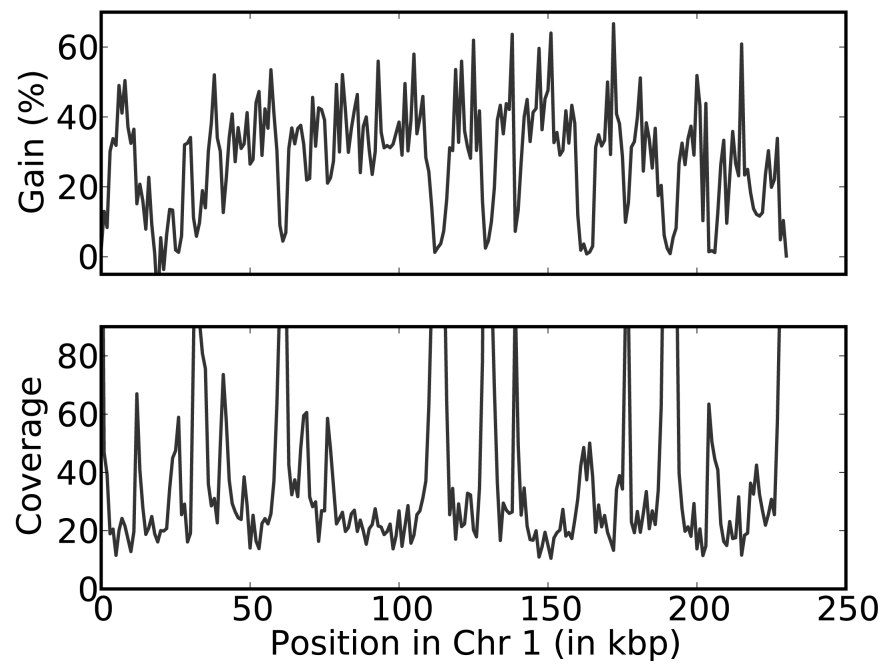


Figure 4

Table 1. Error rates for haploid data before and after running error correction.

Data	Cov	By-base error rate (%)				By-read error rate (%)			
		Uncorr	SA	SHREC	ECHO	Uncorr	SA	SHREC	ECHO
(D1) PhiX174	5	1.00	0.72	0.81	0.45	31.27	21.13	25.36	15.49
	10	0.99	0.83	0.64	0.38	29.48	18.48	21.27	10.16
	15	1.01	0.72	0.65	0.27	28.85	17.48	19.75	8.55
	20	1.00	0.81	0.66	0.25	28.00	17.35	18.53	7.97
	25	1.08	0.96	0.64	0.30	28.33	18.23	18.35	8.75
	30	1.02	1.00	0.56	0.25	28.02	17.82	17.86	8.18
(D2) <i>D. mel.</i>	13	2.93	2.93	2.57	1.15	48.85	48.85	25.68	16.99
(D3) Simulated <i>D. mel.</i>	5	0.96	0.91	0.58	0.53	15.60	15.59	15.94	11.25
	10	0.98	0.91	0.52	0.36	12.34	12.34	13.38	3.78
	15	1.00	0.92	0.50	0.21	12.52	12.51	12.74	1.84
	20	0.99	0.90	0.49	0.10	12.58	12.58	12.57	0.89
	25	1.00	0.93	0.50	0.10	12.71	12.70	12.49	0.86
	30	0.99	0.93	0.50	0.09	12.79	12.79	12.45	0.88
(D4) Simulated Human	5	1.01	1.00	0.68	0.61	26.94	15.62	18.06	12.52
	10	1.01	1.00	0.60	0.41	27.13	13.15	15.58	6.02
	15	1.00	1.01	0.53	0.28	27.08	13.15	13.07	5.53
	20	1.01	0.99	0.53	0.27	27.06	13.23	12.86	5.57
	25	1.00	0.98	0.52	0.27	26.93	13.33	12.73	5.77
	30	1.00	0.97	0.53	0.26	26.95	13.45	12.84	5.88

See Table 2 for the actual numbers of corrected and introduced errors. Cov: sequence coverage depth. Uncorr: uncorrected reads. SA: spectral alignment method used in EULER-USR (Chaisson et al. 2009). SHREC: method introduced by Schröder et al. (2009). ECHO: our error correction method. “By-base” error rate refers to the ratio of the total number of incorrect bases in the reads to the total length of the reads. “By-read” error rate corresponds to the ratio of the number of reads each with *at least* one miscalled base to the total number of reads. The read length for D1, D3, and D4 was 76 bp. The read length for D2 was 36 bp.

Table 2. Numbers of corrected errors and introduced errors after running error correction.

Data	Cov	# Corrected Errors			# Introduced Errors			Gain %		
		SA	SHREC	ECHO	SA	SHREC	ECHO	SA	SHREC	ECHO
(D1) PhiX174	5	78	43	148	2	0	0	28.1	15.9	54.8
	10	196	181	328	109	0	0	16.3	34.0	61.5
	15	300	288	605	64	0	0	28.8	35.1	73.8
	20	376	363	811	169	0	0	19.2	33.6	75.2
	25	476	509	1043	320	1	0	10.8	35.1	72.0
	30	558	618	1245	524	0	0	2.1	37.6	75.8
(D2) <i>D. mel.</i>	13	0	2931051	5098551	0	1919856	40684	0.0	12.2	61.1
(D3) Simulated <i>D. mel.</i>	5	1682	1813	3226	1401	26	0	5.8	36.9	44.9
	10	4083	4272	7701	3346	17	0	7.5	43.4	63.5
	15	6304	7220	14235	4999	11	11	8.9	47.8	78.6
	20	8317	9539	18395	6453	12	45	9.4	47.9	89.2
	25	10304	11916	22355	8577	16	22	6.9	47.7	89.6
	30	12195	14232	28012	10481	21	27	5.8	47.8	90.6
(D4) Simulated Human	5	4082	4658	5419	3870	1047	365	1.7	28.5	39.9
	10	10117	10805	15833	9834	1189	713	1.1	38.0	59.7
	15	14950	17220	27881	15120	397	830	-0.5	44.8	72.1
	20	19731	23190	37933	19013	444	886	1.4	45.1	73.4
	25	24228	28954	46191	23169	585	714	1.7	45.5	72.9
	30	29038	34912	55706	26758	722	491	3.1	45.7	73.9

These results correspond to the results in Table 1. “# Corrected Errors” is the number of errors in the reads that were corrected. “# Introduced Errors” is the number of bases in the reads that were correct but were miscorrected by the error correction. Gain is defined as $(\text{\#Corrected Errors} - \text{\#Introduced Errors})/(\text{\#Actual Errors})$.

Table 3. Error rates for the whole-genome yeast data D6.

	By-base error rate (%)				By-read error rate (%)			
	SHREC		ECHO		SHREC		ECHO	
	Before	After	Before	After	Before	After	Before	After
Entire D6	0.90	0.83	0.9	0.71	14.9	10.8	14.9	10.9
Conditional D6	0.89	0.79	1.28	0.83	14.8	0.6	17.3	8.4

SA results are not shown because it did not perform any error correction for this data set. *Conditional* refers to the subset of data on which error correction was performed by each algorithm. Specifically, for ECHO it refers to the reads that passed the coverage checks. For SHREC, it refers to the reads that did not get discarded.

Table 4. Numbers of corrected errors and introduced errors after running error correction on the whole-genome yeast data D6.

Data	# Corrected Errors		# Introduced Errors		Gain %	
	SHREC	ECHO	SHREC	ECHO	SHREC	ECHO
Entire D6	743434 out of 2302181	498588 out of 2302181	502726	12087	10.5	21.1
Conditional D6	743434 out of 2236313	498588 out of 1393181	502726	12087	10.8	35.8

SA results are not shown because it did not perform any error correction for this data set. As explained in Table 3, *Conditional* refers to the subset of data on which error correction was performed by each algorithm. Gain is defined as $(\# \text{ Corrected Errors} - \# \text{ Introduced Errors}) / (\# \text{ Actual Errors})$. In general, ECHO is more conservative than SHREC. See Figure 4 for position-specific gain information.

Table 5. Error rates for diploid data D5 before and after error correction.

Cov	By-base error rate (%)						By-read error rate (%)					
	Uncorr	SA	SHREC	ECHO with $h =$			Uncorr	SA	SHREC	ECHO with $h =$		
				0	10^{-4}	10^{-3}				0	10^{-4}	10^{-3}
10	1.02	1.22	0.58	0.40	0.35	0.34	27.58	13.15	15.45	5.47	5.30	4.83
20	1.02	1.12	0.54	0.21	0.16	0.16	27.22	12.83	13.27	2.84	2.02	1.88
30	1.01	1.10	0.52	0.10	0.08	0.08	27.17	13.02	13.01	2.04	1.06	1.10
40	1.00	1.01	0.50	0.10	0.07	0.08	27.12	13.21	12.76	1.99	1.16	1.17

h : prior probability of heterozygosity used in ECHO. See the caption of Table 1 for a full description of notation. In computing the error rate of ECHO for $h > 0$, we compared the genotype assigned by ECHO to each short-read with the true genotype sequence in the diploid genome from which the short-read originated. Errors in SA, SHREC, and ECHO for $h = 0$ were computed in the usual way by comparing with the true haplotype. The read length was 76 bp.

Table 6. Heterozygous site detection efficiency for simulated diploid data D5.

h	Cov	Unfiltered		Quality ≥ 20		Quality ≥ 30		Quality ≥ 40	
		Precision	Recall	Precision	Recall	Precision	Recall	Precision	Recall
10^{-3}	10	0.97	0.77	0.99	0.59	0.99	0.48	1.00	0.37
	20	0.90	0.96	0.92	0.93	0.93	0.90	0.95	0.88
	30	0.82	0.99	0.85	0.99	0.86	0.99	0.88	0.97
	40	0.83	0.99	0.86	0.99	0.88	0.99	0.88	0.99
	50	0.84	0.99	0.86	0.99	0.87	0.99	0.88	0.98
	60	0.85	0.98	0.87	0.98	0.88	0.97	0.89	0.97
10^{-4}	10	0.99	0.69	0.99	0.48	1.00	0.37	1.00	0.27
	20	0.91	0.94	0.93	0.90	0.95	0.88	0.95	0.83
	30	0.84	0.99	0.86	0.99	0.88	0.97	0.89	0.96
	40	0.84	0.99	0.87	0.99	0.88	0.99	0.89	0.98
	50	0.85	0.99	0.87	0.99	0.88	0.98	0.89	0.98
	60	0.86	0.98	0.88	0.97	0.89	0.97	0.89	0.97

The precision and recall of the genotypes called by ECHO when modeling a diploid genome are shown. Precision is the number of true positives over the sum of true positives and false positives. Recall is the number of true positives divided by the number of positives. A true positive is a correctly identified heterozygous site. A false positive is a homozygous site mistakenly classified as a heterozygous site. The number of positives is the number of actual heterozygous sites. Every genotype called by ECHO is given a quality score. The “Unfiltered” column shows the results when all called genotypes are considered. “Quality $\geq x$ ” shows the results when only genotypes with quality scores greater than x are considered.

Table 7. *De novo* assembly results for uncorrected and corrected PhiX174 data D1.

Coverage	Reads	N50 (bp)	Max (bp)
5	Uncorrected	219.4	401.5
	SA	238.0	422.4
	SHREC	148.5	246.2
	ECHO	261.2	475.7
10	Uncorrected	841.2	1452.4
	SA	1005.7	1629.6
	SHREC	608.6	1017.9
	ECHO	1246.4	1833.7
15	Uncorrected	2561.4	2704.7
	SA	3095.8	3126.9
	SHREC	2014.5	2369.8
	ECHO	3345.6	3386.7
20	Uncorrected	3725.9	3773.3
	SA	4551.3	4569.5
	SHREC	4179.2	4179.2
	ECHO	4706.7	4706.7
25	Uncorrected	4747.5	4747.5
	SA	5208.6	5208.6
	SHREC	5215.6	5215.6
	ECHO	5335.7	5335.7
30	Uncorrected	5224.1	5224.1
	SA	5337.1	5337.1
	SHREC	4855.5	4855.5
	ECHO	5337.1	5337.1

The length of the PhiX174 genome is 5386 bp. This experiment was repeated 20 times, and Velvet was used to carry out the assembly in each case. Max: the maximum contig length. N50: a statistic commonly used to assess the quality of *de novo* assembly. It is computed by sorting all the contigs by their size in decreasing order and adding the lengths of these contigs until the sum is greater than 50% of the total length of all the contigs. The length of the last added contig corresponds to the N50 value. A larger N50 indicates a better assembly. For nearly all sequence coverage depths, the largest Max and N50 values were observed for the assembly carried out using ECHO-corrected reads.

Table 8. *De novo* assembly results for simulated data D3 from a 5 Mbp region of *D. melanogaster*.

Coverage	Reads	N50 (bp)	Max (bp)	# Contig1000	SeqCov1000	Error1000
10	Uncorrected	840.0	4796.0	1400.5	41.5%	0.009%
	SA	1283.0	6514.5	1835.0	63.9%	0.005%
	SHREC	996.0	5315.5	1625.0	50.8%	0.094%
	ECHO	1194.5	7338.5	1798.0	60.8%	0.006%
	Perfect	1721.5	10044.0	1871.5	75.4%	0.000%
15	Uncorrected	4476.0	20978.5	1293.5	95.0%	0.004%
	SA	9732.0	40215.5	740.0	98.8%	0.001%
	SHREC	7838.0	40424.0	866.0	98.2%	0.015%
	ECHO	10698.0	40739.0	669.0	99.0%	0.001%
	Perfect	13847.0	48053.5	532.5	99.4%	0.000%
20	Uncorrected	29377.0	96954.0	291.5	99.8%	0.001%
	SA	80231.0	233239.0	107.0	100.0%	0.000%
	SHREC	80847.5	225311.0	104.0	100.0%	0.001%
	ECHO	95977.5	242397.0	81.0	100.0%	0.000%
	Perfect	114364.0	312355.0	67.0	100.0%	0.000%
25	Uncorrected	158180.5	483918.5	53.0	100.0%	0.000%
	SA	799798.5	1232231.5	13.0	100.0%	0.000%
	SHREC	951499.0	1260863.5	11.5	100.0%	0.000%
	ECHO	931249.0	1260863.5	10.5	100.0%	0.000%
	Perfect	951499.0	1391995.0	8.5	100.0%	0.000%
30	Uncorrected	895733.5	1399461.5	8.0	100.0%	0.000%
	SA	3598579.0	3598579.0	2.5	100.0%	0.000%
	SHREC	4266643.0	4266643.0	1.5	100.0%	0.000%
	ECHO	4266643.0	4266643.0	1.5	100.0%	0.000%
	Perfect	4266643.0	4266643.0	1.5	100.0%	0.000%

Assembly was performed using Velvet with $k = 39$ and automatic coverage cutoff. Uncorrected reads are the original simulated reads. To correct for sequencing errors, we used SA, SHREC, and ECHO prior to assembly. Perfect reads correspond to the error-free reads from which uncorrected reads were simulated. Max and N50 are as explained in Table 7. SeqCov1000 is the percentage of the genome covered by contigs of length ≥ 1000 bp. Error1000 denotes the percentage of the total number of assembly errors (measured by edit distance from the true genome, with a penalty of 1 for mismatch, insertion, or deletion) in those contigs with length ≥ 1000 bp, with respect to the total length of those contigs.

Table 9. Comparison of running times, in minutes, for simulated data with 76 bp reads.

Method	N , Number of reads in the input data				
	10^4	10^5	10^6	2.5×10^6	5×10^6
SA	0.03	0.36	4.59	12.44	26.69
SHREC	0.19	1.48	17.00	39.52	90.64
ECHO	(1.52) 0.67	(7.48) 5.37	(19.97) 56.03	(42.55) 143.95	(78.2) 300.77

The experiments were performed on a Mac Pro with two quad-core 3.0 GHz Intel Xeon processors and 14 GB of RAM. SA and ECHO each used only a single core, whereas SHREC was allowed to use all 8 cores. For ECHO, the numbers shown inside the parentheses correspond to the running times of parameter selection.

References

- Batzoglou, S., Jaffe, D. B., Stanley, K., Butler, J., Gnerre, S., Mauceli, E., Berger, B., Mesirov, J. P., and Lander, E. S., 2002. Arachne: a whole-genome shotgun assembler. *Genome Research*, **12**(1):177–189.
- Birol, I., Jackman, S., Nielsen, C., Qian, J., Varhol, R., Stazyk, G., Morin, R., Zhao, Y., Hirst, M., Schein, J., *et al.*, 2009. De novo transcriptome assembly with ABySS. *Bioinformatics*, **25**(21):2872–2877.
- Brockman, W., Alvarez, P., Young, S., Garber, M., Giannoukos, G., Lee, W. L., Russ, C., Lander, E. S., Nusbaum, C., and Jaffe, D. B., *et al.*, 2008. Quality scores and SNP detection in sequencing-by-synthesis systems. *Genome Research*, **18**:763–770.
- Butler, J., MacCallum, I., Kleber, M., Shlyakhter, I. A., Belmonte, M. K., Lander, E. S., Nusbaum, C., and Jaffe, D. B., 2008. ALLPATHS: De novo assembly of whole-genome shotgun microreads. *Genome Research*, **18**(5):810–820.
- Chaisson, M., Pevzner, P., and Tang, H., 2004. Fragment assembly with short reads. *Bioinformatics*, **20**(13):2067–74.
- Chaisson, M. J. P., Brinza, D., and Pevzner, P. A., 2009. De novo fragment assembly with short mate-paired reads: Does the read length matter? *Genome research*, **19**(2):336–346.
- Chin, F., Leung, H., Li, W.-L., and Yiu, S.-M., 2009. Finding optimal threshold for correction error reads in DNA assembling. *BMC Bioinformatics*, **10**(Suppl 1):S15.
- Erlich, Y., Mitra, P., Delabastide, M., McCombie, W., and Hannon, G., 2008. Alta-Cyclic: a self-optimizing base caller for next-generation sequencing. *Nature Methods*, **5**:679–682.
- Ewing, B. and Green, P., 1998. Base-calling of automated sequencer traces using Phred. II. Error probabilities. *Genome Research*, **8**(3):186–194.
- Gajer, P., Schatz, M., and Salzberg, S. L., 2004. Automated correction of genome sequence errors. *Nucleic Acids Research*, **32**(2):562–569.
- Gresham, D., Desai, M. M., Tucker, C. M., Jenq, H. T., Pai, D. A., Ward, A., DeSevo, C. G., Botstein, D., and Dunham, M. J., 2008. The repertoire and dynamics of evolutionary adaptations to controlled nutrient-limited environments in yeast. *PLoS Genetics*, **4**(12).
- Hellmann, I., Mang, Y., Gu, Z., Li, P., Vega, F. M. D. L., Clark, A. G., and Nielsen, R., 2008. Population genetic analysis of shotgun assemblies of genomic sequences from multiple individuals. *Genome Research*, **18**(7):1020–9.
- Jiang, R., Tavaré, S., and Marjoram, P., 2009. Population genetic inference from resequencing data. *Genetics*, **181**(1):187–97.
- Jokinen, P. and Ukkonen, E., 1991. Two algorithms for approximate string matching in static texts. *Proceedings of Mathematical Foundations of Computer Science*, **520**:240–248.

- Kao, W.-C. and Song, Y., 2010. naiveBayesCall: An efficient model-based base-calling algorithm for high-throughput sequencing. *Proc. 14th Annual International Conference on Research in Computational Molecular Biology (RECOMB)*, **6044**:233–247.
- Kao, W. C., Stevens, K., and Song, Y. S., 2009. BayesCall: A model-based basecalling algorithm for high-throughput short-read sequencing. *Genome Research*, **19**(10):1884–1895.
- Kircher, M., Stenzel, U., and Kelso, J., 2009. Improved base calling for the Illumina Genome Analyzer using machine learning strategies. *Genome Biology*, **10**(8):R83.
- Langmead, B., Trapnell, C., Pop, M., and Salzberg, S., 2009. Ultrafast and memory-efficient alignment of short DNA sequences to the human genome. *Genome Biology*, **10**(3):R25.
- Li, H. and Durbin, R., 2010. Fast and accurate long-read alignment with Burrows-Wheeler transform. *Bioinformatics*, **26**(5):589–595.
- Li, H., Ruan, J., and Durbin, R., 2008. Mapping short DNA sequencing reads and calling variants using mapping quality scores. *Genome Research*, **18**(11):1851–8.
- Martin, J., Han, C., Gordon, L. A., Terry, A., Prabhakar, S., She, X., Xie, G., Hellsten, U., Chan, Y. M., Altherr, M., *et al.*, 2004. The sequence and analysis of duplication-rich chromosome 16. *Nature*, **432**(7020):988–94.
- Medvedev, P. and Brudno, M., 2008. Ab Initio whole genome shotgun assembly with mated short reads. In *Proceedings of the 12th Annual Research in Computational Biology Conference (RECOMB)*, pages 50–64.
- Metzker, M., 2010. Sequencing technologies—the next generation. *Nature Reviews Genetics*, **11**(1):31–46.
- Peng, Y., Leung, H., Yiu, S., and Chin, F., 2010. IDBA - a practical iterative de Bruijn graph de novo assembler. In *Proc. 14th Annual International Conference on Research in Computational Molecular Biology (RECOMB)*, volume 6044 of *Lecture Notes in Computer Science*, pages 426–440–440.
- Pevzner, P. A., Tang, H., and Waterman, M. S., 2001. An Eulerian path approach to DNA fragment assembly. *Proc Natl Acad Sci*, **98**(17):9748–53.
- Qu, W., Hashimoto, S.-i., and Morishita, S., 2009. Efficient frequency-based de novo short-read clustering for error trimming in next-generation sequencing. *Genome Research*, **19**(7):1309–1315.
- Rasmussen, K. R., Stoye, J., and Myers, E. W., 2005. Efficient q-gram filters for finding all epsilon-matches over a given length. *Proc. 9th Annual International Conference on Research in Computational Molecular Biology (RECOMB)*, **3678**:189–203.
- Rougemont, J., Amzallag, A., Iseli, C., Farinelli, L., Xenarios, I., and Naef, F., 2008. Probabilistic base calling of Solexa sequencing data. *BMC Bioinformatics*, **9**:431.
- Salmela, L., 2010. Correction of sequencing errors in a mixed set of reads. *Bioinformatics*, **26**(10):1284–1290.
- Sanger, F., Nicklen, S., and Coulson, A., 1977. DNA sequencing with chain-terminating inhibitors. *Proc. Nat. Acad. Sci.*, **74**(12):5463–5467.

- Schröder, J., Schröder, H., Puglisi, S. J., Sinha, R., and Schmidt, B., 2009. SHREC: a short-read error correction method. *Bioinformatics*, **25**(17):2157–2163.
- Shi, H., Schmidt, B., Liu, W., and Muller-Wittig, W., 2009. Accelerating error correction in high-throughput short-read DNA sequencing data with CUDA. In *Proc. IEEE International Parallel & Distributed Processing Symposium*, pages 1–8.
- Simpson, J. T., Wong, K., Jackman, S. D., Schein, J. E., Jones, S. J., and Birol, I., 2009. ABySS: a parallel assembler for short read sequence data. *Genome Research*, **19**(6):1117–1123.
- Sundquist, A., Ronaghi, M., Tang, H., Pevzner, P., and Batzoglou, S., 2007. Whole-genome sequencing and assembly with high-throughput, short-read technologies. *PLoS One*, **2**(5):e484.
- Whiteford, N., Skelly, T., Curtis, C., Ritchie, M., Lohr, A., Zaranek, A., Abnizova, I., and Brown, C., 2009. Swift: Primary Data Analysis for the Illumina Solexa Sequencing Platform. *Bioinformatics*, **25**(17):2194–2199.
- Wijaya, E., Frith, M. C., Suzuki, Y., and Horton, P., 2009. Recount: expectation maximization based error correction tool for next generation sequencing data. *International Conference on Genome Informatics*, **23**(1):189–201.
- Yang, X., Dorman, K. S., and Aluru, S., 2010. Reptile: representative tiling for short read error correction. *Bioinformatics*, **26**(20):2526–2533.
- Zerbino, D. R. and Birney, E., 2008. Velvet: Algorithms for de novo short read assembly using de Bruijn graphs. *Genome Research*, **18**(5):821–829.