



Objective review of *de novo* stand-alone error correction methods for NGS data

Andy S. Alic,^{1*} David Ruzafa,² Joaquin Dopazo^{3,4,5} and Ignacio Blanquer^{1,6}

The sequencing market has increased steadily over the last few years, with different approaches to read DNA information prone to different types of errors. Multiple studies demonstrated the impact of sequencing errors on different applications of next-generation sequencing (NGS), making error correction a fundamental initial step. Different methods in the literature use different approaches and fit different types of problems. We analyzed 50 methods divided into five main approaches (k-spectrum, suffix arrays, multiple-sequence alignment, read clustering, and probabilistic models). They are not published as a part of a suite (stand-alone), and target raw, unprocessed data without an existing reference genome (*de novo*). These correctors handle one or more sequencing technologies using the same or different approaches. They face general challenges (sometimes with specific traits for specific technologies) such as repetitive regions, uncalled bases, and ploidy. Even assessing their performance is a challenge in itself because of the approach taken by various authors, the unknown factor (*de novo*), and the behavior of the third-party tools employed in the benchmarks. This study aims to help the researcher in the field to advance the field of error correction, the educator to have a brief but comprehensive companion, and the bioinformatician to choose the right tool for the right job. © 2016 John Wiley & Sons, Ltd

How to cite this article:

WIREs Comput Mol Sci 2016, 6:111–146. doi: 10.1002/wcms.1239

INTRODUCTION

Next-generation sequencing (NGS) appeared in 2005, and since then, its market has increased steadily, with various technologies being developed. NGS has evolved faster than Moore's law in computer science, allowing us to sequence and assemble large genomes like the Loblolly Pine with 22 Gb¹ or the Norway Spruce with 20 Gb² for a reasonable cost in time and resources. However, there are many other species (e.g., the Amoeba Dubia with a 670 Gb estimated genome size,³ 200× human genome's size) that are still challenging to assemble. The errors introduced by the sequencing process are one of the main reasons NGS data has to be corrected before any further use. Multiple studies have demonstrated the impact of sequencing errors on different applications of NGS, making error correction a fundamental initial step.^{4–7} There are many error correction tools

*Correspondence to: asalic@posgrado.upv.es

¹Institute of Instrumentation for Molecular Imaging (I3M), Universitat Politècnica de València, Valencia, Spain

²Departamento de Química Física e Instituto de Biotecnología, Facultad de Ciencias, Universidad de Granada, Granada, Spain

³Department of Computational Genomics, Príncipe Felipe Research Centre (CIPF), Valencia, Spain

⁴CIBER de Enfermedades Raras (CIBERER), Valencia, Spain

⁵Functional Genomics Node (INB) at CIPF, Valencia, Spain

⁶Biomedical Imaging Research Group GIBI 2³⁰, Polytechnic University Hospital La Fe, Valencia, Spain

Conflict of interest: The authors have declared no conflicts of interest for this article.

Additional Supporting Information may be found in the online version of this article.

in the literature that cope with different technologies and error types. However, to our knowledge, there is no complete, objective review of the modern methods that could help researchers, educators, and users at the same time. There are benchmarks summarizing a number of methods, but there is none extensively focusing on the implementation, features, and the overall domain (including challenges). Our work synthesizes 50 *de novo* stand-alone error-correction software. The Supporting Information includes the description of the approach used to search the literature along with the inclusion criteria.

The article continues with the motivation (also containing a brief description of the sequencing technologies and various error sources) followed by a presentation of the correctors. Next, the 'Discussion' section presents some important points related to challenges faced by correctors and how their performance is assessed in the literature. Finally, we conclude our paper with some general remarks about the current state of the field of the NGS error correction.

We also introduce the concept of gradual recommendations. The recommendations are gradual because they progress with the text, and each one is based on the previous information. The 'Error Correction Software' section includes general recommendations based on the features presented in Table 2. The recommendations from the 'Error correction in real projects' section use the previous ones as a foundation and extend the suggestions now that the reader has read about some real-world examples. The subsection 'Recommendations' from the section 'Challenges' focuses on proposals, taking into account the challenges that the correctors must address. Finally, the subsection 'Recommendations' from the section 'Testing' offers advice (now that the reader knows the methods, where these have been used, and what the challenges are) based on real-world performances using different metrics.

MOTIVATION

The market size of NGS was estimated at \$2.5 billion in 2014.⁸ Furthermore, Illumina (San Diego, CA) managed to lower the cost of sequencing with its HiSEQ X to ~\$1000 (in 2015) for the human genome.^{9,10} This price is quite an achievement when considering that not so long ago (2000–2003), the draft of the human genome costed about \$300 million.¹¹ Overall, NGS has become widely used by the medical and scientific community not only for basic biological research but also in numerous applied

fields such as medical diagnosis, forensic biology, virology, and biotechnology. These are just a few clear proofs of the increasing importance of NGS in the world (not mentioning the increase in size of the reads, faster sequencing machines, and improved quality of the generated data). This quickly evolving and advancing environment facilitated the development of a myriad of methods with different applications for the NGS data. One of the most important steps (usually the first) is the correction of errors, yielding many benefits for the ulterior ones as demonstrated in the subsection 'Benefits of Error Correction'. Our readers may assume that an easy way to deal with the errors is to increase the coverage (i.e., add more sequencing data). While the increase in coverage indeed helps the correction process, there are still many challenges that the correctors must address (especially in *de novo* sequencing). Furthermore, an increase in coverage comes with an increase in costs, sequencing/processing time, and storage requirements.

After an extensive literature search (see approach and details in the Supporting Information), we selected 50 correctors. As one may expect, there is a tremendous amount of information scattered across these papers. We strive to summarize the deluge of information for an audience from many fields such as bioinformatics, biology, chemistry, computer science, and others with an interest in NGS. Our aim is to help the researcher in the field of error correction by grouping the information and synthesizing the existing work. Secondly, our work also comes in handy for educators because it summarizes and presents the key points of the information found in the selected articles. We tried to present the information gradually, without an abrupt and direct presentation of the correctors (our readers are not expected to have *a priori*, in depth knowledge of the domain). Finally, the actual users of the correction software will find Table 5 useful when searching for the right tool for their specific requirements. The Supporting Information contains an additional table with all the testing results from the reviewed articles.

Sequencing Technologies

DNA sequencing was started in 1977 with the publication of the Sanger method.¹² This method uses a large amount of DNA as a template for each read and needs an independent polymerase chain reaction (PCR) for each possible nucleotide. The PCRs are produced in the presence of four deoxynucleotides and a single dideoxynucleotide, which stops the elongation. Once synthesized, the truncated DNAs are

resolved by electrophoresis. During the synthesis reaction, a radioactive nucleotide (usually dATP - Deoxyadenosine triphosphate) that simplifies the determination of the sequence is incorporated into the elongating strands.

NGS methods are more efficient than Sanger sequencing in two different ways. On the one hand, in Sanger sequencing, only 1 Kb (maximum) can be sequenced in a single experiment, whereas NGS is parallel by definition, allowing a throughput of hundreds/thousands of gigabases per run. Note that in this article, Kb, Mb, and Gb are the acronyms for kilobases, megabases, and gigabases, respectively. On the other hand, the chemical reactions are usually combined with the signal detection in some versions of NGS, whereas in Sanger sequencing, they are two separate processes. Factors like the reduction of time, manpower, and reagents in NGS lead to lower costs, making it possible to do more repeats than with the Sanger method. This results in a more accurate, reliable sequencing and better coverage. In NGS, the first step is the DNA cleavage into short segments (or reads) with lengths depending on the particular sequencer used.

In this review, we focus on NGS technologies based on sequencing by synthesis (SBS), using DNA polymerase/ligase enzymes to generate a complementary strand. As defined by,¹³ Pacific Biosciences is the only SBS approach that has a real-time sequencing strategy. All the others are synchronously controlled as we shall see in the following subsections where we present a brief but comprehensive description of the sequencing chemistries of the five aforementioned technologies. Fuller et al. also divide the methods in single molecule-based (Pacific Biosciences and Oxford Nanopore - not specified in Ref 13) and ensemble-based (Illumina, Roche 454, Ion Torrent - not specified in Ref 13 and SOLiD). The former sequences single molecules of DNA as they are obtained from the source, while the latter relies on the amplification (cloning) of DNA segments before starting the actual sequencing process.

The sequencing technologies explicitly supported by the methods from our review (ordered by the number of correctors supporting them) are Illumina, Roche 454, Pacific Biosciences, Ion Torrent, Oxford Nanopore, and SOLiD. Note that we placed strong emphasis on the actual DNA 'reading' step because this is the main step where sequencing errors are generated. For further and detailed information regarding the platforms and the entire sequencing process, please check the 'Further Reading' section where we listed some resources that cover the themes in a more in depth manner. Additionally, the

interested reader can find more details about the chemistry of SBS sequencing in.¹³

Table 1 offers some information about a number of well-known sequencers.

Illumina/Solexa

The Illumina reads receive adapters at their ends. These adapters attach themselves to their respective complementary adapters, with the latter hooked on a board with many variants of (complementary) adapters placed on a solid surface. Next, a segment is cloned by PCR amplification, creating a spot with many copies of the same initial read. The last step before the actual sequencing splits each read in the two complementary strands. Once the board contains only the single-stranded reads held in place by adapters, fluorescent-labeled, terminated nucleotides and DNA polymerase are added as a mix on the board (Figure 1, Step 1). Fluorescent bases produce unique colors for each matching base. A polymerase is a protein that rebuilds the double helix starting from a single-stranded template. It adds the complementary base for each of the templates' composing nucleotides. Due to the terminated property of the free nucleotides added earlier in the mix, the polymerase attach to one and only one base per cycle (Figure 1, Step 2). The sequencer registers the color of the latest incorporated nucleotide for each read by taking a snapshot of the board (Figure 1, Step 3). The process continues with the elimination of the terminator with the fluorescent label and the starting of a new cycle. The number of cycles gives the length of the read, with all reads normally having the same length. Using the snapshots, the sequencer determines the nucleotides composing a read (Figure 1, Step 4).

Roche 454

As in Illumina's method, the 454 reads pass through a PCR amplification step and bind to adapters for which the complementaries lay hooked on a bead. Roche 454 uses the same fluorescent signaling to read the attached nucleotides. Therefore, the addition of each nucleotide releases a light signal. The main difference consists in the approach taken at each cycle. Instead of adding a solution containing all four fluorescent and terminated bases, the sequencer adds the solution with one and only one type of base without the terminator. As a result, a variable number of bases can bind on a read at each cycle. The intensity of the signal represents the number of nucleotides added at each cycle. Roche tries to reconstruct entire homopolymers (runs of identical bases) at each cycle to save time. As an example, at Step 2 from Figure 1, the sequencer adds two cytosines (instead of one like

TABLE 1 | Information (as of November 2015) About Sequencing Machines as Reported by the Vendors Themselves

Platform	Instrument	Unit	Reads/Unit (single reads)	Avg./Max Read Len. (bp)	Read Type	Quality	Error Type	Info
Illumina	HiSeq X Ten	Lane	375,000,000	150/150	PE	≥75% bp > Q30	mismatch	15
	HiSeq 3000/4000	Lane	312,500,000	150/150	PE	≥75% bp > Q30	mismatch	16
	HiSeq NextSeq 500	Run	400,000,000	150/150	SR/PE	≥75% bp > Q30	mismatch	17
	High-Output	Run	130,000,000	150/150	PE	≥75% bp > Q30	mismatch	17
	HiSeq NextSeq 500	Run	130,000,000	150/150	PE	≥75% bp > Q30	mismatch	17
Illumina	Mid-Output	Run	130,000,000	150/150	PE	≥75% bp > Q30	mismatch	17
	HiSeq 2500 High-Output	Lane	250,000,000	125/125	SR/PE	≥80% bp > Q30	mismatch	18
	HiSeq SBS V4	Lane	250,000,000	125/125	SR/PE	≥80% bp > Q30	mismatch	18
	HiSeq 2500 High-Output	Lane	186,048,000	100/100	SR/PE	≥80% bp > Q30	mismatch	18
	TRUSEQ SBS V3	Lane	150,696,000	250/250	SR/PE	≥75% bp > Q30	mismatch	19
	HiSeq 2500 Rapid Run	Lane	93,024,000	100/100	SR/PE	≥80% bp > Q30	mismatch	19
	HiScanSQ	Lane	42,075,000	150/150	SR/PE	≥80% bp > Q30	mismatch	20
	Genome Analyzer Ix	Lane	42,075,000	150/150	SR/PE	≥80% bp > Q30	mismatch	21
	Genome Analyzer Iie	Lane	42,075,000	150/150	SR/PE	≥80% bp > Q30	mismatch	22
	Genome Analyzer II	Flow Cell	50,000,000	75/75	SR/PE	≥70% bp > Q30	mismatch	23
Thermo	Genome Analyzer	Flow Cell	26,000,000	50/50	SR/PE	≥98.5% bp error-free	mismatch	24
	MiSeq Reagent Kit v3	Lane	25,000,000	300/300	SR/PE	>75% bp > Q30	mismatch	24
	MiSeq Reagent Kit v2	Lane	15,000,000	250/250	SR/PE	>70% bp > Q30	mismatch	24
	Ion Proton I	Chip	80,000,000	125 ¹ /200	SR	NA	indel	25
	Ion PGM 318	Chip	4,000,000	363 ¹ /400	SR	>99% aligned/measured accuracy	indel	26
	Ion PGM 316	Chip	2,000,000	333 ¹ /400	SR	>99% aligned/measured accuracy	indel	26
	Ion PGM 314	Chip	400,000	181 ¹ /400	SR	>99% aligned/measured accuracy	indel	26

TABLE 1 | Continued

Platform	Instrument	Unit	Reads/Unit (single reads)	Avg./Max Read Len. (bp)	Read Type	Quality	Error Type	Info
Fisher Scientific	SOLID 5500xl W	Lane	3,200,000,000	75/75	SR/PE/MP	NA	mismatch	27
	SOLID 5500 W	Lane	1,600,000,000	75/75	SR/PE/MP	NA	mismatch	27
	SOLID 5500	Lane	120,000,000	75/75	SR/PE/MP	up to 99.99% \geq QV40	mismatch	28,29
	SOLID 5500xl	Lane	200,000,000	75/75	SR/PE/MP	up to 99.99% \geq QV40	mismatch	28,29
PacBio	RS II	SMRT Cell	55,000	18,181 ¹ /60,000	SR	>99.999% (QV50)	indel	30
	RS	SMRT Cell	22,375	4,575/20,848	SR	NA	indel	31
	GS FLX							
Roche 454	Titanium XL+	Run	1,000,000	700/1000	SR	99.997% Consensus accuracy	indel	32
	GS FLX							
	Titanium XLR70	Run	1,000,000	450/600	SR	99.995% Consensus accuracy	indel	32
	Junior	Run	100,000	400/- ²	SR	99% accuracy at 400 bp	indel	33
Oxford Nanopore	Junior+	Run	100,000	700/- ²	SR	99% accuracy at 700 bp	indel	34
	Minton	Run	4,400,000	9,545 ^{1,3} /300,000	1D/2D	96% accuracy base calling	indel	35
	PromethION	Run	26,000,000	9,846 ^{1,3} /300,000	1D/2D	96% accuracy base calling	indel	35

SR stands for single read, PE stands for pair-end reads, MP stands for mate-paired reads, 1D stands for a segmented read in one direction, and 2D stands for a segment read both forward and reverse; Q or QV followed by a number represents the quality score for a base (depending on the reference the score can be in Phred¹⁴ format or not).

¹ Calculated from maximum throughput divided by maximum number of reads.

² Unspecified max length.

³ Values from Fast Mode.

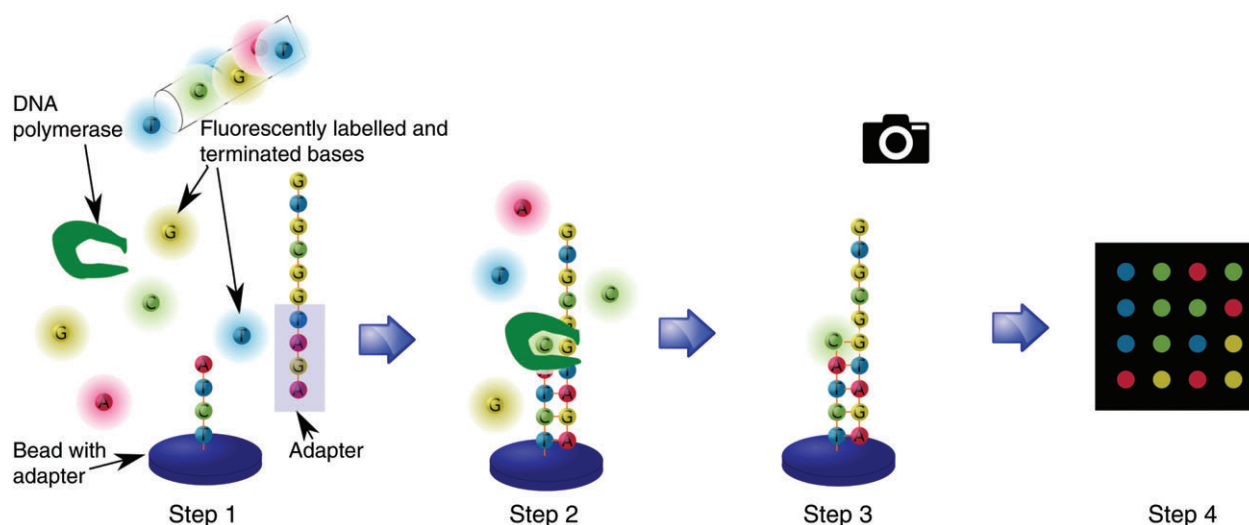


FIGURE 1 | Main sequencing steps for Illumina.

in Illumina's case) in the depicted cycle. Generally, the sequences generated by 454 instruments have different lengths because different numbers of bases are incorporated at each cycle.

Ion Torrent/Personal Genome Machine (PGM)

Unlike Illumina and 454, Ion Torrent sequencing is not based on the detection of optical signals. Instead, it takes advantage of the release of protons (H^+) following the addition of deoxynucleotides to the DNA strands by the DNA polymerases. The fluctuation in the pH of the solution can be easily measured and, using its level of acidity, the instrument can determine how many bases have been attached in each cycle. The bases are identified as in the case of 454.

Applied Biosystems Instruments (ABI) Sequencing by Oligonucleotide Ligation and Detection (SOLiD)

In this case, the reads are used to prepare clonal bead populations.³⁶ Instead of binding one nucleotide or a homopolymer per cycle (like the previously described sequencers), ABI uses fluorescent-labeled dinucleotide probes. Instead of individual bases, SOLiD encodes the transition between bases. At each ligation step, four DNA primers attached to different fluorescent dyes (out of the 16 DNA possibilities) are added to the reads that match the complementary DNA primers on the bead. Next, the fluorescent part is read and afterwards cleaved from the probe. The sequencer repeats this cycle of ligation/reading/cleavage as many times as needed in order to obtain a read of a certain length. In each cycle, two positions

of every five are determined. Once the sequencer has executed enough cycles, it resets the template with the primer going one position back by removing one base of the primer. In order to determine the complete sequence of the read, SOLiD sequencers perform this resetting step five times. As the primer is moved one base back, the sequencer reads each base twice, improving the robustness.

Pacific Biosciences

Pacific Biosciences uses a single-molecule real-time (SMRT) sequencing approach.³⁷ It employs the same fluorescent labeling as the previous technologies, but instead of executing cycles of incorporating nucleotides and taking snapshots, it detects the signals in real time as they are emitted when the incorporations occur (using a zero-mode waveguide system³⁸). Like Illumina, Pacific Biosciences uses all four bases at the same time floating in the mix. It has a bead with many wells that have a diameter between 70 and 100 nm, lower than the wavelength of the visible light. Due to the physical properties of the fluorescent additives, light is needed in order to make the fluorescent dye glow. Owing to this requirement, the bottoms of the wells are illuminated, but due to their very small diameter, the light intensity decays exponentially along the wells, creating a shadow zone. The wells have a DNA polymerase attached to their bottoms (the illuminated zone) that rebuilds the DNA complementary strand of the DNA segments floating in the mix. Each time a nucleotide is added, the fluorescent dye is cleaved. As a result, two consecutive signals do not overlap because the sensors only record the noncleaved fluorescent dyes as the

previous ones move up in the well into the shadow zone. This approach does not require cycles because each polymerase works independently of the others.

Oxford Nanopore

As its name suggests, this sequencing technology uses very small nanopores (allowing one nucleotide at a time) to read the DNA sequences.³⁹ The idea of using nanopores to decipher the DNA's code has been around since 1989, but it was not viable until 2010 when a DNA polymerase capable of attracting the DNA to the nanopore was discovered.⁴⁰ The main components of this technology are the protein nanopores resembling those found on the cellular membrane. These structures are inserted into an electrical-resistant artificial membrane onto which an electrical potential is applied. Owing to the flowing of the potential only through the aperture of the nanopore, any molecule passing through generates a variation in the current, resulting in a specific signature. Using the previously described processes, the sequencer is able to decode the DNA (or RNA or proteins). In order to get the DNA segments to pass through the hole, the segments are mixed with copies of a carrier enzyme. These carrier enzymes attach to the DNA strands. They are pulled to a nanopore where the DNA is unzipped (if necessary), and the resulting single strand passes through the aperture, producing variations in the potential. One interesting feature is the capability to sequence both strands of the DNA segment using the same nanopore, generating the so-called 2D reads (5'→3' and its complementary 3'→5' strands linked together). In order to do this, the DNA segment receives a hairpin structure at the end to keep the two templates together after unzipping. This way, once the first strand has passed through the hole, the complementary one is pulled through.

Errors in NGS

There are four types of basic sequencing errors: insertions, deletions, mismatches, and uncalled/unknown bases (or Ns).⁴¹ The differences in the sequencing process of the aforementioned technologies lead to different types of errors. Table 1 lists the predominant error type for each NGS technology. This constitutes an important factor when choosing the values of the parameters for the correctors. Mismatches are prevalent in Illumina and SOLiD while indels constitute the main error type in Roche 454, Ion Torrent, Pacific Biosciences, and Oxford Nanopore.^{5,42–46} More details about the types of errors in the aforementioned technologies and practical experiments appear in.^{5,44,47} The ensemble-based

methods are prone to pre-sequencing errors (generated by the library preparation method and the choice of primers),⁴⁸ unless PCR-free kits are used.^{65–67}

Owing to its one nucleotide incorporation per cycle, the Illumina sequencers avoid insertions and deletions almost entirely.^{5,44} Sleep et al.⁴⁹ describe the substitution errors for various Illumina sequencers. They found that the percentage of error increases toward the 3' end of the reads^{7,50} because of a phenomenon called dephasing/phasing. It is caused by the fact that an error generated at one cycle affects the next cycles and hence the increased number of errors toward the end.⁵² The same phenomenon is the main cause for the limited length of the reads in all ensemble-based SBS methods where some strands in a group of clones may fall behind, resulting in a desynchronization of the emissions of each clone in a group.¹³ Another important cause of sequencing errors is the cross-talk arising due to the overlap of dye emission frequencies. The Illumina Genome Analyzer[®] uses a red laser to read A and C and a green laser to read G and T. As a result, the Illumina Genome Analyzer[®] produces many substitution $G \rightarrow T$ and $C \rightarrow A$.^{7,54} Related to the previous cause for the Illumina MiSeq[®] sequencer, Schirmer et al. determined that $A \leftrightarrow C$ substitution errors appeared more often than $G \leftrightarrow T$ (red laser/filtering problem). They also studied the relation between the position of bases in a read and the quality scores for Illumina MiSeq[®] data. Generally, errors occurring between the start and the middle of the reads had much higher quality scores than those in the second half of a read. Furthermore, the authors found several 3-mer motifs usually preceding substitutions and indels, resulting from the selection of primers and library design. The estimated error rate for Illumina sequencers is between 1 and 2.5%.^{7,55,56}

Following the brief description of the Roche 454's sequencing approach, it becomes clear that, analogous to Illumina, some nucleotides are misclassified. Furthermore, the exact length of the homopolymers cannot exactly be determined each time,^{43,57} with the sequencer introducing:

- insertions (when recorded homopolymers are longer than real ones) and
- deletions (when recorded homopolymers are shorter than the real ones)

Luo et al.⁵⁸ demonstrate the relationship between homopolymers and their length, where pyrosequencing (Roche 454 FLX Titanium[®], Basel, Zurich) loses

accuracy as the length of the homopolymer increases. Gilles et al.⁵⁹ found a chemistry-related source of errors termed the CAFIE effect (carry forward and incomplete extension). Carry forward is generated by the inability to clean a well fully (unincorporated nucleotides are not removed) after a cycle. As a result, during the next base flow, nucleotides are prematurely incorporated to specific sequence combinations, hence generating noise. The incomplete extension effect appears when some DNA strands on a bead miss the nucleotides incorporation at a certain flow cycle. They must wait for the next flow cycles, but they are already out-of-phase with the other strands. The Roche 454 GS Junior[®] has an indel error rate of 0.38 per 100 bases.⁶ Gilles et al. report a mean error rate of 1.07%.

The Ion Torrent sequencing approach has indels as the dominant error type.⁴⁶ Bragg et al. also observe that insertions appear more often than deletions and that (in contrast to Illumina) indels are an order of magnitude more likely to be generated. Due to the similarity in the sequencing idea between Ion Torrent and 454, it becomes clear that homopolymers pose a problem for this technology too.⁶ The sequencing accuracy of the reads steadily decreased from their start to their end.^{6,46} Loman et al. observed that in comparison to 454 GS Junior[®], the Ion Torrent PGM[®] is less accurate when dealing with homopolymers (accuracy of 60% for homopolymers with six or more bases). Furthermore, for homopolymers shorter than two bases, insertion is the main error type, but the situation changes with the increase in length of the homopolymers where deletions become the norm.⁴⁶ For long homopolymers (more than 14 bases), Ion Torrent does not generate reads at all.⁶⁰ The same study reinforces the problem with homopolymers by mentioning the inability to predict the correct number of bases for homopolymers longer than eight bases. The observed error rate is 1.78% (all types of errors) in Ref 60, between 1.68 and 4.86% (all types of errors depending on the used kit) with 96–97% of them being homopolymers errors in Ref 46 and 1.5 indels per 100 bases in Ref 6.

Pacific Biosciences generates longer reads than other sequencing technologies, but the error rate is still high.⁶¹ The errors seem to be uniformly distributed and independent of the sequence context.⁶² The same authors and those from Ref 63 suggest that Pacific Biosciences is more susceptible to insertions than to deletions. Currently, the error rate for Pacific Biosciences is between 15% and 20%.^{63,64}

Oxford Nanopore is an emergent technology, generating long reads with a small and portable

device (the MinION^{®65}). It is still in development, but there are some publications studying the sequencing results.^{66,67} The accuracy is still low, with insertions as the predominant type of error.⁴² Goodwin et al. report a very high error rate, between 25 and 40%.

GC Content

It is widely accepted that the extreme base composition of some regions poses a problem for sequencing technologies.⁵ For example, the GC content (rich and poor regions) is often a source of bias and unevenness in coverage. The coverage is an extremely important aspect of the NGS as it is needed to process the output data successfully as we discuss in the ‘Low-Coverage Regions and Uniformity’ section. The problem is even more important as the bias can be introduced during the library preparation step, before the actual sequencing process.⁵ This holds true for ensemble-based SBS technologies where the amplification step (emulsion PCR or bridge amplification) generates (much) lower coverage on the very GC-rich and GC-poor regions.⁵ Quail et al. state that this problem appears for Ion Torrent due to its double amplification step (library and template). They managed to lower the bias by using the Kapa HiFi enzyme for the fragment amplification. Furthermore, the bias can be eliminated by using PCR-free preparation kits for Illumina,⁴⁹ Ion Torrent,⁵⁰ and 454.⁵¹

Ross et al.⁵ provide an excellent measurement of the bias caused by the GC regions. They use the genomes of four species as the correct and trusted source and compare it with the data generated by Illumina MiSeq[®], Ion Torrent PGM[®], Pacific Biosciences RS[®], and Complete Genomics[®]. Figures 3 and 4 from the aforementioned article depict the strong variation introduced by these GC-extreme zones. Pacific Biosciences sequencing seems to obtain better sequencing results because of its lack of amplification before sequencing,⁵ but bias still plagues this technology (slight but noticeable) when faced with genomes with GC-rich regions like *Staphylococcus aureus*.⁶⁰ The Pacific Biosciences RS[®], like the Illumina MiSeq[®] and Ion Torrent PGM[®], is also susceptible to dissociation of fragment ends in adapter ligation.⁵ High and low GC content seems to influence Oxford Nanopore too as the coverage is more variable than in zones with a 20–60% GC content. As a result, this extreme GC content partially motivates the lack of coverage for certain regions in the genome.⁶⁸

Benefits of Error Correction

The most important application of error correction is in the field of genome assembly where the input data

is corrected before the actual assembly. Many error correction publications include tests with assemblers and real data. Various assembly metrics demonstrate the need of error correction to generate meaningful assembly output (see subsection 'Assembly' from the section 'Testing').⁶⁹

A second application is re-sequencing, where multiple samples from an organism with an already-known genome are sequenced. The main purpose of this operation is to compare the variability among different genomes from the same species. Another purpose is the comparison of datasets from the same organism sequenced using different technologies or sample preparation procedures. Re-sequencing indirectly uses the same metrics like gain and accuracy, which compare the corrected reads against a reference genome.^{4,70}

Thirdly, the authors in Refs 71–74 stress the impact of error correction on short reads aligners. Errors are dangerous because they can cause an aligner to miss the real locus of a read in the reference genome. Furthermore, in the case of repetitive regions, a faulty read from a unique path in the genome can end up in multiple locations provided it matches the repetitive region due to the errors.

Another affected application of NGS is the detection of single nucleotide polymorphisms (SNPs). Normally, an aligner maps the reads against a reference genome to search for variants, but the errors in the reads can be misleading, increasing the total number of differences.⁷ Furthermore, as the distribution of SNPs is not uniform, a region can have a high density of SNPs. Errors have a higher impact in these areas.

Additionally, there are other steps following sequencing that can benefit from error correction (e.g., identification of copy number variation or chromosomal rearrangement).⁴ In conclusion, almost any possible operation on NGS data benefits from the corrected input. The 'Error correction in real projects' section lists many real studies that used the correctors included in this review.

ERROR CORRECTION SOFTWARE

The following subsections deal with the correctors included in this review. The exact sequencing project type (RNA/DNA; Whole Genome Sequencing/ChIP-seq/etc) destined for a corrector is not specified in most papers. The authors normally work with DNA reads. The benchmarks performed in the same articles contain only datasets from whole genome sequencing (WGS) projects. One exception is

PAGANtec,⁷⁵ which works with transcriptome assemblies.

Technology Support

Illumina is the market leader, with a 70% market share.⁸ The majority of software in our review support Illumina (and in some cases, other technologies), a fact that reinforces the status of the aforementioned company. The second major player is Roche with its 454 line of sequencers, which, despite the shut-down of its technology in 2013, is still widely used (officially supported until 2016).⁷⁶ As a matter of fact, **Karect**, one of the most recent error correctors (2015), targets indel errors from 454. We can see an increasing support of Pacific Biosciences, but all current correctors rely on an additional dataset from a different technology to perform the correction. Ion Torrent is not widely supported as of now, but as the prevalent errors for this technology are indels,⁵ the tools handling indels should also work with it. Finally, there is only one program that targets SOLiD color-space data, namely **HSHREC**.

In our review, we have found programs supporting more than one technology. Table 2 enumerates the technologies supported by the correctors (column 'Tech'). Figure 2 depicts the categories in which the correctors fall. All but one of the tools supporting only one technology work with Illumina and only target mismatch errors. **Hector** is the exception to the above rule, designed only for 454 reads, supporting indels. All Pacific Biosciences software focus only on Pacific Biosciences, but they use Illumina/454 reads for the cross-correction; therefore, they are classified in a separate group.

There are several software tools handling multiple technologies that can tackle all types of errors. Our readers can determine the support for different types of errors by consulting Table 2, columns 'N' and 'Indel'. All programs support mismatches; therefore, it is not mentioned in the aforementioned table. Some programs like **HSHREC** treat all datasets in the same manner with no special handling for different technologies (however, **HSHREC** has a special version that can correct color space reads as a different executable program). We included it in the first category because the base space version does not have a target technology.

The software with different profiles can be further divided into software using the same correction method for all technologies but setting different values for parameters and software with internal algorithmic modifications for a certain technology. For the first group, **Coral** is a perfect example as it

TABLE 2 | The Software and Important Features Support

Type	Name	Par. Tech	Lang.	k	Qual.	Tech	N	Indel	Var. L.	Hzyg.	Rep.	Trim.	Tech. Supp
ksb	CUDA-EC ⁵⁶	CUDA	C++	y	n	Illum.	NA	NA	NA	n	n	y	a
ksb	Reptile ⁸⁴	OMP	Perl/C++	y	y	Illum.	→A	n	NA	n	n	n	a
ksb	Quake ⁷	OMP	C++	y	y	Illum.	NA	n	NA	y	y	y	a
ksb	Edar ⁸⁵	NA	NA	y	n	All	NA	y	y	n	y	n	a
ksb	Hammer ⁸⁶	–	C++	y	y	Illum.	y	n	y	n	n	n	a
ksb	REDEEM ⁸⁷	–	C++	y	n	Illum.	n	n	y	n	y	n	a
ksb	DecGPU ¹⁰⁶	CUDA/MPI	C++	y	n	Illum.	NA	NA	NA	n	n	y	a
ksb	CUDA-EC2 ⁸⁸	CUDA	NA	y	y	Illum.	NA	n	y	n	n	y	a
ksb	Qamar ⁴⁵	PThr.	C++	y	n	Sanger/Illum.	y	n	y	n	n	n	d
ksb	Parallel Reptile ⁸⁹	MPI	C++	y	y	Illum.	→A	n	NA	n	n	n	a
ksb	BayesHammer ⁹⁰	OMP	C++, python	y	y	Illum.	n	n	NA	n	y	y	a
ksb	QuorUM ⁹¹	PThr.	C++	y	y	Illum.	NA	n	y	n	n	y	a
ksb	RACER ⁹²	OMP	C++	y	n	Illum.	→A	n	NA	NA	n	n	a
ksb	Musket ⁷¹	PThr./OMP	C++	y	n	Illum.	y	n	NA	n	y	n	a
ksb	Hector ⁴³	OMP	C++	y	n	454	y	y	y	n	n	n	b
ksb	Lighter ⁹³	PThr.	C++	y	y	Illum.	y	n	y	n	n	n	a
ksb	HERCoI ⁹⁴	Threads	Java	y	n	454/Ion	NA	y	y	n	n	n	d
ksb	Trowel ⁹⁵	Boost	C++	y	y	Illum.	NA	n	NA	n	y	n	a
ksb	LoRDEC ⁶²	PThr.	C++	y	n	PacBio	NA	y	y	n	y	y	a
ksb	BLESS ⁹⁶	–	C++	y	n	Illum.	n	n	n	n	n	n	a
ksb	Blue ⁹⁷	Threads	C#	y	y	Illum./454	y	y	y	n	y	n	c,e
ksb	BFC ⁹⁸	PThr.	C	y	y	Illum.	NA	n	y	y	n	n	a
ksb	Scribble ⁹⁹	NA	NA	y	n	Illum.	NA	n	y	n	y	n	a
ksb	PAGAntec ⁷⁵	OMP	C++	y	n	Illum.	n	n	y	n	n	n	a
ksb	ACE ¹⁰⁰	OMP	C++	y	n	Illum.	y	n	n	y	n	n	a
ksb	FADE ¹⁰¹	FPGA	Verilog	y	n	Illum.	y	n	n	n	n	n	a

TABLE 2 | Continued

Type	Name	Par. Tech	Lang.	k	Qual.	Tech	N	Indel	Var. L.	Hzyg.	Rep.	Trim.	Tech. Supp
ksb	Pollux ¹⁰²	–	C	y	n	Illum./Ion/454	y	y	y	n	n	n	d
ksb	Gu et al. ¹⁰³	NA	C++	y	n	Illum.	y	NA	NA	NA	NA	n	a
ksb	Jabba ¹¹²	NA	C++	y	n	PacBio	n	y	y	y	n	y	c
stab	SHREC ⁵⁵	Threads	Java	n	n	Illum.	n	n	n	n	n	n	a
stab	HSHREC ¹⁰⁴	Threads	Java	n	n	All	y	y	y	y	n	n	d
stab	PSAEC ¹⁰⁵	PThr.	C/C++	n	n	Illum.	n	n	n	n	n	n	a
stab	HITEC ⁴	–	C/C++	n	n	All Subst.	n	n	n	n	y	n	a
stab	MyHybrid ⁷⁰	NA	NA	n	n	All	y	y	y	n	y	n	d
stab	Pluribus ¹⁰⁷	NA	NA	n	n	All	n	y	y	n	n	n	d
stab	Fiona ⁴¹	OMP	C++	y	n	Illum./454/Ion	y	y	y	n	y	n	e
msab	ECHO ¹⁰⁸	Python/PThr.	Python/C++	y	y	Illum.	y	y	y	n	y	n	a
msab	Coral ⁷³	OMP	C	y	y	Illum./454	y	y	y	n	n	n	f
msab	LSC ⁶⁴	NA	Python	n	b	PacBio	y	y	y	n	n	y	c
msab	CloudRS ¹⁰⁹	Hadoop	Java	y	y	Illum.	NA	n	NA	NA	y	n	a
msab	Chung et al. ¹¹⁰	Hadoop	Java	y	y	Illum.	n	n	n	NA	y	n	a
msab	proovread ⁶³	Grid	Perl	n	y	PacBio	NA	y	y	NA	y	y	c
msab	Nanocorr ⁶⁸	Grid	Python	n	n	Nanopore	y	y	y	n	n	n	c
msab	Karect ¹¹¹	PThr.	C++	y	y	Illum./454/Ion	y	y	y	y	n	y	d
rcb	FreClu ⁷⁴	–	Java	n	n	Illum.	NA	n	n	n	n	n	a
rcb	Sleep et al. ⁵²	–	NA	n	n	Illum.	n	n	y	n	n	y	a
pmb	RECOUNT ⁵³	NA	C++	n	y	Illum.	NA	NA	NA	n	n	n	a
pmb	Premier ⁸²	NA	NA	y	y	Illum.	n	n	NA	n	n	n	a
pmb	Premier Turbo ⁸³	NA	NA	y	y	Illum.	n	n	NA	n	n	n	a
pmb	kGEM ¹¹³	NA	NA	n	NA	All	y	y	y	n	n	n	d

The columns have the following meaning: Par. Tech, the parallel technology (if any); k, whether or not a software makes use of k-mers; Qual., support for quality scores; N, support for uncalled bases; Indel, support for indels; Var. L., support for variable length reads; Hzyg., support for heterozygosity; Rep., support for repetitive regions; Trim., whether or not the algorithm incorporates trimming; Tech. Supp, the categories in which a software fits from Figure 2.

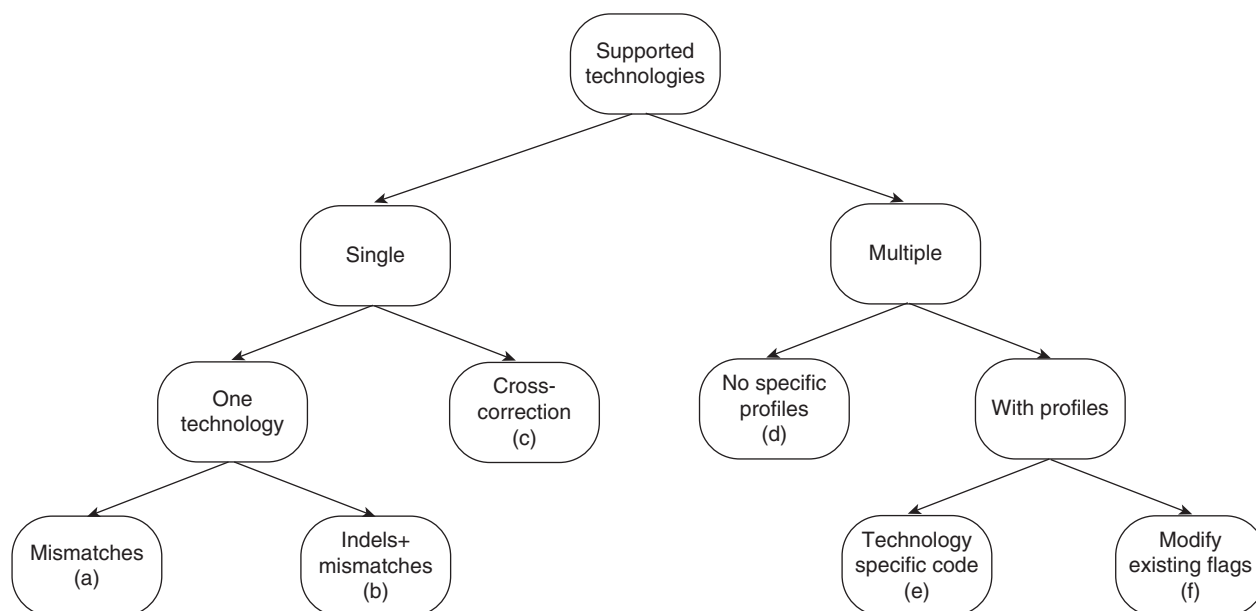


FIGURE 2 | Classification using the technology support among correctors; letters between paranthesises on the leaves used to group the algorithms in Table 2.

uses the same algorithm to correct both Illumina and 454, but in the case of Illumina, the algorithm sets very high values for gaps, forcing mismatches only. **Blue**, on the other hand, has a flag for 454 to enable searching for homopolymer errors. **Karect** has generic support for multiple sequencing technologies, running in two modes, with indels or without.

A new approach is the cross-correction using a high-quality short reads dataset to correct a dataset that has (much) longer, lower-quality reads. There are correctors targeting the very long reads produced by Pacific Biosciences (LoRDEC, proovread, Jabba, and LSC), 454 (**Blue**), and Oxford Nanopore (**Nanocorr**).

The latest review that includes software-supporting indels is from 2013,⁷⁷ and it does not include the latest additions to indel-aware software, like **Blue**, **Fiona**, **Pollux**, or **Karect**. The authors of the review stressed the need for better software solutions with indels support as the results of the existing algorithms at that time (**HSHREC** and **Coral**) were not comparable to the Illumina-specific solutions.

Software Categories

We clustered the algorithms according to their core functionality, extending the work in Refs 77,78. Owing to the length limitation of the article, the individual description of the correctors is located in the Supporting Information. Table 2 summarizes some important features of the analyzed software.

The **k**-spectrum-based (**ksb**) software corrects the reads employing the **k**-mer spectrum.⁷⁹ A **k**-mer is a segment from a read with **k**-bases. The set of all **k**-mers of a read is generated by using a sliding window of dimension **k**. At each step, the window is shifted by one element, and 'the visible' segment of the read is added to the spectrum set. This is by far the most popular approach. Generally, the applications use the **k**-mer spectrum (Figure 3) to decide whether a **k**-mer is correct or not. The error-free **k**-mers are those appearing in a number of reads entering a pre-defined distribution (Gaussian in our example). Roughly speaking, **k**-mers appearing in a small number of reads are considered erroneous; as the coverage is not uniform, the **k**-mers in the low-coverage areas are under-represented (more information about **k**-mers can be found in the section 'K-mer').

The suffix trie/array-based (**stab**) method generally builds a suffix structure with the common parts of the reads. These correctors try to locate inconsistencies in their path while exploring the trie/array. Figure 4 depicts an example where a low frequency of a divergent suffix signals a possible error case. Normally, the reads on a trie follow the same path, but it happens to diverge at some point. A corrector has to decide if the split is an error or not. Figure 4 (a depicts a divergence point (different nucleotide) where the frequency of one of the resulting paths is very low ($<k/2$), and the bases of this path, after the divergence point till the leaves, are exactly the same as for the path with the frequency $\lesssim k/2$; hence, it is

exemplifies the branching caused by an error. The $\$$ symbol marks the end of a suffix (a standard way of depicting suffix tries). It is clear that due to the low frequency of the suffixes containing the bad base, a corrector can isolate the error and can take a valid decision given enough coverage. Owing to the shortness of the reads in our example, we consider one base to be sufficient proof of inclusion in one branch or another. As a result, for the suffix AAA\$, the third base will match its counterpart from the suffix AGA\$ (the first base is the same, as we are talking about the same family of suffixes); therefore, A should be G. Next, the branch TAAA\$ triggers a warning for the corrector due to the low frequency of its sub-branch AA\$. The problematic base is again surrounded by bases that match a sibling path (i.e., TAAA can be converted to TAGA, with the latter having a higher frequency); therefore, it is safe to assume that A is in fact G. Finally, after analyzing these cases, a corrector can support its decision by detecting the relationship between the suffixes AAA\$ and TAAA\$, where the former is in fact included in the latter, and the corrections on both sides have an even higher degree of validity when taken together.

Multiple sequence alignment-based (**msab**) software focus on aligning the reads to identify the overlap between them (see Figure 5). The methods use different algorithms (like Needleman-Wunsch in **Coral**⁷³) to build a consensus from a set of reads that are likely to fit together. Generally, these methods cluster together a number of related reads (e.g., those

FIGURE 3 | Typical distribution of k-mers used by **ksb** correctors; vertical axis shows the number of k-mers that appear in the number of reads displayed on the horizontal axis; first peak corresponds to erroneous k-mers that appears only in a few reads; correct k-mers typically exist in a number of reads close to the coverage; k-mers found in many reads (right part of the spectrum) typically correspond to repetitive regions.

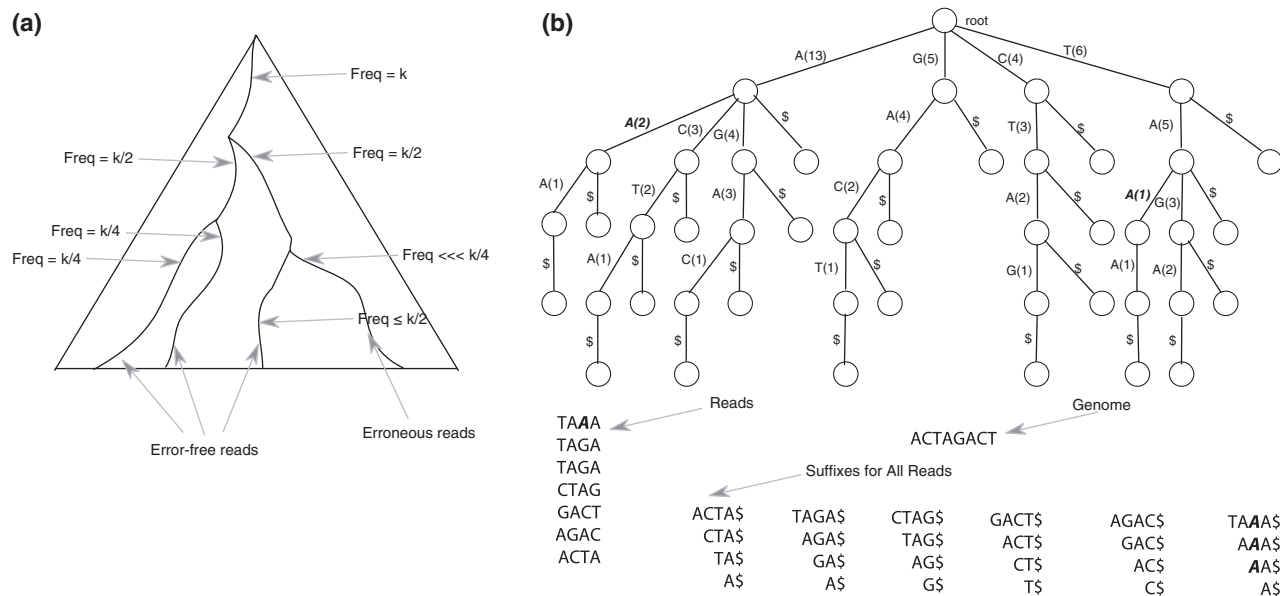


FIGURE 4 | Suffix trie example; (a) an error on the rightmost path results in branch having a very low frequency ($\ll k/2$) compared with its sibling branch ($\leq k/2$); (b) example of a trie for a very short genome with read TA~~A~~A having an error on its third position

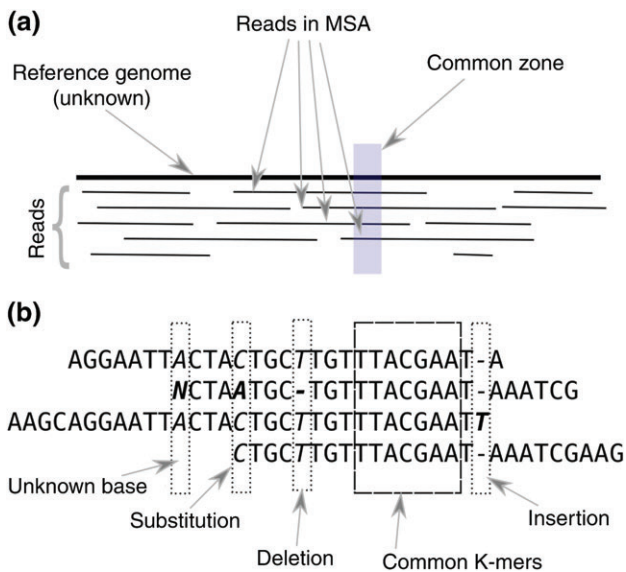


FIGURE 5 | (a) Multiple-sequence alignment of reads versus the (prospective) reference genome; (b) example of four reads with the common k-mer "TTACGAA" and the four basic types of errors.

having at least one k-mer in common, like *Coral*), which may belong to (as the corrector may wrongly include reads from other regions) the same genomic locus. Reads containing k-mers appearing in multiple loci or erroneous k-mers matching wrong locations will normally be detected in the alignment process. Being part of the same region, an *msab* corrector can generate a multiple-sequence alignment and try to determine and fix the anomalies in the resulting consensus. The example Figure 4(b) demonstrates that given sufficient coverage, a corrector is able to group a number of reads, isolate the erroneous bases, and make a decision if possible. In the above example, we are able to take a decision in every case. On the contrary, if we have a mismatch between the first position of the first read and the fifth position of the third read, the decision is not straightforward any more (if possible altogether). For instance, if instead of [A,A] (the correct version) the pair would be [A,N], the corrector would have to either ignore the case or apply some kind of heuristic like converting N to A as A is a valid nucleotide. On the other hand, this heuristic could be rendered useless by the use of quality scores where N has a very high score compared to A (the previously considered valid base may not be so valid after all). In this case, it is up to the corrector to take the appropriate action using different approaches and the context of the problem.

Read clustering-based (*rcb*) methods use different clustering methods to group reads that fit together. This group resembles the *msab* one, but

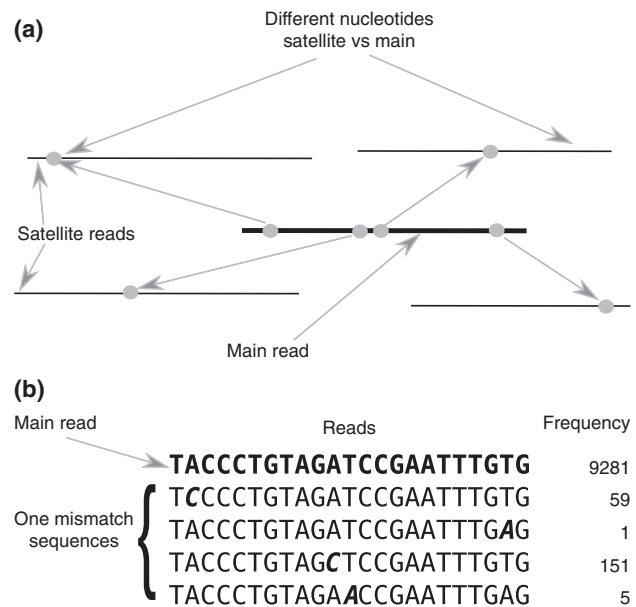


FIGURE 6 | (a) Clustering approach for one reference read and four related having one difference each; (b) real example with the main read market in bold and the satellites aligned and with the different locus market with bold and italic.

the algorithms in it do not generate an alignment. Instead, they search for reads that are similar and choose a consensus that is the correct form for all these similar reads. Figure 6 shows a central read (having the most common part with all the others) and its satellites. For simplicity, we only exemplify one different case. In our example, each of the four satellites have one distinct nucleotide. A corrector should group them together as they show a high degree of similarity; hence, they are in fact clones of the same read but with errors. Figure 6(b) is an example (extracted from the bibliography⁷⁴) where the consensus read is the one with the highest frequency. The other reads differ from the main read by just one nucleotide and also have much lower frequencies. Note that the algorithms included here do not perform a multiple-sequence alignment to determine the correct read; they just group them by differences and search for a valid consensus, the error-free existing read.

Probabilistic model-based (*pmb*) methods use the expectation maximization (EM) algorithm to determine the correct base at each position by calculating the likelihood of the existing variants at that specific position. The problem of error correction boils down to selecting the right nucleotide at a certain position where two or more reads overlap, and there is more than one choice. The *pmb* software base their approach on the fact that this problem has unknown parameters (unobserved component), in

this case, the correct base. As a result, using the existing input data (observed component) and maybe more information (like the error rate), they try to generate a model (after multiple iterations over the same data). This model can state the correction of a certain nucleotide with a certain degree of accuracy. The EM alternates between two steps, the E (guessing the probability) and the M (re-estimating the model parameters using the new probability), until it converges to the desired model. Figure 7 presents the basic algorithm. For an extensive explanation of the EM algorithm, the readers should check the article of Do and Batzoglou from Ref 80. Different algorithms in this category use different position comparison methods (position part of k-mer or read) and convergence points.

Please note that many methods can be included in more than one category. For instance, **Coral**⁷³ is listed in this review in the category of msb algorithms,^{77,78,81} but it also uses the k-mer spectrum to determine the related reads. The same case arises with **Premier**⁸² and **Premier Turbo**,⁸³ which use k-mers to update the probabilities for the variants on a position.

Recommendations

From Table 2, it is clear that depending on the nature of the project, some programs are better than others.

For Illumina projects, almost all correctors can be used. Although, as we shall see in the coming sections (mostly in the section ‘Testing’), the Illumina-only correctors offer a better performance on Illumina data when compared to multi-technology software.

From a computational resources point of view, the correctors written in a low-level language like C++ should be used. One must take this last advice with a grain of salt as the performance is highly dependent on the quality of the code and the algorithms used. Another very important aspect is the multi-core and multi-computer support. Nowadays, even mobile phones are multi-core, and the speed of CPUs has hit a hard limit; therefore, any piece of software capable of scaling on multiple cores should be preferred over the others. These scalable applications are very useful when the time frame is very short. Furthermore, the same programs win when testing multiple combinations of parameters at the same time and running multiple instances of a single-threaded program is not an option (e.g., when the user has to run the next instance of the program with a combination of parameters based on a previous run). Lastly, the multi-threaded programs would normally consume less memory than multiple single-threaded instances running at the same time. The best example for this last observation is an OpenMP

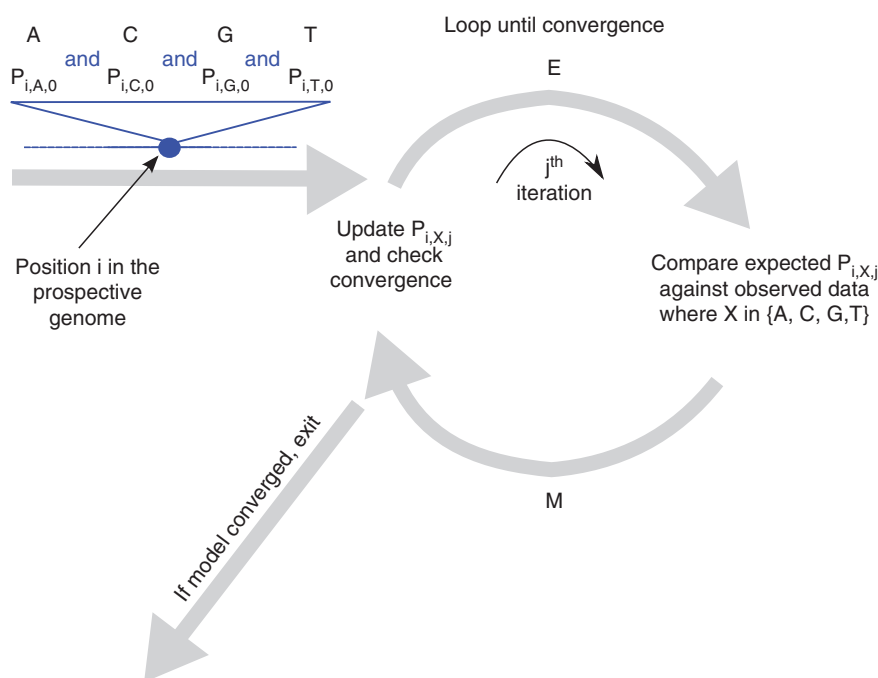


FIGURE 7 | The EM algorithm initializes the probabilities of the bases before entering the loop where it alternates between E-step and M-step; once the convergence threshold has been reached, the method exits and enters the correction stage; the capital P represents the probability for a base to be the real one.

corrector creating a k-mer spectrum. An optimized multi-threaded program would create just one structure to keep the k-mers and their count, and it would allow thread-safe access to the structure. An optimized single-threaded program using the same mechanism would avoid the locks, but for multiple instances, the same structure would be replicated as many times as running instances.

ERROR CORRECTION IN REAL PROJECTS

The software presented so far was used in many research projects. This section is the extension of the motivation where we present instances where scientists successfully employed error correctors. Hopefully, this section opens the door for further usage in comparable or new cases. We concerted our efforts to find real-life biological projects where the utility of correctors is demonstrated by practical use. In contradiction to what we found in the articles accompanying each corrector, some real-life projects use the correctors for additional applications like mitochondrial genome correction¹¹⁴ and RNA-seq.^{115–118}

Recommendations

Continuing the discussion from the subsection ‘Recommendations,’ section ‘Error Correction Software,’ we can now see the role of correctors in real projects. **Quake**⁷ is the most used corrector as it targets the sequencing technology with the largest market share. Furthermore, as we shall see in the ‘Testing,’ it has a good level of correction. As a general rule, Illumina data should be handled by Illumina-only correctors as they should be better tuned for the technology than their general counterparts. There are exceptions like **Coral**⁷³ that is used for both Illumina and 454. The same software is utilized with Ion Torrent for which there is a generic support. Generally, the software supporting all types of errors can be used with unsupported technologies, but the user must understand that the result might not be what (s)he expected. In the above case, Ion Torrent is somewhat similar to 454, and hence, **Coral** works. In the case of datasets from multiple technologies, one can use more than one corrector for each technology as Wang et al. did.¹¹⁵ Another possibility is to use a cross-corrector like **Blue**, **LorDEC**, **proovread**, and **LSC** where instead of stacking up all the reads from multiple technologies, one can use one technology to correct the other. The correctors are used in many types of projects, with the most targeted application being assembly. Variant calling

and different transcriptome studies are also very common in the existing projects. A very important fact emerging from Table 3 is the range of genome data tackled with the stand-alone correctors. The size and complexity ranges from bacteria (*S. pyogenes*) to mammals (Dromedary) and plants (Loblolly Pine).

DISCUSSION

In this section of the paper, we discuss the most important topics for the state-of-the-art error correction. It starts with the general problem of errors in NGS data followed by the key features of the methods and ending with the main testing approaches.

Challenges

Data Preparation and Post-processing Steps

There are cases in which the input data must pass through some additional pre-processing steps like the conversion to a certain format. **Blue** performs a preparation step to generate the k-mer spectrum. **Reptile** has a pre-processing step to separate the reads from their quality scores and to filter reads containing ambiguous characters. **CloudRS** converts the FASTQ input file to a specific format. Furthermore, it must upload the converted input to the Hadoop cluster and download the result locally when the job has finished. **HSHREC** generates the corrected output files without the initial descriptions of the reads. Some tools may need this information for further processing like **SolexaQA++**,¹⁶⁷ which generates statistics from multiple technologies. The dataset requires a post-processing step (that the user must implement) to restore the initial information. Secondly, **HSHREC** generates two files, one containing all corrected reads and the other the skipped reads. Generally, the output of the error correction tools is FASTA/FASTQ, and it does not require any explicit processing.

K-mer

K-mers Handling: Many methods base their decision on k-mers and apply different techniques to deal with the memory limitation and CPU requirements. **BLESS** uses the hard-drive to store the k-mers during the counting.^{168,169} **RACER** encodes the bases in a k-mer as a 2-bit representation to save memory. A newer version of **Quake** integrates Jellyfish¹⁷⁰ to count k-mers instead of its own implementation to stay competitive against the more recent algorithms. It also provides a distributed approach for those cases in which the local memory is not enough to handle the k-mers. To speed up the k-mer spectrum

TABLE 3 | Work Using the Correctors Included in the Current Review

Year	Study Description	Where
Quake/Illumina		
2012	Exomes comparison of <i>C. Carpio</i> & D. Rerio	119
2012	GAGE, evaluation of genome assemblies/ algorithms	69
2013	Salinarchaeum HArchT-Bsk1T genome	120
2013	<i>L. arenae</i> draft genome	121
2013	<i>C. sinensis</i> draft genome	122
2013	Genetic variants in <i>C. sinensis</i>	123
2013	Genome-wide mutations in diploid yeast	124
2014	Results of correction of heterozygous NGS	125
2014	<i>O. sativa</i> de novo assemblies, novel gene space aus/indica	126
2014	Study of hydrocarbons production from fatty acids in Cyanobacteria	127
2014	Genetic diversity in <i>P. pacificus</i> from population-scale re-sequencing	128
2014	Genetic parameters estimation and response to selection in breeding program of <i>M. galloprovincialis</i>	129
2014	Metagenomic characterization of <i>C. defluviicoccus tetraformis</i>	130
2014	Prediction of antibiotic resistance by gene expression profiles	131
2014	Methicillin resistance in <i>S. aureus</i>	132
2014	De novo creation of repeat libraries from whole-genome NGS reads	133
2014	Aerobic fungal degradation of cellulose	134
2014	<i>B. tryoni</i> draft genome	135
2014	Genome reorganization	136
2015	<i>P. vulgata</i> / <i>P. lamarcki</i> draft genomes	137
2015	The domestic dromedary genome	138
2015	The brown kiwi genome	139
2015	Comparative genomics of <i>S. pyogenes</i> M1	140
2015	Approach for identification and characterization of foodborne pathogens	141
2015	<i>P. glaucus</i> complete mitochondrial genome	114
2015	Mechanisms for speciation and caterpillar chemical defense	142
BayesHammer/Illumina		
Year	Study description	Where
2014	GABenchToB, assembly benchmark for bacteria genomes	143

TABLE 3 | Continued

Year	Study Description	Where
2014	<i>C. burnetii</i> genome	144
2014	<i>P. atrosepticum</i> genome	145
2014	Hidden diversity in honey bee gut symbionts	146
2014	<i>S. lemnae</i> draft genome	147
2015	Discovering natural products from Cyanobacteria	148
2015	Characterize the metabolism of <i>M. thiooxydans</i> L4 in the marine environment	149
2015	Utilization of alginate and other algal polysaccharides by marine <i>Alteromonas macleodii</i> ecotypes	150
2015	Genome-Wide Re-distribution in Active Yeast Genes	151
2015	Study of the metabolome of <i>M. producents</i> JHB	152
Reptile/Illumina		
2014	Decrypting cryptobiosis-analyzing anhydrobiosis using transcriptome sequencing	115
2015	SNP genotyping and population genomics from expressed sequences	153
HSHREC/Illumina, 454		
2014	Decrypting cryptobiosis-analyzing anhydrobiosis using transcriptome sequencing	115
BLESS/Illumina		
2014	Transcriptome, sequence polymorphism, and natural selection in <i>P. eremicus</i>	116
Blue/Illumina		
2014	<i>S. scitamineum</i> genome	154
Coral/Ion Torrent(a), Illumina(b), 454(c)		
2014	GABenchToB, assembly benchmark for bacteria genomes(a)	143
2014	Global gene expression in the exocarp of developing <i>P. avium</i> L.(b)	117
2015	Comparative genomics/gene expression applied on <i>P. xuthus</i> and <i>P. machaon</i> genomes(c)	155
DecGPU/Illumina		
2013	Genomic analysis of <i>S. dulcamara</i>	156
Echo/Illumina		
2012	Pipeline for small RNA-seq data analysis	118
2014	Results of correction of heterozygous NGS	125
2014	Assembly/annotation for <i>T. pratense</i>	157
Freclu/Illumina		

TABLE 3 | Continued

Year	Study Description	Where
2011	MicroRNA-mediated gene regulation role	158
2011	Purification of monocyte subsets from <i>H. sapiens</i> blood and their transcriptomes analysis	159
2013	Identification of functional cis-regulatory elements	160
Hector/454		
2015	Triple-negative breast cancers in patients with no BRCA1 or BRCA2 mutation	161
Lordec/PacBio, Illumina		
2015	<i>De novo</i> tandem repeat detection using short & long reads	162
LSC/PacBio, Illumina		
2014	<i>D. officinale</i> genome and genes analysis	163
2015	Detect fusion genes, determine fusion sites and identify and quantify fusion isoforms	164
2015	<i>S. miltiorrhiza</i> transcriptome and tanshinone biosynthesis insights	164
proovread/PacBio, Illumina		
2015	Characterization of venom toxin-encoding genes in <i>E. coloratus</i>	165
QuorUM/Illumina		
2014	<i>P. taeda</i> reference genome	1
RECOUNT/Illumina		
2012	Brain tumor glioblastoma-derived neural stem cells transcriptome analysis	166

L. arenae, *Litoribacter arenae*; *C. sinensis*, *Citrus sinensis*; *O. sativa*, *Oryza sativa*; *P. pacificus*, *Pristionchus pacificus*; *M. galloprovincialis*, *Mytilus galloprovincialis*; *C. defluviicoccus tetraformis*, *Candidatus Defluviicoccus tetraformis*.

generation, **Parallel Reptile** implements a parallel counter. The authors of Ref 106 use general-purpose computing on graphics processing units (GPGPUs) to generate the k-mer spectrum. A rare feature is the support for variable k-mers for grouping reads as in Ref 109, where the corrector uses a wildcard-based k-mer.

K-mer Size Selection: The value of *k* is extremely important.^{7,82} A very low value for *k* would result in many k-mers appearing in most reads, thus joining in group reads without any real relation. On the contrary, very large values would generate too many unique k-mers, which also have a higher probability of including more errors, therefore introducing noise into the grouping. Long k-mers may also require more RAM memory.

The analyzed methods set the size of the k-mer by accepting a user value, having a fixed default

TABLE 4 | Formulas to Determine the k-Mer Size for Non-automatic k-mer Determination

Formula	Where
$k = \log_4 200N$	7,70
$k \geq \log 4N$	73
$k > N$	84
$k = \lceil l/6 \rceil$	108
$k = \log_4 2Np^{-1}$; $p = 10^{-4}$	94
$Ns/4k \leq 0.0001$	96

N = Genome length, *l* = read length; *p* = probability that a random k-mer appears in a random string of length *N*, using the alphabet {A, C, G, T}; *Ns* = number of unique solid k-mers as reported by BLESS.

value, and/or performing automatic selection. **Hammer** requires the user to set it. **Quake** and **ECHO** define a formula to determine the optimum value as presented in Table 4. **Reptile** considers $10 \leq k \leq 16$ enough for microbial genomes, fitting the spectrum in less than 4 GB of RAM. **Coral** uses a default value of 21 for the k-mer. **Hector** and **Musket** require both the k-mer length and the total estimated number of k-mers from all reads. **HiTEC** and **Fiona** automatically identify the optimum k-mer length at each step. **CloudRS** stacks reads using a wildcard-based 25-mer and, later in the correction procedure, a fixed 24-mer. The k-mer size should be odd in order to avoid palindromic k-mers.¹⁷¹ The software using k-mers in their pipeline are marketed in Table 2, column '*k*'.

K-mer Distribution: Software relying on correct and erroneous k-mers try to fit the k-mer spectrum on a certain distribution. The correctors compute the histogram with the frequencies for each k-mer in the set of reads. A valid estimation tries to model the initial, complex distribution as a combination of multiple, simpler distributions.⁷ **Quake** divides the solid k-mer distribution in a combination of a normal and a Zeta distribution, and it considers (like **BLESS**) the weak k-mers to generate a Gamma distribution. **Lighter** assumes a Poisson distribution, like **Fiona**. **REDEEM** models the k-mer distribution as a multinomial one. For 454, **Hector** encodes homopolymers using the base and the multiplicity. The authors observe that the distribution of the original reads tends to be unimodal. With the encoding applied in the homopolymers space, the distribution of the homopolymers spectrum is analogous to the one (bimodal) obtained by **Musket** in base space. The same authors conclude that, generally, the homopolymer spectra are bimodal.

Coverage Cut-off and K-mer Distance: K-mer-based error correction methods can cut the k-mer histogram to remove k-mers with too high or too low

frequencies, which normally reduces the noise caused by highly repetitive regions or singular errors. Some algorithms automatically compute the best cut-off value, allowing users to override this value. **Quake** uses the Broyden-Fletcher-Goldfarb-Shanno method to calculate the histogram cut-off, but this method fails when the curve of the distribution is not smooth enough.⁷¹ The authors of **Musket** empirically determined that the lowest count for a k-mer around the valley can be a good cut point. **Musket** also has an option to use a user-provided value. **Trowel** uses a different approach by using the contiguity of high-quality-score bases instead of the coverage. It also expands the trusted k-mer set, adding new k-mers after they are corrected. **QuorUM** bases its decision on the quality of bases from k-mers. Therefore, all bases in a solid k-mer must have a quality greater than a threshold. **RACER** uses an internal threshold to deem a k-mer followed or preceded by a certain base either as solid or weak. **Hammer** and **Reptile** create a Hamming graph for the array of all k-mers, locating groups of similar k-mers that only differ in a few positions and then collapsing all those k-mers into a consensus k-mer. To improve memory consumption, **Lighter** uses a random method to decide whether or not to store a k-mer, assuming that a correct k-mer appears multiple time in a dataset; thus, the chance of being selected is high.

Repetitive Regions

In general, the problem of repetitive regions cannot be tackled by considering individual reads or k-mers in isolation. Some argue that in the case of highly repetitive genomes, a sequencing error has a greater probability to change a solid k-mer to another solid k-mer.⁷ They calculated the percentage of all one-base mutations for a k-mer k that will convert k into a sequence that also exists in the genome. The results show a 2.25% for *E. coli* and 13.8% for the human chromosome 1, with a 15-mer and an 18-mer, respectively. Increasing the k-mer length up to 19 did not significantly change the result, dropping the percentage to 11.1% for the *Homo sapiens*' first chromosome. The different percentages obtained for the two organisms result from the higher complexity of the human genome. **REDEEM** was specifically designed to handle repeats. The main problem with repetitive regions is the similarity of two sequences that reside on different loci of the genome. A corrector may try to convert them to a consensus, hence destroying the existing valid zones. These wrongly fixed reads would prevent an assembler from correctly composing the real genome (or make it generate chimeric assemblies). It is a real problem for

highly repetitive genomes (plants).⁷⁷ Furthermore, misreads in repetitive regions can cause an abnormal high frequency of a k-mer,¹⁷² which could result in an erroneous classification as solid by some correctors. The methods based on multiple-sequence alignment are more resilient to challenges posed by repeats, although they do not totally solve the problem.¹⁰⁸ The relationships between reads can tackle some small repetitive regions to some extent because of the higher length of the analyzed strings compared to k-mers. Moreover, a read from a repetitive region has a higher probability to enter the right group provided it shows enough dissimilarity with other repetitive regions. **Fiona** implements a filter to remove suffixes with an unreasonably high frequency and supports tandem repeats. **Blue** addresses the problem with repeats by evaluating alternative fixes for a read (it works in the case of significant differences among reads). **proofread** takes into account the loci of the long reads where large blocks of short reads map, collecting many reads. In contrast, nonrepetitive regions may not even participate in alignments because of the resulting uniqueness from the Pacific Biosciences's high error rate. **CloudRS** makes use of a high frequency k-mer filter to avoid stacking reads from repetitive regions. Column 'Rep' from Table 2 lists the support for repetitive regions.

Ploidy

A corrector must distinguish between errors and variants. As most error correctors were tested on bacterial genomes, the information about the behaviour of most tools is restricted to the haploid case. The support for heterozygosity is stated in the column 'Hzy.' in Table 2. Authors of Ref 52 apply a smoothing technique to avoid removing zones with only biological variants. They build a tree using the frequency of reads and consider a true variant as a sequence appearing with a high enough frequency compared with the parent sequence. Their decision is based on the fact that the frequencies of a variant should be much higher than the ones from the sequencing errors. The authors of **ECHO** explain a modification to support a diploid genome with homozygous and heterozygous genotypes. Their approach is to consider a uniform distribution over all possible genotypes. They skip a correction if the estimated coverage is much greater than the expected coverage at an analyzed locus.

Read Trimming and Splitting

To avoid the propagation of errors to the next steps, the correctors may eliminate bases from both ends of a read, which can be considered a complementary

method to reduce errors.¹⁰⁴ Users should take great care in using trimming because it can heavily influence the next steps like assembly, where the final result may become fragmented⁹¹ as the assembler is not able to find proper overlaps among reads. However, not trimming faulty bases may result in erroneous assemblies.⁶⁹ Some authors^{56,106,173} try to fix the read first, and when this is not possible, they trim it. If the result remains unsatisfactory, they discard the read. Another approach is to pre-process the reads and cleave the suffixes and prefixes that have low-quality scores to decrease the number of false positives in the k-mer spectrum.⁸⁸ One must be careful with the quality scores and remember that the accuracy of the quality scores depends on the library preparation method as demonstrated for Illumina MiSeq® data.⁴⁸ When dealing with very long reads like Pacific Biosciences with a high percentage of errors,⁶² along with considering trimming their ends, an additional approach is to split the long reads into smaller, high-quality segments. The authors of Ref 85 do not correct the spurious bases, opting instead to split the read at the locus of the error and to remove the faulty base(s). They argue that for a further assembly step based on k-mers such as Velvet,¹⁷¹ their method should not pose any problems.

Unknown/Uncalled Bases

The unknown bases (denoted by N in Illumina and Roche 454 sequencing) are one of the four basic types of errors. Schirmer et al. conclude that this type of error does not occur randomly as supported by their nonuniform distribution. Column 'N' from Table 2 indicates whether the correctors support Ns or not. In some cases,^{4,52,55,104} the authors prefer to exclude all reads that contain one or more uncalled bases. Other programs (such as LSC) eliminate the reads that have a frequency of Ns above a threshold. A different strategy is to convert each unknown to a real base, like RACER, Reptile, and Parallel Reptile. Tools such as Refs 73,97 tackle the unknown bases in the correction process. Some authors fail to mention the support for uncalled bases in their papers; therefore, it is up to the user to experiment. As the unknown bases are a type of error,⁴¹ the authors should make it clear whether their software supports them or not. In Ref 97, the authors put together a table with a selection of correcting software and their features, including the handling of uncalled bases.

Low-Coverage Regions and Uniformity

Illumina sequencing generally has a higher average coverage than other platforms,¹⁷⁴ but the reads may not be suited for phylogenetic profiling due to their

limited length when a high resolution is required. However, many studies^{54,175–177} have found that GC-poor and GC-rich regions have low coverage or even no coverage at all. An error corrector must consider these platform-specific shortcomings to increase sensitivity and specificity.⁷

The authors of **MyHybrid** and **Coral** state that correction methods expect the coverage to be relatively high and use that multiplicity for a meaningful decision. Therefore, they cannot do much for reads from low-coverage regions. The methods that use a threshold for weak/solid k-mers will work if the coverage is high enough or uniform, but they will end up destroying the low-coverage regions (**QuorUM** tries to avoid this problem). **Edar** takes into account the bias introduced by GC regions when calculating the k-mer coverage by actively considering the GC content of the k-mer. For a reliable result, the authors recommend using a reference genome to accurately calculate the coverage. While many authors do not state the minimum required coverage for a successful correction, **ECHO**'s paper specifies a coverage of 15 or higher. **Hammer** and **BayesHammer** are specifically designed for error correction without uniformity assumptions. Due to uniformity, some authors admit their algorithm's limit, like in case of **Reptile** where a nonuniform coverage and the existence of more than one acceptable tile force the algorithm to skip a correction decision. The authors of **Blue** mention the caveat of a simple k-mer cut-off due to uniformity that can result in the rejection of correct k-mers in low-coverage regions and the acceptance of erroneous k-mers in very high-coverage regions. **Fiona** detects erroneous k-mers by calculating the expected coverage for each k-mer, given a uniform sampling of genomic positions. It uses a hierarchical statistical model to describe the expected coverage distribution of k-mers based on library preparation and sequencing.

Parameters

All the methods rely on specific thresholds, lengths, ratios, and/or probabilities to drive their correction. As the methods evolve, they tend to move the burden of choosing the best-suited values for their parameter from the user to themselves.

Generally, the k-mer size has a default (user adjustable) value. However, some correctors like **Quake** require an explicit value from the user but offer a formula to determine it. **Coral** has a default value, but it also proposes a formula in order to obtain the best results.

The technology flag can explicitly set the source technology. For example, **Coral** has flags for

'Illumina/454,' while **Fiona** has for 'Illumina/Ion Torrent/454', which helps the software to decide the best approach for correction. Others have options that target the errors specific to a specific platform, that is, homopolymers errors in 454 with the 'hp' flag for **Blue**. **Karect** can run with indels support (Ion Torrent, Roche 454) or without it (Illumina).

Parameter selection automatic/manual: A manual method to select parameters requires the user to try different values for different parameters to obtain the best results. An automatic method would prevent the user to provide additional valuable information to infer the best actions the algorithm has to take during the correction process. We must distinguish between automatic determination of the best value for the dataset/analyzed case and the default value of a parameter (deemed by the authors to be an acceptable value). **HiTEC** and **Fiona** support fully automatic parameters values selection. Note that **HiTEC** needs the length of the genome and the percentage of errors as input, but these two parameters remain the same for a certain dataset.

Single Threaded vs Parallel

Generally, the programs from this review support parallel processing using multiple threads. There are methods, like **BLESS**, that can compete against multi-threaded software due to their approach, despite being single threaded. Other methods like **Reptile** have been updated to run on multiple CPUs,⁸⁹ using the same initial correction mechanism. The parallel implementations are a normal trend as both the NGS data size and the length of the reads increase.

A distinction must be made between those programs being natively parallel (they internally split the jobs between multiple workers), like **Coral** and **HSHTREC**, and those that have no parallel implementation, but their input can be divided in chunks, and multiple processes can be launched on different fragments of the initial dataset, like in the case of **proovread**.

Parallel Technology: The readers can check the parallel technology used by a corrector in column 'Par. Tech.' from Table 2. Most of the parallel implementations use OpenMP to distribute the workload among the threads. **DecGPU** and **Parallel Reptile** support distributed-memory computing by using the Message Passing Interface (MPI). The distributed-memory model requires more complex programming and configuration but enables the use of a larger amount of RAM memory. This advantage may enable tackling larger-scale correction problems that cannot be addressed on a single node. Other methods, such as **Quake**, use parallelism only for counting

k-mers. Furthermore, **proovread** and **Nanocorr** can run on multi-core desktops and distributed clusters using queuing engines like Sun Grid Engine (SGE) (commercial)^a and Simple Linux Utility for Resource Management (SLURM).¹⁷⁸ As of now, there are two fully distributed methods using the Map-Reduce (Hadoop) paradigm.^{109,110} An interesting addition to the field is **FADE**,¹⁰¹ the Field Programmable Gate Array (FPGA) error corrector that unleashes the massive parallelism available on FPGA devices to tackle the errors.

Operating System and Programming Language

Resource consumption is a problem because of the continuous growth of the size of the NGS data. Owing to this, we observe that the majority of authors chose a low-level language like C or C++ to implement their solution. An interesting trend is the use of C++ over C in writing the software, with just one program (from those being available online for us to analyze), **Coral**, being implemented in pure C. The C# implementation of **Blue** obtains the best performance when compared with other algorithms on Illumina and Roche 454. Even though C# is not considered to be a high-performance language, **Blue** performs really well against the rest of the algorithms. Some correctors appear to be implemented in Perl and/or Python. These are often scripts that are used to execute third party software. In case of **LCS**, the authors offer a software wrapper written in Perl that uses an external aligner to map the short reads against the long ones. **Nanocorr** is a python wrapper for BLAST and pbdagcon^b. For the readers' convenience, we list the programming language of choice for each corrector in Table 2.

Overall, the correctors should work on the three most important desktop/server operating systems: Windows, Linux, and Mac OS. Some authors⁹³ mention the supported platform in their papers. Generally, the authors prefer to support Unix flavors and to distribute the source code and the instructions to build it. Furthermore, the tests for the majority of correctors were performed under Linux, with some authors also using Windows.⁵² Besides personal computer-based methods, we included a corrector that runs on FPGAs—**FADE** (even though it uses the help of a computer to handle the data transfer and storage).

Recommendations

Depending upon the pre-processing type, a corrector that can separate the steps of the correction can save a lot of time when processing big datasets. For

instance, **Blue** can generate the k-mer table in a separate process. In its case, the advantage is twofold. Firstly, it can use the same k-mer table for multiple runs with the different combinations of values for the majority of parameters that are not involved in the histogram's generation. Secondly, it makes cross-correction possible as the k-mer table can be generated from one technology that can be used to correct data from a different one. Another important aspect is the k-mer size selection. Any program able to determine the size of k-mers automatically (**HiTEC**) or use variable sizes (**Fiona**) is recommended over those with user-defined only k-mer size selection. Another recommendation is to select a software like **Fiona** or **Blue** that considers repetitive regions as they may avoid altering similar zones with SNPs. The unknown bases support depends upon the sequencing technology used as some technologies do not produce this type of error. Furthermore, one can easily write a script that can convert the unknowns into random or specific nucleotides. The low-coverage issue must always be a top priority as some algorithms can skip those zones because of limited information. Finally, the user must be careful with the trimming and the splitting of the reads. In the former case, a corrector (like **Quake** and **DecGPU**) may trim a read if the correction is not possible. If the result is then fed to an assembler, the correction may negatively influence the overlap detection. **Edar** applies a distinct correction mechanism by cleaving the reads. This approach may be detrimental for any further steps because the reads become much shorter and the relationships between sub-segments that are part of the same read are lost forever.

Testing

This section focuses on the testing part included in the analyzed papers. We extracted all datasets that we could identify in the papers along with the results provided by the authors. Owing to space limitation, we split the datasets in two categories. Table 5 lists all those datasets appearing in Refs 77,78. The rest of the datasets (second category) along with the benchmark information are located in the Supporting Information. The gain metric for both categories is calculated (by the authors or by the reviewers) using the formula from Ref 77. The readers must be careful though because there is no standard way to count true positives (TP), false positives (FP) and false negatives (FN). As a result, the numbers given by the authors and the reviewers must be taken with a grain of salt. The reliable gain appears in the same column (same review, the authors used the same testing

approach for all software). The hardware configurations used by the correctors' authors and in Ref 78 as more sections related to testing are located in the Supporting Information.

Methods

Some metrics are general and do not focus on certain types of error correction mechanisms. For instance, the sensitivity and specificity appear in k-mer-spectrum methods like **Reptile**, suffix trie/array methods like **SHREC**, and multiple sequence alignment (MSA) methods like **Coral**. The sensitivity, the specificity, the gain, and the genome assembly statistics are the most widespread metrics.

Simply counting the mapping reads that did not map before and after the correction can prove the effectiveness of a corrector.^{53,64,104} However, the differences obtained heavily depend on the aligner's parameters. For example, the authors of Ref 77 test with different values for the aligner, albeit only for datasets with indels.

In the case of artificial datasets, it is possible to report quite reasonable error rates before and after correction.^{85,108} However, the exact error rate for real data can only be estimated.⁴⁴

Gain/Specificity/Sensitivity

The gain (G), specificity (SP), and sensitivity (SE) metrics (for formulae, see Equation (1)) seem to have become the *de facto* on error correction. SP and SE first appeared in Ref 55. G^{84} represents the percentage of eliminated errors. They are all based on counting:

- TP (true positives) existing errors that are corrected,
- TN (true negatives) correct bases left unmodified,
- FP (false positives) correct bases that are wrongly considered being faulty, and
- FN (false negatives) erroneous bases left unmodified.

$$G = \frac{TP - FP}{TP + FN}, \quad SE = \frac{TP}{TP + FN}, \quad SP = \frac{TN}{TN + FP} \quad (1)$$

There are differences in how the authors of each tool compute TP/TN/FP/FN. For **Reptile**, they compute the errors on base level, while for **SHREC** and **RACER**, they count the errors on reads level (a read is either error-free or erroneous, without considering the number of bad bases). The lack of a standard approach on counting the errors leads to some

TABLE 5 | Testing Result From Benchmarks and Original Papers

Dataset	Genome	Tech	# Rds	# Bs	Software	Mem O	Mem S	Rt O (Num Th)	Rt S	Gain O	Gain S	Gain R
SRX000429	<i>E. coli</i> 36	Illum	10.4	749.4	SHREC	–	–	–	–	–	–	0.75
					HSHREC	–	14.3	–	153.23	–	–2.886	0.989
					HiTEC	–	13	–	143.13	0.861	0.846	0.952
					DecGPU	–	–	–	–	–	–	0.772
					Coral	–	7.5	–	36.19	–	0.579	0.857
					ECHO	–	–*	–	–*	–	–*	0.715
					Hammer	–	–	–	–	–	–	0.944
					Quake	–	4.1	–	38.88	–	0.784	–*
					REDEEM ⁹	9	–	120(1)	–	–	–	0.926
					RACER ⁴	1.1	–	15.7(1)	–	0.908	–	–
					Trowel	3.5	–	2.9(32)	–	0.765	–	–
					CloudRS	–	–	–	–	0.207	–	–
					Reptile ⁵	1.1	3.7	47.4d 149.4d2(1)	23.32	0.757d1 0.802d2	0.933	–0.047
SRR001665_1 ¹⁰	<i>E. coli</i> K12 MG1655	Illum.	10.4	749.4	Par. Reptile ⁵	–	–	0.18d1 1.38d2 (512)	–	0.757d1 0.802d2	–	–
					SHREC	–	–	–	–	–	–	0.822
					HSHREC	–	–	–	–	–	–	0.989
					HiTEC	–	–	–	–	0.857	–	0.795
					DecGPU ²	–	–	12.31(4)	–	–	–	0.999
					Coral	–	–	–	–	–	–	0.999
					ECHO	–	–	–	–	–	–	0.999
					Hammer	–	–	–	–	–	–	0.668
					REDEEM	–	–	–	–	–	–	0.231
					BLESS	0.01	–	23(1)	–	0.967	–	–
					Qamar ⁶	3.5	–	38.85(–)	–	0.913	–	–
					Reptile ⁵	0.84	–	21d1 73.8d2(1)	–	0.652d1 0.709d2	–	0.998

TABLE 5 | Continued

Dataset	Genome	Tech	# Rds	# Bs	Software	Mem O	Mem S	Rt O (Num Th)	Rt S	Gain O	Gain S	Gain R
SRR006332	Acinetob. ADP1	Illum.	18.13	652.7	FADE	—	—	0.51(FPGA)	—	0.923	—	—
					SHREC	—	—	—	—	—	—	0.89
					HiTEC	—	—	—	—	—	—	0.324
					DecGPU	—	—	—	—	—	—	0.548
					Coral	—	—	—	—	—	—	0.816
					ECHO	—	—	—	—	—	—	0.711
					Hammer	—	—	—	—	—	—	0.984
					Reptile ⁵	2.2	—	99.6d1(1)	—	0.632d1	—	0.982
					Quake	—	—	—	—	—	—	0.926
					CUDA-EC	—	—	—	—	—	—	0.953
					REDEEM	—	—	—	—	—	—	0.595
					CloudRS	—	—	—	—	0.265	—	—
NC_005966	Acinetob. ADP1	Illum.	4	144	SHREC	—	—	—	—	—	—	0.997
					HSHREC	—	—	—	—	—	—	0.995
					HiTEC	—	—	—	—	—	—	0.994
					DecGPU	—	—	—	—	—	—	0.995
					Coral	—	—	—	—	—	—	0.997
					ECHO	—	—	—	—	—	—	0.993
					Hammer	—	—	—	—	—	—	0.997
					Reptile ⁵	0.66	—	15.6d1(1)	—	0.599d1	—	0.995
					Quake	—	—	—	—	—	—	0.993
					CUDA-EC	—	—	—	—	—	—	0.995
					HSHREC	—	29.9	—	779.15	—	−1.606	—
					Reptile	—	19.1	—	225.43	—	0.91	—
ERA000206	<i>E. coli</i> K12 MG1655	Illum.	14.21	2800	HiTEC	—	9.8	—	683.87	0.874	0.949	—
					Coral	—	30	—	450.29	—	0.112	—
					Blue ⁷	1.6	—	+0.4	—	9.2+14.4(8)	—	—

TABLE 5 | Continued

Dataset	Genome	Tech	# Rds	# Bs	Software	Mem O	Mem S	Rt O (Num Th)	Rt S	Gain O	Gain S	Gain R
-	-											
SRR022918	E. coli 47	Illum.	7.2	677.2	HSHREC	-	12.6	-	74.13	-	-0.316B -0.269R	-
					Reptile ⁵	1.9	3.4	56.4d1(1)	71.64	0.381d1	0.852B 0.791R	-
					Quake	-	2	-	80.43	-	0.313B 0.285R	-
					HiTEC	-	6.2	-	59.19	-	0.929B 0.912R	-
					ECHO	-	16	-	304.21	-	0.907B 0.897R	-
					Coral	2.6	7.7	8.12(4)	8.32	0.974	0.002B 0.002R	-
					RACER ⁴	1.5	-	13.65(1)	-	0.826	-	-
					Trowel	4.5	-	3.7(32)	-	0.447	-	-
SRR000868	E. coli UTI89	454	0.4	118.2	Fiona	1	-	2.5(8)	-	0.768	-	-
					Hector ³	-	-	3(2)	-	0.882	-	-
											0.293E1	-
											0.224E2	-
SRR000868	E. coli UTI89	454	0.4	118.2	HSHREC ⁸	-	9.9	-	16.72	-	0.470E3	-
											0.473E4	-
											0.451E5	-
											0.497E1	-
											0.425E2	-
					Coral ⁸	1.8	2.4	5.3(4)	2.57	0.857	0.698E3	-
											0.702E4	-
											0.734E5	-
					HSHREC	-	30.1	-	1027.48	-	-2.497	-
					Reptile	-	4.12	-	165.88	-	0.224	-
SRX100885	S. cerevisiae	Illum.	26.03	4000	Coral	-	34.5	-	544.31	-	0.067	-
					RACER ⁴	3.2	-	198.31(1)	-	0.146	-	-
					CloudRS	-	-	-	-	0.779	-	-
					HSHREC	-	30.2	-	813.65	-	-4.24	-

TABLE 5 | Continued

Dataset	Genome	Tech	# Rds	# Bs	Software	Mem O	Mem S	Rt O (Num Th)	Rt S	Gain O	Gain S	Gain R
SRR022866	<i>S. aureus</i> USA300	Illum.	12.77	1900	Reptile Coral	– 4	13.1 40	– 80(4)	295.94 210.17	– 0.974	0.613 0.025	– –
SRX023452			18.96	3600	RACER ⁴	2.1	–	81.68(1)	–	0.47	–	–
SRX006152	<i>D. melanog.</i>		9.46	1400	Trowel	5.7	–	6.4(32)	–	0.447	–	–
SRX006151			20.8	1900	CloudRS HSHREC Reptile Quake Coral RACER ⁴ Par. Reptile ⁵ Fiona	– – – – – 39.4 – – 1	– – 30.3 24 12.1 30 – – –	– – – – – 501.95(1) 2.82d1 412.98d2(512) 3.1(8)	– 2562.88 532.76 222.35 373.19 – – – –	– – – – – 0.57 – – 0.905	– –5.85 0.667 –0.126 0.449 – – – –	– – – – – – – –
ERR039477	<i>E. coli</i>											
DH10B	lon	0.39	36	HSHREC ⁷	–	10.21	–	8.72	–	0.399T1	–	–
0.450T2	–				Coral ⁷	–	2.24	–	1.13	–	0.515T1	–
0.604T2	–											

Mem/Rt/Gain, memory in GB/runtime in mins (num of threads in parentheses for Rt O)/gain as percentage reported in the original paper (represented by O) and in the survey by⁷⁷ (represented by S); Gain R, gain(percentage) reported by⁷⁸; # Rds, number of reads, # Bs - number of bases; the programs that did not run successfully are marked by “-”.

¹ Resources for preparing the dataset and correcting the data.

² The result for 1xGPU execution, for more threads and GPUs please refer to.

³ Approximate values for runtime extracted from the chart in the paper for the original results.

⁴ Results for the datasets filtered using BWA.

⁵ Runtime for Hamming dist (1,2) between two k-mers.

⁶ Gain calculated by us using the values for TP, FP, FN given by the authors and the formula in⁷⁷.

⁷ Profiles: T1 - With all mapped reads; T2 - excluding reads with more than 10 errors.

⁸ Different profiles for the aligner and a k-mer length of 10; The parameters set for E(1–5) can be found in⁷⁷.

⁹ Dataset not explicitly pinpointed, we inferred the value.

¹⁰ SRR001665_1 is the same as SRX000429, but only the forward direction is used.

serious inconsistencies in the results published in the literature by the same tool in different benchmarks even using the same dataset and formula. For example, in Ref 77, **Coral** obtains a score of 0.002 for an Illumina dataset (SRR022918), while with the same dataset, it scores 0.97 in its own paper. There is no doubt that **Coral** is a good corrector (as demonstrated by Ref 41), and even in the aforementioned survey, it performs really well on datasets with indels. The problem lies with the different approaches for the tests, which are not infallible. Moreover, the previous difference in score may arise just by changing the way of preparing the datasets before correction. Even though the approach is the same (filter non- and multi-mapping reads), the aligner can make a difference too. Salmela et al.⁷³ use Soap, while Yang et al.⁷⁷ use BWA.^{179,180} Table 5 contains the results obtained by some correctors in Refs 77,78 on a number of datasets. The different results obtained in the original article versus the surveys can be explained by the difference in dataset preparation, FP/TP/FN/TN counting, and maybe different versions of the tested programs. The Supporting Information contains the results.

Assembly

The majority of the recent publications include some information about the assembly performance. Many list the N50 metric and the contigs count, but there are variations. Salzberg et al.⁶⁹ define N50 value as 'the size of the smallest contig (or scaffold) such that 50% of the genome is contained in contigs of size N50 or larger.' A contig is a multiple-sequence alignment of reads represented as a consensus, while the scaffold is a list of contigs that defines their order, orientation, and the length of the gaps between them.¹⁸¹ **Pluribus**' paper provides the number of nodes in the Bruijn graph generated by Velvet, which gives a measure of the fragmentation of the assembly. For **QuorUM**, the E-size statistics⁶⁹ complements the N50 value.

While N50 and the maximum contig length measure the quality of error correction and give valuable feedback over the correction, the authors of Ref 97 state that these metrics are not always accurate. This happens mainly because the assemblers can generate chimeric contigs in overcorrected datasets. In any case, the correctness of an assembly is hard to verify.¹⁸² As a consequence, more refined assembly evaluation approaches are considered (e.g., the Mauve Assembly Metrics¹⁸³ for **Blue**). **BLESS** uses several assemblers from two different categories (de Bruijn and string graph-based). This approach increases the reliability of the capability to produce

valid results that do not fit a certain type of assembler (or worse, a certain assembler) and to prove the corrector's generality. Furthermore, the same authors do not just provide the value of N50; they also assess the quality of the assembly using GAGE⁶⁹ (like the authors of **Musket**).

For **BLESS**, the authors chose only artificial datasets to demonstrate the capabilities of their implementation. **DecGPU** contains the assembly information only for their corrector, not the other correctors in their benchmark. Salmela⁷³ eliminates the trimmed reads generated by **Quake** because the Illumina-only assembler Edena¹⁸⁴ can only handle reads with the same length.

Our readers can find the assembly results (where available in the original paper) in the table with the performance assessment from the Supporting Information.

Genomes Used for Testing

A recent review⁸¹ tests seven correctors on three large genomes (*H. sapiens*, *Drosophila melanogaster*, and *Caenorhabditis elegans*) among others. The datasets for the aforementioned species are very large, between 31 million and 1.7 billion reads. As discussed in Ref 108, the more complex, diploid genomes bring up the problem of heterozygosity and how to discriminate between true variants and sequencing errors. The repetitive regions also pose a problem to the corrector as mentioned by the authors of **Musket** and **HiTEC**. We can see a clear focus on the human genome as the largest and most complex datasets for benchmarking come from this organism (for **BLESS**, **Blue** and **Fiona**).

Real versus Artificial Datasets

We discern three situations for the datasets used in benchmarks: correctors tested only on simulated datasets, only on real data or both types of datasets. The main stated reasons for avoiding artificial datasets are the lack of simulators capable of producing meaningful data and the nonexistence of some real challenges that only appear in already existing real data.^{73,84} To generate the artificial data, simLibrary was used for **BLESS**, simNGS¹⁸⁵ for **Lighter**, Mason¹⁸⁶ for **Hector** and ART¹⁸⁷ for **Pluribus**. Pbsim¹⁸⁸ is cited in Ref 63, but the authors did not use it as they preferred to perform their tests on real data only. Correction software from 2013 onwards are tested mainly on data generated with dedicated software opposed to previous use of in-house mechanisms. Some authors test their work on existing artificial datasets, for example,¹⁷² for **Edar** and¹⁸⁹ for **Qamar**.

Resource Consumption

Despite the fact that early methods were not explicitly targeting a reduction in the requirements on CPU and memory consumption, all the current methods try to address this aspect. Some of the first stand-alone error correction methods were developed in Java while the latest prefer C/C++. **Blue** (2014) runs on the Microsoft .NET[®] platform while offering a very good performance. Many authors do not specify the exact method of determining the resource consumption. The authors of Refs 62,73 use the Unix time command.

There are multiple approaches to measure memory consumption. This is especially troublesome when testing programs in C/C++ against the ones in Java. For the latter, the simplest way is to measure the memory used by a process, but one must be careful with the memory allocation of the Virtual Machine (VM). In case of very small datasets, the overhead added by the VM can result in a skewed view of the real behaviour. Even the memory usage of native programs is hard to assess under Linux given the fact that there are two main memory types: virtual memory and resident set size. Most of the papers do not state how memory is measured, making comparisons difficult. We also observed a lack of information regarding the number of threads used to assess the performance in some cases. For example, the authors of Ref 55 state the use of multi-cores but does not mention how many threads were actually used in their test. The scalability of the program, that is, how the software behaves with an increasing number of threads, is often not evaluated. Some authors like Ref 97 present additional test cases in which they only assess their program, usually for very large datasets. Table 5 contains the results for a number of correctors. For the readers' convenience, we also included the results obtained in the original articles. One can see some differences in the memory and time obtained for the same dataset. This is not necessarily a sign of overinflated results in the original work but more of a difference in the testing system and how the authors prepared the data for correction. Furthermore, there is no clear indication of the version of the program used in the original benchmark and survey benchmark, respectively. The aforementioned table is a guideline for the interested user that can help him/her choose the right tool given the target hardware.

Recommendations

Using the three benchmark reviews cited earlier, we can see some correctors that emerge as the winners.

Molnar and Ilie⁸¹ consider **BLESS**, **Musket**, **RACER**, and **SGA** the best choices for HiSeq data. The last three were also able to handle *H. sapiens* datasets with over 1.3 billion reads of 100–103 bp on a Dell computer with 32 cores and a 1TB RAM. For Illumina MiSeq[®] **RACER** wins in three from a total of four tests. Tahir et al.⁷⁸ recommend **HiTEC**, **ECHO**, and **DecGPU**. In their opinion, the first two have a positive with their automatic parameter selection that can optimize performance. Finally, Yang et al.⁷⁷ obtain good results with **Reptile** and **HiTEC** for Illumina data. Unfortunately, **HiTEC** fails to run for three of seven datasets. This review also includes datasets from Roche 454 and Ion Torrent, where **Coral** wins in all cases against **HSHREC**. The two aforementioned correctors supporting indels do not obtain a high gain for Illumina datasets, making them inferior to Illumina-only correctors.

An important advice for the reader is to consider more than one metric when selecting a program. We recommend that (s)he should consider not only gain, sensitivity, and specificity but also other metrics like genome assembly and short read alignment. Another important aspect is the type of datasets used in testing. A corrector able to handle heterozygous organisms such as *H. sapiens* (like **Blue**, **Fiona**, and **BFC**) should perform well with other complex organisms. From a resource consumption perspective, **BLESS** uses the least memory, but it is single threaded and quite slow as it uses the disk. **Blue**, on the other hand, obtains some very good results in its own publication, offering a trade-off between memory and CPU consumption.

CONCLUSION

As the sequencing market share suggests, Illumina has become an important player in the industry. Our review strongly supports this claim based on the general support for this technology, with almost all programs supporting either only Illumina or Illumina plus additional sequencing technologies. Pacific Biosciences and Oxford Nanopore technologies with their (very) long reads gave birth to a new trend. This trend requires the evolution of error correction techniques to support longer reads and to deal with the high error rate that these technologies currently have. Overall, we can see improvements in the area of error correction for different technologies as the newest methods are both resource efficient and offer a very good level of correction. A reliable and structured way to measure the accuracy is also very important.

There is room for further improvement especially on the biological aspects of the correction. Here, we refer to concepts from biology like ploidy, heterozygosity, and repetitive regions not the more computer science-oriented concepts like the memory consumption, genome representation on two bits per base, and multi-core support. Now that the error correction field has been sufficiently explored, the newer methods improve over the existing ones. A not-so-favorable trend is the nonexistence of some mature-

enough methods that are constantly enriched with new features as in other related fields like assembly (Mira¹⁹⁰) or short sequence alignment (BWA¹⁸⁰).

NOTES

^a Available at <http://www.univa.com/products/grid-engine.php>

^b Available at <https://github.com/PacificBiosciences/pbdagcon>

ACKNOWLEDGMENTS

We want to thank our colleague Eloy Romero Alcala who has provided valuable advice regarding the structure of the document. This work was supported by Generalitat Valenciana [GRISOLIA/2013/013 to A.A.].

FURTHER READING

Rodriguez-Ezpeleta N, Hackenberg M, Aransay AM, eds. *Bioinformatics for High Throughput Sequencing*. Springer Science & Business Media; 2011.

<http://www.ebi.ac.uk/training/online/course/ebi-next-generation-sequencing-practical-course>. (Accessed January 18, 2015).

<https://nanoporetech.com/applications/publications>. (Accessed January 18, 2015).

REFERENCES

1. Zimin A, Stevens KA, Crepeau MW, Holtz-Morris A, Koriabine M, Marçais G, Puiu D, Roberts M, Wegrzyn JL, de Jong PJ, et al. Sequencing and assembly of the 22-Gb loblolly pine genome. *Genetics* 2014, 196:875–890.
2. Nystedt B, Street NR, Wetterbom A, Zuccolo A, Lin YC, Scofield DG, Vezzi F, Delhomme N, Giacomello S, Alexeyenko A, et al. The Norway spruce genome sequence and conifer genome evolution. *Nature* 2013, 497:579–584.
3. Friz CT. The biochemical composition of the free-living Amoebae *Chaos chaos*, *Amoeba dubia* and *Amoeba proteus*. *Comp Biochem Physiol* 1968, 26:81–90.
4. Ilie L, Fazayeli F, Ilie S. HiTEC: accurate error correction in high-throughput sequencing data. *Bioinformatics* 2011, 27:295–302.
5. Ross MG, Russ C, Costello M, Hollinger A, Lennon NJ, Hegarty R, Nusbaum C, Jaffe DB. Characterizing and measuring bias in sequence data. *Genome Biol* 2013, 14:R51.
6. Loman NJ, Misra RV, Dallman TJ, Constantinidou C, Gharbia SE, Wain J, Pallen MJ. Performance comparison of benchtop high-throughput sequencing platforms. *Nat Biotechnol* 2012, 30:434–439.
7. Kelley DR, Schatz MC, Salzberg SL. Quake: quality-aware detection and correction of sequencing errors. *Genome Biol* 2010, 11:R116.
8. marketsandmarkets.com. Next Generation Sequencing (NGS) Market by Platforms (Illumina HiSeq, MiSeq, HiSeqX Ten, NextSeq 500, Thermo Fisher Ion Proton/PGM), Bioinformatics (Exome Sequencing, RNA-Seq, ChIP-Seq), Technology (SBS, SMRT) & by Application (Diagnostics, Personalized Medicine) Global Forecast to 2020; 2014. Available at: <http://www.marketsandmarkets.com/Market-Reports/next-generation-sequencing-ngs-technologies-market-546.html>. (Accessed September 9, 2014).
9. Illumina. Human whole-genome sequencing power; 2015. Available at: <http://www.illumina.com/systems/hiseq-x-sequencing-system.html>.
10. Check Hayden E. Is the \$1,000 genome for real?; 2015. Available at: <http://www.nature.com/news/is-the-1-000-genome-for-real-1.14530>.
11. Institute NHGR. International Human Genome Sequencing Consortium Announces “Working Draft” of Human Genome; 2000. Available at: <https://www.genome.gov/10001457>.
12. Sanger F, Nicklen S, Coulson AR. DNA sequencing with chain-terminating inhibitors. *Proc Natl Acad Sci USA* 1977, 74:5463–5467.
13. Fuller CW, Middendorf LR, Benner SA, Church GM, Harris T, Huang X, Jovanovich SB, Nelson JR, Schloss JA, Schwartz DC, et al. The challenges of sequencing by synthesis. *Nat Biotechnol* 2009, 27:1013–1023.

14. Cock PJ, Fields CJ, Goto N, Heuer ML, Rice PM. The Sanger FASTQ file format for sequences with quality scores, and the Solexa/Illumina FASTQ variants. *Nucleic Acids Res* 2010, 38:1767–1771.
15. Illumina. Specification Sheet: Illumina HiSeq X Series; 2015. Available at: <http://www.illumina.com/documents/products/datasheets/datasheet-hiseq-x-ten.pdf>.
16. Illumina. Specification Sheet: HiSeq 3000/HiSeq 4000 Sequencing Systems; 2015. Available at: <https://www.illumina.com/content/dam/illumina-marketing/documents/products/datasheets/hiseq-3000-4000-specification-sheet-770-2014-057.pdf>.
17. Illumina. Specification Sheet: NextSeq Series Specifications; 2015. Available at: <http://www.illumina.com/content/dam/illumina-marketing/documents/products/datasheets/datasheet-nextseq-500.pdf>.
18. Illumina. HiSeq 2500 Specifications; 2015. Available at: http://www.illumina.com/systems/hiseq_2500_1500/performance_specifications.html.
19. Illumina. Specification Sheet: HiScan SQ System; 2012. Available at: http://www.illumina.com/documents/products/datasheets/datasheet_hiscansq.pdf.
20. Illumina. Specification Sheet: Genome AnalyzerIIx System; 2011. Available at: https://support.illumina.com/content/dam/illumina-marketing/documents/products/datasheets/datasheet_genome_analyzeriix.pdf.
21. Illumina. Specification Sheet: Genome AnalyzerIIe System; 2010. Available at: http://www.illumina.com/documents/products/datasheets/datasheet_genome_analyzer_IIe.pdf.
22. Illumina. Specification Sheet: Genome AnalyzerII System; 2009. Available at: http://tucf.org/htseq_GenomeAnalyzer_SpecSheet.pdf.
23. Illumina. Specification Sheet: Genome Analyzer System; 2007. Available at: http://www.geneworks.com.au/library/GenomeAnalyzer_SpecSheet.pdf.
24. Illumina. Specification Sheet: MiSeq System; 2015. Available at: http://www.illumina.com/content/dam/illumina-marketing/documents/products/datasheets/datasheet_miseq.pdf.
25. Scientific TF. Ion Proton System Specifications; 2015. Available at: <http://www.thermofisher.com/es/en/home/life-science/sequencing/next-generation-sequencing/ion-torrent-next-generation-sequencing-workflow/ion-torrent-next-generation-sequencing-run-sequence/ion-proton-system-for-next-generation-sequencing/ion-proton-system-specifications.html>.
26. Scientific TF. Ion PGM System Specifications; 2015. Available at: <http://www.thermofisher.com/es/en/home/life-science/sequencing/next-generation-sequencing/ion-torrent-next-generation-sequencing-workflow/ion-torrent-next-generation-sequencing-run-sequence/ion-pgm-system-for-next-generation-sequencing/ion-pgm-system-specifications.html>.
27. Scientific TF. 5500 W & 5500xl W Series; 2015. Available at: <https://www.thermofisher.com/es/en/home/life-science/sequencing/next-generation-sequencing/solid-next-generation-sequencing-systems-reagents-accessories.html>.
28. Scientific TF. 5500 & 5500xl Series; 2013. Available at: <http://www.appliedbiosystems.com/absite/us/en/home/applications-technologies/solid-next-generation-sequencing/next-generation-systems.html>.
29. Scientific TF. SOLiD System accuracy with the Exact Call Chemistry module; 2011. Available at: https://tools.thermofisher.com/content/sfs/brochures/cms_091372.pdf.
30. Biosciences P. PacBio RS II; 2015. Available at: http://files.pacb.com/pdf/PacBio_RS_II_Brochure.pdf.
31. Biosciences P. PacBio RS; 2015. Available at: http://files.pacb.com/pdf/PacBio_RS_Brochure.pdf.
32. 454R. GS FLX+ System; 2015. Available from: <http://454.com/products/gs-flx-system/index.asp>.
33. 454 R. GS Junior System; 2015. Available at: <http://454.com/products/gs-junior-system/index.asp>.
34. 454 R. GS Junior+ System; 2015. Available at: <http://454.com/products/gs-junior-plus-system/index.asp>.
35. Technologies ON. Specification MinION & PromethION; 2015. Available at: <https://www.nanoporetech.com/community/specifications>.
36. Biosystems A. Overview of SOLiD Sequencing Chemistry; 2013. Available at: <http://www.appliedbiosystems.com/absite/us/en/home/applications-technologies/solid-next-generation-sequencing/next-generation-systems/solid-sequencing-chemistry.html>.
37. Biosciences P. How does SMRT work?; 2014. Available at: <http://www.pacificbiosciences.com/products/smrt-technology/>.
38. Zhu P, Craighead HG. Zero-mode waveguides for single-molecule analysis. *Annu Rev Biophys* 2012, 41:269–293.
39. Technologies ON. DNA: Nanopore sequencing; 2015. Available at: <https://nanoporetech.com/applications/dna-nanopore-sequencing>.
40. Lieberman KR, Cherf GM, Doody MJ, Olasagasti F, Kolodji Y, Akeson M. Processive replication of single DNA molecules in a nanopore catalyzed by phi29 DNA polymerase. *J Am Chem Soc* 2010, 132:17961–17972.
41. Schulz MH, Weese D, Holtgrewe M, Dimitrova V, Niu S, Reinert K, Richard H. Fiona: a parallel and automatic strategy for read error correction. *Bioinformatics* 2014, 30:i356–i363.
42. Mikheyev AS, Tin MM. A first look at the Oxford nanopore MinION sequencer. *Mol Ecol Resour* 2014, 14:1097–1102.
43. Wirawan A, Harris RS, Liu Y, Schmidt B, Schröder J. HECTOR: a parallel multistage homopolymer

- spectrum based error corrector for 454 sequencing data. *BMC Bioinformatics* 2014, 15:131.
44. Wang XV, Blades N, Ding J, Sultana R, Parmigiani G. Estimation of sequencing error rates in short reads. *BMC Bioinformatics* 2012, 13:185.
45. Sahli M, Shibuya T. Qamar-A More Accurate DNA Sequencing Error Correcting Algorithm. *Int Proc Chem Biol Environ Eng* 2012, 31:53–58.
46. Bragg LM, Stone G, Butler MK, Hugenholtz P, Tyson GW. Shining a light on dark sequencing: characterising errors in Ion Torrent PGM data. *PLoS Comput Biol* 2013, 9:e1003031.
47. Nakamura K, Oshima T, Morimoto T, Ikeda S, Yoshikawa H, Shiwa Y, Ishikawa S, Linak MC, Hirai A, Takahashi H, et al. Sequence-specific error profile of Illumina sequencers. *Nucleic Acids Res* 2011, 39:e90.
48. Schirmer M, Ijaz UZ, D'Amore R, Hall N, Sloan WT, Quince C. Insight into biases and sequencing errors for amplicon sequencing with the Illumina MiSeq platform. *Nucleic Acids Res* 2015, 43:e37.
49. Illumina. TruSeq DNA PCR-Free Sample Preparation Kit; 2013. Available at: <http://www.appliedbiosystems.com/absite/us/en/home/applications-technologies/solid-next-generation-sequencing/next-generation-systems/solid-sequencing-chemistry.html>.
50. Biosciences S. Accel-NGS DNA Library Kit for Ion Torrent; 2015. Available at: <http://www.swiftbiosci.com/products/accel-ngs-dna-library-kit>.
51. Gen G. Fast, PCR free DNA library construction for 454; 2015. Available at: http://resource.jereci.com/10896/13041512044595_0.pdf.
52. Sleep JA, Schreiber AW, Baumann U. Sequencing error correction without a reference genome. *BMC Bioinformatics* 2013, 14:367.
53. Wijaya E, Frith MC, Suzuki Y, Horton P. Recount: expectation maximization based error correction tool for next generation sequencing data. *Genome Inform* 2009, 23:189–201.
54. Dohm JC, Lottaz C, Borodina T, Himmelbauer H. Substantial biases in ultra-short read data sets from high-throughput DNA sequencing. *Nucleic Acids Res* 2008, 36:e105.
55. Schröder J, Schröder H, Puglisi SJ, Sinha R, Schmidt B. SHREC: a short-read error correction method. *Bioinformatics* 2009, 25:2157–2163.
56. Shi H, Schmidt B, Liu W, Müller-Wittig W. A parallel algorithm for error correction in high-throughput short-read data on CUDA-enabled graphics hardware. *J Comput Biol* 2010, 17:603–615.
57. Shao W, Boltz VF, Spindler JE, Kearney MF, Maldarelli F, Mellors JW, Stewart C, Volfovsky N, Alexander Levitsky A, Stephens RM, Coffin JM. Analysis of 454 sequencing error rate, error sources, and artifact recombination for detection of low-frequency drug resistance mutations in HIV-1 DNA. *Retrovirology* 2013, 10:18.
58. Luo C, Tsementzi D, Kyrpides N, Read T, Konstantinidis KT. Direct comparisons of Illumina vs. Roche 454 sequencing technologies on the same microbial community DNA sample. *PLoS One* 2012, 7:e30087.
59. Gilles A, Meglécz E, Pech N, Ferreira S, Malausa T, Martin JF. Accuracy and quality assessment of 454 GS-FLX Titanium pyrosequencing. *BMC Genomics* 2011, 12:245.
60. Quail MA, Smith M, Coupland P, Otto TD, Harris SR, Connor TR, Bertoni A, Harold P, Swerdlow HP, Gu Y. A tale of three next generation sequencing platforms: comparison of ion torrent, pacific biosciences and Illumina MiSeq sequencers. *BMC Genomics* 2012, 13:341.
61. Koren S, Schatz MC, Walenz BP, Martin J, Howard JT, Ganapathy G, Wang Z, Rasko DA, McCombie WR, Jarvis ED, Phillippy AM. Hybrid error correction and de novo assembly of single-molecule sequencing reads. *Nat Biotechnol* 2012, 30:693–700.
62. Salmela L, Rivals E. LoRDEC: accurate and efficient long read error correction. *Bioinformatics* 2014, 30:3506–3514.
63. Hackl T, Hedrich R, Schultz J, Förster F. Proovread: large-scale high accuracy PacBio correction through iterative short read consensus. *Bioinformatics* 2014, 30:3004–3011.
64. Au KF, Underwood JG, Lee L, Wong WH. Improving PacBio long read accuracy by short read alignment. *PLoS One* 2012, 7:e46679.
65. Technologies ON. The MinION device: a miniaturised sensing; 2014. Available at: <https://nanoporetech.com/technology/the-minion-device-a-miniaturised-sensing-system/the-minion-device-a-miniaturised-sensing-system>.
66. Loman NJ, Quinlan AR. Poretools: a toolkit for analyzing nanopore sequence data. *Bioinformatics* 2014, 30:3399–3401.
67. Watson M, Thomson M, Risse J, Talbot R, Santoyo-Lopez J, Gharbi K, Blaxter M. poRe: an R package for the visualization and analysis of nanopore sequencing data. *Bioinformatics* 2014, 31:114–115.
68. Goodwin S, Gurtowski J, Ethe-Sayers S, Deshpande P, Schatz MC, McCombie WR. Oxford Nanopore sequencing, hybrid error correction, and de novo assembly of a eukaryotic genome. *Genome Res* 2015, doi: 10.1101/gr.191395.115.
69. Salzberg SL, Phillippy AM, Zimin A, Puiu D, Magoc T, Koren S, Treangen TJ, Schatz MC, Delcher AL, Roberts M, et al. GAGE: a critical evaluation of genome assemblies and assembly algorithms. *Genome Res* 2012, 22:557–567.

70. Zhao Z, Yin J, Li Y, Xiong W, Zhan Y. An efficient hybrid approach to correcting errors in short reads. *Model Decision Artif Intell* 2011;198–210.
71. Liu Y, Schröder J, Schmidt B. Musket: a multistage k-mer spectrum-based error corrector for Illumina sequence data. *Bioinformatics* 2013, 29:308–315.
72. Liu Y, Schmidt B, Maskell DL. CUSHAW: a CUDA compatible short read aligner to large genomes based on the Burrows–Wheeler transform. *Bioinformatics* 2012, 28:1830–1837.
73. Salmela L, Schröder J. Correcting errors in short reads by multiple alignments. *Bioinformatics* 2011, 27:1455–1461.
74. Qu W, Hashimoto S, Morishita S. Efficient frequency-based de novo short-read clustering for error trimming in next-generation sequencing. *Genome Res* 2009, 19:1309–1315.
75. Joppich M, Schmidl D, Bolger AM, Kuhlen T, Usadel B. PAGANtec: OpenMP parallel error correction for next-generation sequencing data. In: *OpenMP: Heterogenous Execution and Data Movements*. Switzerland: Springer; 2015, 3–17.
76. Genomeweb. Roche Shutting Down 454 Sequencing Business; 2013. Available at: <http://www.genomeweb.com/sequencing/roche-shutting-down-454-sequencing-business>.
77. Yang X, Chockalingam SP, Aluru S. A survey of error-correction methods for next-generation sequencing. *Brief Bioinform* 2013, 14:56–66.
78. Tahir M, Sardaraz M, Ikram AA, Bajwa H. Review of genome sequence short read error correction algorithms. *Am J Bioinformatics Res* 2013, 3:1–9.
79. Pevzner PA, Tang H, Waterman MS. An Eulerian path approach to DNA fragment assembly. *Proc Natl Acad Sci USA* 2001, 98:9748–9753.
80. Do CB, Batzoglou S. What is the expectation maximization algorithm? *Nat Biotechnol* 2008, 26:897–900.
81. Molnar M, Ilie L. Correcting Illumina data. *Brief Bioinform* 2015, 16:588–599.
82. Yin X, Song Z, Dorman K, Ramamoorthy A. PREMIERProbabilistic error-correction using Markov inference in errored reads. In: *2013 I.E. International Symposium on Information Theory Proceedings (ISIT)*, IEEE, 1626–1630, 2013.
83. Yin X, Song Z, Dorman K, Ramamoorthy A. PREMIER Turbo: Probabilistic error-correction using Markov inference in errored reads using the turbo principle. In: *2013 I.E. Global Conference on Signal and Information Processing (GlobalSIP)*, IEEE, 73–76, 2013.
84. Yang X, Dorman KS, Aluru S. Reptile: representative tiling for short read error correction. *Bioinformatics* 2010, 26:2526–2533.
85. Zhao X, Palmer LE, Bolanos R, Mircean C, Fasulo D, Wittenberg GM. EDAR: an efficient error detection and removal algorithm for next generation sequencing data. *J Comput Biol* 2010, 17:1549–1560.
86. Medvedev P, Scott E, Kakaradov B, Pevzner P. Error correction of high-throughput sequencing datasets with non-uniform coverage. *Bioinformatics* 2011, 27:i137–i141.
87. Yang X, Aluru S, Dorman KS. Repeat-aware modeling and correction of short read errors. *BMC Bioinformatics* 2011, 12(Suppl 1):S52.
88. Shi H, Schmidt B, Liu W, Müller-Wittig W. Quality-score guided error correction for short-read sequencing data using CUDA. *Proc Comput Sci* 2012, 1:1129–1138.
89. Shah AR, Chockalingam S, Aluru S. A parallel algorithm for spectrum-based short read error correction. In: *2012 I.E. 26th International Parallel & Distributed Processing Symposium (IPDPS)*. IEEE, 60–70, 2012.
90. Nikolenko SI, Korobeynikov AI, Alekseyev MA. BayesHammer: Bayesian clustering for error correction in single-cell sequencing. *BMC Genomics* 2013, 14(Suppl 1):S7.
91. Marçais G, Yorke JA, Zimin A. QuorUM: an error corrector for Illumina reads. arXiv preprint arXiv:13073515; 2013.
92. Ilie L, Molnar M. RACER: Rapid and accurate correction of errors in reads. *Bioinformatics* 2013, 29:2490–2493.
93. Song L, Florea L, Langmead B. Lighter: fast and memory-efficient error correction without counting. *Genome Biology* 2014, 15:509.
94. Milicchio F, Prosperi MC. HErCoOl: High-throughput Error Correction by Oligomers. In: *2014 I.E. 27th International Symposium on Computer-Based Medical Systems (CBMS)*. IEEE, 227–232, 2014.
95. Lim EC, Müller J, Hagmann J, Henz SR, Kim ST, Weigel D. Trowel: a fast and accurate error correction module for Illumina sequencing reads. *Bioinformatics* 2014, 30:3264–3265.
96. Heo Y, Wu XL, Chen D, Ma J, Hwu WM. BLESS: bloom filter-based error correction solution for high-throughput sequencing reads. *Bioinformatics* 2014, 30:1354–1362.
97. Greenfield P, Duesing K, Papanicolaou A, Bauer DC. Blue: correcting sequencing errors using consensus and context. *Bioinformatics* 2014, 30:2723–2732.
98. Li H. Correcting Illumina sequencing errors for human data. arXiv preprint arXiv:150203744; 2015.
99. Duma D, Cordero F, Beccuti M, Ciardo G, Close TJ, Lonardi S. Scribe: ultra-accurate error-correction of pooled sequenced reads. In: *Algorithms in Bioinformatics*. Berlin/Heidelberg: Springer; 2015, 162–174.
100. Sheikhezadeh S, de Ridder D. ACE: Accurate Correction of Errors using K-mer tries. *Bioinformatics* 2015, 31:3216–3218.

101. Ramachandran A, Heo Y, Hwu Wm, Ma J, Chen D. FPGA accelerated DNA error correction. In: *Proceedings of the 2015 Design, Automation & Test in Europe Conference & Exhibition*. EDA Consortium, 1371–1376, 2015.
102. Marinier E, Brown DG, McConkey BJ. Pollux: platform independent error correction of single and mixed genomes. *BMC Bioinformatics* 2015, 16:10.
103. Gu Y, Liu X, Zhu Q, Dong Y, Brown CT, Pramanik S. A new method for DNA sequencing error verification and correction via an on-disk index tree. In: *Proceedings of the 6th ACM Conference on Bioinformatics, Computational Biology and Health Informatics*. ACM, 503–504, 2015.
104. Salmela L. Correction of sequencing errors in a mixed set of reads. *Bioinformatics* 2010, 26:1284–1290.
105. Zhao Z, Yin J, Zhan Y, Xiong W, Li Y, Liu F. PSAEC: an improved algorithm for short read error correction using partial suffix arrays. *Front Algorithm Asp Inform Manage* 2011:220–232.
106. Liu Y, Schmidt B, Maskell DL. DecGPU: distributed error correction on massively parallel graphics processing units using CUDA and MPI. *BMC Bioinformatics* 2011, 12:85.
107. Savel DM, LaFramboise T, Grama A, Koyutürk M. Suffix-tree based error correction of NGS reads using multiple manifestations of an error. In: *Proceedings of the International Conference on Bioinformatics, Computational Biology and Biomedical Informatics*. ACM, 351, 2013.
108. Kao WC, Chan AH, Song YS. ECHO: a reference-free short-read error correction algorithm. *Genome Res* 2011, 21:1181–1192.
109. Chen CC, Chang YJ, Chung WC, Lee DT, Ho JM. CloudRS: An error correction algorithm of high-throughput sequencing data based on scalable framework. In: *2013 International Conference on Big Data IEEE*. IEEE, 717–722, 2013.
110. Chung WC, Chang YJ, Lee D, Ho JM. Using geometric structures to improve the error correction algorithm of high-throughput sequencing data on MapReduce framework. In: *2014 I.E. International Conference on Big Data (Big Data)*. IEEE, 784–789, 2014.
111. Allam A, Kalnis P, Solovyev V. Karect: accurate correction of substitution, insertion and deletion errors for next-generation sequencing data. *Bioinformatics* 2015, 31:3421–3428.
112. Miclotte G, Heydari M, Demeester P, Audenaert P, Fostier J. Jabba: hybrid error correction for long sequencing reads using maximal exact matches. In: *Algorithms in Bioinformatics*. Berlin/Heidelberg: Springer; 2015, 175–188.
113. Artyomenko A, Mancuso N, Skums P, Mandoiu I, Zelikovskiy A. kGEM: an expectation maximization error correction algorithm for next generation sequencing of amplicon-based data. In: *Proceedings of International Symposium Bioinformatics Research and Applications*, 2013.
114. Shen J, Cong Q, Grishin NV. The complete mitochondrial genome of *Papilio glaucus* and its phylogenetic implications. *Meta Gene* 2015, 5:68–83.
115. Wang C, Grohme MA, Mali B, Schill RO, Frohme M. Towards decrypting cryptobiosis-analyzing anhydrobiosis in the tardigrade milnesium tardigradum using transcriptome sequencing. *PLoS One* 2014, 9:e92663.
116. MacManes MD, Eisen MB. Characterization of the transcriptome, nucleotide sequence polymorphism, and natural selection in the desert adapted mouse *Peromyscus eremicus*. *PeerJ* 2014, 2:e642.
117. Alkio M, Jonas U, Declercq M, Van Nocker S, Knoche M. Transcriptional dynamics of the developing sweet cherry (*Prunus avium* L.) fruit: sequencing, annotation and expression profiling of exocarp-associated genes. *Horticult Res* 2014, 1.
118. Gupta V, Markmann K, Pedersen CN, Stougaard J, Andersen SU. shorttran: a pipeline for small RNA-seq data analysis. *Bioinformatics* 2012, 28:2698–2700.
119. Henkel CV, Dirks RP, Jansen HJ, Forlenza M, Wiegertjes GF, Howe K, van den Thillart GEEJM, Spaink HP. Comparison of the exomes of common carp (*Cyprinus carpio*) and zebrafish (*Danio rerio*). *Zebrafish* 2012, 9:59–67.
120. Dominova I, Sorokin D, Kublanov I, Patrushev M, Toshchakov S. Complete genome sequence of *Salinarchaeum* sp. strain HArchT-Bsk1T, isolated from hypersaline Lake Baskunchak, Russia. *Genome Announc* 2013, 1:e00505–e00513.
121. Riedel T, Fiebig A, Petersen J, Gronow S, Kyrpides NC, Göker M, Klenk HP. Genome sequence of the *Litoreibacter arenae* type strain (DSM 19593T), a member of the *Roseobacter* clade isolated from sea sand. *Stand Genomic Sci* 2013, 9:117.
122. Xu Q, Chen LL, Ruan X, Chen D, Zhu A, Chen C, Bertrand D, Jiao WB, Hao BH, Lyon MP, et al. The draft genome of sweet orange (*Citrus sinensis*). *Nat Genet* 2013, 45:59–66.
123. Jiao WB, Huang D, Xing F, Hu Y, Deng XX, Xu Q, Chen LL. Genome-wide characterization and expression analysis of genetic variants in sweet orange. *Plant J* 2013, 75:954–964.
124. Lada AG, Stepchenkova EI, Waisertreiger I, Noskov VN, Dhar A, Eudy JD, Boissy RJ, Hirano M, Rogozin IB, Pavlov YI. Genome-wide mutation avalanches induced in diploid yeast cells by a base analog or an APOBEC deaminase. *PLoS Genet* 2013, 9:e1003736.
125. Fujimoto MS, Bodily PM, Okuda N, Clement MJ, Snell Q. Effects of error-correction of heterozygous next-generation sequencing data. *BMC Bioinformatics* 2014, 15(Suppl 7):S3.

126. Schatz MC, Maron LG, Stein JC, Wences AH, Gurtowski J, Biggers E, Lee H, Kramer M, Antoniou E, Ghiban E, et al. Whole genome de novo assemblies of three divergent strains of rice, *Oryza sativa*, document novel gene space of aus and indica. *Genome Biol* 2014, 15:506.
127. Coates RC, Podell S, Korobeynikov A, Lapidus A, Pevzner P, Sherman DH, Allen EE, Gerwick L, Gerwick WH. Characterization of cyanobacterial hydrocarbon composition and distribution of biosynthetic pathways. *PLoS One* 2014, 9:e85140.
128. Rödelberger C, Neher RA, Weller AM, Eberhardt G, Witte H, Mayer WE, Dieterich C, Sommer RJ. Characterization of genetic diversity in the nematode *Pristionchus pacificus* from population-scale resequencing data. *Genetics* 2014, 196:1153–1165.
129. Nguyen TT, Hayes BJ, Ingram BA. Genetic parameters and response to selection in blue mussel (*Mytilus galloprovincialis*) using a SNP-based pedigree. *Aquaculture* 2014, 420:295–301.
130. Nobu MK, Tamaki H, Kubota K, Liu WT. Metagenomic characterization of *Candidatus Defluviicoccus tetraformis* strain TFO71TM, a tetrad-forming organism, predominant in an anaerobic-aerobic membrane bioreactor with deteriorated biological phosphorus removal. *Environ Microbiol* 2014, 16:2739–2751.
131. Suzuki S, Horinouchi T, Furusawa C. Prediction of antibiotic resistance by gene expression profiles. *Nat Commun* 2014, 5.
132. Hill-Cawthorne GA, Hudson LO, El Ghany MFA, Piepenburg O, Nair M, Dodgson A, Forrest MS, Clark TG, Pain A. Recombinations in staphylococcal cassette chromosome *mec* elements compromise the molecular detection of methicillin resistance in *Staphylococcus aureus*. *PLoS One* 2014, 9:e101419.
133. Koch P, Platzer M, Downie BR. RepARK-de novo creation of repeat libraries from whole-genome NGS reads. *Nucleic Acids Res* 2014, 42:e80.
134. Busk PK, Lange M, Pilgaard B, Lange L. Several genes encoding enzymes with the same activity are necessary for aerobic fungal degradation of cellulose in nature. *PLoS One* 2014, 9:e114138.
135. Gilchrist AS, Shearman DC, Frommer M, Raphael KA, Deshpande NP, Wilkins MR, Sherwin WB, Sved JA. The draft genome of the pest tephritid fruit fly *Bactrocera tryoni*: resources for the genomic analysis of hybridising species. *BMC Genomics* 2014, 15:1153.
136. Burke GR, Walden KKO, Whitfield JB, Robertson HM, Strand MR. Widespread genome reorganization of an obligate virus mutualist. *PLoS Genet* 2014, 10:e1004660.
137. Kenny NJ, Namigai EK, Marlétaz F, Hui JH, Shimeld SM. Draft genome assemblies and predicted microRNA complements of the intertidal lophotrochozoans *Patella vulgata* (Mollusca, Patellogastropoda) and *Spirobranchus lamarcki* (Annelida, Serpulida). *Mar Genomics* 2015, 24:139–146.
138. Fitak RR, Mohandesan E, Corander J, Burger PA. The de novo genome assembly and annotation of a female domestic dromedary of North African origin. *Mol Ecol Resour* 2016, 16:314–324.
139. Le Duc D, Renaud G, Krishnan A, Almén MS, Huynen L, Prohaska SJ, Ongyerth M, Bitarello BD, Schiöth HB, Hofreiter M, et al. Kiwi genome provides insights into evolution of a nocturnal lifestyle. *Genome Biol* 2015, 16:1–15.
140. Fiebig A, Loof TG, Babbar A, Itzek A, Koehorst JJ, Schaap PJ, Nitsche-Schmitz DP. Comparative genomics of *Streptococcus pyogenes* M1 isolates differing in virulence and propensity to cause systemic infection in mice. *Int J Med Microbiol* 2015, 305:532–543.
141. Lambert D, Carrillo CD, Koziol AG, Manninger P, Blais BW. GeneSippr: a rapid whole-genome approach for the identification and characterization of foodborne pathogens such as Priority Shiga Toxigenic *Escherichia coli*. *PLoS One* 2015, 10:e0122928.
142. Cong Q, Borek D, Otwinowski Z, Grishin NV. Tiger swallowtail genome reveals mechanisms for speciation and caterpillar chemical defense. *Cell Rep* 2015, 10:910–919.
143. Jnemann S, Prior K, Albersmeier A, Albaum S, Kalinowski J, Goesmann A, Stoye J, Harmsen D. GABenchToB: a genome assembly benchmark tuned on bacteria and benchtop sequencers. *PLoS One* 2014, 9:e107014.
144. Walter MC, Öhrman C, Myrtenäs K, Sjödin A, Byström M, Larsson P, Macellaro A, Forsman M, Frangoulidis D. Genome sequence of *Coxiella burnetii* strain Namibia. *Stand Genomic Sci* 2014, 9:22.
145. Nikolaichik Y, Gorshkov V, Gogolev Y, Valentovich L, Evtushenkov A. Genome sequence of *Pectobacterium atrosepticum* strain 21A. *Genome Announc* 2014, 2.
146. Engel P, Stepanauskas R, Moran NA. Hidden diversity in honey bee gut symbionts detected by single-cell genomics. *PLoS Genet* 2014, 10:e1004596.
147. Aeschlimann SH, Jönsson F, Postberg J, Stover NA, Petera RL, Lipps HJ, Nowacki M, Swart EC. The draft assembly of the radically organized *Stylonychia lemnae* macronuclear genome. *Genome Biol Evol* 2014, 6:1707–1723.
148. Kleigrewe K, Almaliti J, Tian IY, Kinnel RB, Korobeynikov A, Monroe EA, Duggan BM, Di Marzo V, Sherman DH, Dorrestein PC, Gerwick L, Gerwick WH. Combining mass spectrometric metabolic profiling with genomic analysis: a powerful

- approach for discovering natural products from cyanobacteria. *J Nat Prod* 2015, 78:1671–1682.
149. Grob C, Taubert M, Howat AM, Burns OJ, Dixon JL, Richnow HH, Jehmlich N, von Bergen M, Chen Y, Murrell JC. Combining metagenomics with metaproteomics and stable isotope probing reveals metabolic pathways used by a naturally occurring marine methylotroph. *Environ Microbiol* 2015, 17:4007–4018.
150. Neumann AM, Balmonte JP, Berger M, Giebel HA, Arnosti C, Voget S, Simon M, Brinkhoff T, Wietz M. Different utilization of alginate and other algal polysaccharides by marine *Alteromonas macleodii* ecotypes. *Environ Microbiol* 2015, 17:3857–3868.
151. Lada AG, Kliver SF, Dhar A, Polev DE, Masharsky AE, Rogozin IB, Pavlov YI. Disruption of transcriptional coactivator Sub1 leads to genome-wide re-distribution of clustered mutations induced by APOBEC in active yeast genes. *PLoS Genet* 2015, 11:e1005217.
152. Boudreau PD, Monroe EA, Mehrotra S, Desfor S, Korobeynikov A, Sherman DH, Murray TF, Gerwick L, Dorrestein PC, Gerwick WH. Expanding the described metabolome of the marine Cyanobacterium *moorea producens* JHB through orthogonal natural products workflows. *PLoS One* 2015, 10:e0133297.
153. De Wit P, Pespeni MH, Palumbi SR. SNP genotyping and population genomics from expressed sequences—current advances and future possibilities. *Mol Ecol* 2015, 24:2310–2323.
154. Taniguti LM, Schaker PD, Benevenuto J, Peters LP, Carvalho G, Palhares A, Quecine MC, Nunes FRS, Kmit MCP, Wai A, et al. Complete genome sequence of *Sporisorium scitamineum* and biotrophic interaction transcriptome with sugarcane. *PLoS One* 2015, 10:e0129318.
155. Li X, Fan D, Zhang W, Liu G, Zhang L, Zhao L, Fang X, Lei Chen L, Dong Y, Chen Y, et al. Outbred genome sequencing and CRISPR/Cas9 gene editing in butterflies. *Nat Commun* 2015, 6.
156. D'Agostino N, Golas T, Van de Geest H, Bombarely A, Dawood T, Zethof J, Driedonks N, Wijnker E, Bargsten J, Nap JP, et al. Genomic analysis of the native European *Solanum* species, *S. dulcamara*. *BMC Genomics* 2013, 14:356.
157. Ištváněk J, Jaroš M, Křenek A, Řepková J. Genome assembly and annotation for red clover (*Trifolium pratense*; Fabaceae). *Am J Bot* 2014, 101:327–337.
158. Yang R, Dai Z, Chen S, Chen L. MicroRNA-mediated gene regulation plays a minor role in the transcriptomic plasticity of cold-acclimated zebrafish brain tissue. *BMC Genomics* 2011, 12:605.
159. Zawada AM, Rogacev KS, Rotter B, Winter P, Marell RR, Fliser D, Heine GH. SuperSAGE evidence for CD14⁺⁺ CD16⁺ monocytes as a third monocyte subset. *Blood* 2011, 118:e50–e61.
160. Roccaro M, Ahmadinejad N, Colby T, Somssich IE. Identification of functional cis-regulatory elements by sequential enrichment from a randomized synthetic DNA library. *BMC Plant Biol* 2013, 13:164.
161. Ollier M, Radosevic-Robin N, Kwiatkowski F, Ponelle F, Viala S, Privat M, Uhrhammer N, Bernard-Gallon D, Penault-Llorca F, Bignon YJ, Bidet Y. DNA repair genes implicated in triple negative familial non-BRCA1/2 breast cancer predisposition. *Am J Cancer Res* 2015, 5:2113.
162. Fertin G, Jean G, Radulescu A, Rusu I. Hybrid de novo tandem repeat detection using short and long reads. *BMC Med Genomics* 2015, 8(Suppl 3):S5.
163. Yan L, Wang X, Liu H, Tian Y, Lian J, Yang R, Hao S, Wang X, Yang S, Li Q, et al. The Genome of *Dendrobium officinale* Illuminates the Biology of the Important Traditional Chinese Orchid Herb. *Mol Plant* 2014, 8:922–934.
164. Weirather JL, Afshar PT, Clark TA, Tseng E, Powers LS, Underwood J, Zabner J, Korlach J, Wong WH, Au KF. Characterization of fusion genes and the significantly expressed fusion isoforms in breast cancer by hybrid sequencing. *Nucleic Acids Res* 2015, 43:e116.
165. Mulley JF, Hargreaves AD. Snake venom gland cDNA sequencing using the Oxford Nanopore MinION portable DNA sequencer. *bioRxiv* 2015:025148.
166. Engström PG, Tommei D, Stricker SH, Ender C, Pollard SM, Bertone P. Digital transcriptome profiling of normal and glioblastoma-derived neural stem cells identifies genes associated with patient survival. *Genome Med* 2012, 4:1–20.
167. Cox MP, Peterson DA, Biggs PJ. SolexaQA: at-a-glance quality assessment of Illumina second-generation sequencing data. *BMC Bioinformatics* 2010, 11:485.
168. Rizk G, Lavenier D, Chikhi R. DSK: k-mer counting with very low memory usage. *Bioinformatics* 2013, 29:652–653.
169. Deorowicz S, Debudaj-Grabysz A, Grabowski S. Disk-based k-mer counting on a PC. *BMC Bioinformatics* 2013, 14:160.
170. Marçais G, Kingsford C. A fast, lock-free approach for efficient parallel counting of occurrences of k-mers. *Bioinformatics* 2011, 27:764–770.
171. Zerbino DR, Birney E. Velvet: algorithms for de novo short read assembly using de Bruijn graphs. *Genome Res* 2008, 18:821–829.
172. Chaisson M, Pevzner P, Tang H. Fragment assembly with short reads. *Bioinformatics* 2004, 20:2067–2074.
173. Shi H, Schmidt B, Liu W, Muller-Wittig W. Accelerating error correction in high-throughput short-read DNA sequencing data with CUDA. In: *IPDPS*

- 2009 I.E. *International Symposium on Parallel & Distributed Processing*. IEEE, 1–8, 2009.
174. Fichot EB, Norman RS. Microbial phylogenetic profiling with the Pacific Biosciences sequencing platform. *Microbiome* 2013, 1:10.
175. Bentley DR, Balasubramanian S, Swerdlow HP, Smith GP, Milton J, Brown CG, Hall KP, Evers DJ, Barnes CL, Bignell HR, et al. Accurate whole human genome sequencing using reversible terminator chemistry. *Nature* 2008, 456:53–59.
176. Hillier LW, Marth GT, Quinlan AR, Dooling D, Fewell G, Barnett D, Fox P, Glasscock JI, Hickenbotham M, Huang W, et al. Whole-genome sequencing and variant discovery in *C. elegans*. *Nat Methods* 2008, 5:183–188.
177. Kozarewa I, Ning Z, Quail MA, Sanders MJ, Berriman M, Turner DJ. Amplification-free Illumina sequencing-library preparation facilitates improved mapping and assembly of (G+ C)-biased genomes. *Nat Methods* 2009, 6:291–295.
178. Yoo AB, Jette MA, Grondona M. SLURM: Simple linux utility for resource management. In: *Job Scheduling Strategies for Parallel Processing*. Berlin/Heidelberg: Springer; 2003, 44–60.
179. Li R, Li Y, Kristiansen K, Wang J. SOAP: short oligonucleotide alignment program. *Bioinformatics* 2008, 24:713–714.
180. Li H, Durbin R. Fast and accurate short read alignment with Burrows–Wheeler transform. *Bioinformatics* 2009, 25:1754–1760.
181. Miller JR, Koren S, Sutton G. Assembly algorithms for next-generation sequencing data. *Genomics* 2010, 95:315–327.
182. Bradnam KR, Fass JN, Alexandrov A, Baranay P, Bechner M, Birol I, Boisvert S, Chapman JA, Chapuis G, Chikhi R, et al. Assemblathon 2: evaluating de novo methods of genome assembly in three vertebrate species. *GigaScience* 2013, 2:1–31.
183. Darling AE, Tritt A, Eisen JA, Facciotti MT. Mauve assembly metrics. *Bioinformatics* 2011, 27:2756–2757.
184. Hernandez D, François P, Farinelli L, Østerås M, Schrenzel J. De novo bacterial genome sequencing: millions of very short reads assembled on a desktop computer. *Genome Res* 2008, 18:802–809.
185. Massingham T, Goldman N. simNGS and simLibrary—software for simulating next-gen sequencing data; 2012.
186. Holtgrewe M. Mason—a read simulator for second generation sequencing data. Technical Report FU Berlin; 2010.
187. Huang W, Li L, Myers JR, Marth GT. ART: a next-generation sequencing read simulator. *Bioinformatics* 2012, 28:593–594.
188. Ono Y, Asai K, Hamada M. PBSIM: PacBio reads simulator toward accurate genome assembly. *Bioinformatics* 2013, 29:119–121.
189. Dohm JC, Lottaz C, Borodina T, Himmelbauer H. SHARCGS, a fast and highly accurate short-read assembly algorithm for de novo genomic sequencing. *Genome Res* 2007, 17:1697–1706.
190. Chevreux B, Wetter T, Suhai S. Genome sequence assembly using trace signals and additional sequence information. In: *German Conference on Bioinformatics*, 45–56, 1999.