Genome Biology

METHOD

Open Access

ReSeq simulates realistic Illumina high-throughput sequencing data

Stephan Schmeing[1,2*] and Mark D. Robinson[1,2*]

*Correspondence:
stephan.schmeing@uzh.ch;
mark.robinson@mls.uzh.ch
[1]Institute of Molecular Life Sciences, University of Zurich, Winterthurerstrasse 190, 8057 Zurich, Switzerland
[2]SIB Swiss Institute of Bioinformatics, Winterthurerstrasse 190, 8057 Zurich, Switzerland

Abstract

In high-throughput sequencing data, performance comparisons between computational tools are essential for making informed decisions at each step of a project. Simulations are a critical part of method comparisons, but for standard Illumina sequencing of genomic DNA, they are often oversimplified, which leads to optimistic results for most tools. ReSeq improves the authenticity of synthetic data by extracting and reproducing key components from real data. Major advancements are the inclusion of systematic errors, a fragment-based coverage model and sampling-matrix estimates based on two-dimensional margins. These improvements lead to more faithful performance evaluations. ReSeq is available at https://github.com/schmeing/ReSeq.

Keywords: Simulation, Genomic, High-throughput sequencing, Illumina

Background

High-throughput sequencing has revolutionized biology and medicine since it allows a myriad of applications, such as studying entire genomes at base-pair resolution. The accuracy of the obtained results after applying computational methods heavily depends on collected data and the tools used to process it. This paper focuses on standard Illumina short-read sequencing of genomic DNA (gDNA) and its BGI counterpart [1, 2], which are both obtainable for almost every molecular biology lab. In order to fully capitalize on these datasets, it is important to know the best tools for a given task, the typical error modes of these tools, and whether a result is robust to fluctuations in the data or changes in the analysis.

With the ever-growing number of computational tools, evaluating their performance across the various situations in which they are applied has become an essential part of bioinformatics [3, 4]. There are two fundamental ways of doing benchmarks and validations. On the one hand, results can be compared to an *estimated* "gold-standard" ground truth derived from real data, which can be based on consensus or an independent dataset (e.g., technology). On the other hand, tools can be compared on synthetic data, which

are simulated from a specified ground truth that defines the desired results. Both strategies can introduce biases, through deriving the ground truth or by failing to mimic the properties of real data, respectively. Ideally, a mix of assessments from real and synthetic data is used, where differences in the results can highlight biases and provide estimates of uncertainty.

On real data, performance evaluations require a ground truth and estimating it with methods similar or identical to evaluated methods can induce a bias. To reduce such a bias, evaluations often only take into account situations where a confident consensus can be determined or where an alternative technology delivers reliable results [5]. This limitation necessarily reduces the breadth of method comparisons. Sometimes, these shortcomings can be mitigated with deeper sequencing, by using multiple alternative technologies and by carefully selecting the set of methods that go into the estimation. For example, extensive effort went into generating datasets with ground truth to assess variant calling on human gDNA datasets [6], such as the Genome in a Bottle Consortium [7, 8] and Platinum Genomes [9]. Their detailed variant truth sets were derived using a consensus from multiple aligner/variant caller combinations. Having multiple technologies or pedigree information further increases the confidence in their ground truth calls. Alternatively, as an example of consensus-free approaches, Li et al. used Pacific Biosciences SMRT sequencing (PacBio) to create two independent assemblies, each of a homozygous human cell line. The combined assemblies provide the ground truth for a synthetic diploid dataset [10]. The advantage is that no variant callers are used to estimate the truth, while the disadvantage is that PacBio-specific errors remain. Despite the effort that went into the three mentioned truth sets, their results do not agree on variant caller performance, neither by value (e.g., false-positive rates of single-nucleotide polymorphisms) nor by rank [10]. Due to an abundance of truth sets in this field, the uncertainty in the comparisons can at least be assessed with real data alone. In other areas of research, often no published datasets with an estimated ground truth are readily available. For example, assessing the influence of polyploidy on variant calls using real data is limited to concordance checks [11].

Simulated data are a cheap and orthogonal way to benchmark computational methods and can readily address the shortcomings of real data based evaluations (e.g., biases toward certain tools or against certain genomic regions). Additionally, robustness toward properties of data (e.g., error rates) can be easily assessed. However, accurate method assessments require that simulated data recapitulate the important features of real data and do not oversimplify or bias the challenge for tested methods. Despite many published simulators, research comparing simulations has so far neglected to test for these important features. Reviews include Escalona et al. [12], which did not show any benchmarks, and Alosaimi et al. [13], which based the performance report on sensitivity and precision of mapping (of simulated reads). Unfortunately, this metric says little about the quality of the simulation; for example, a simulator that samples from unique regions of the reference and does not include any errors would receive a perfect score. A proper benchmark requires in-depth testing across a range of use-cases, since the most important features to mimic from real data depend on the application. We assess here many aspects of real data and in particular whether the key features for assembly have been reproduced. Furthermore, we show using the example of mapping how to evaluate the scope of simulations in a benchmark.

For Illumina gDNA datasets, the simulation frameworks ART [14], pIRS [15], and NEAT [16] represent the state-of-the-art. Additionally, BEAR [17] was included to evaluate the effect of their design choices that simplify metagenomic simulations. We show below that all current simulators do an unsatisfactory job of reproducing, e.g., the k-mer spectrum of real data, due to their incomplete models for coverage, quality, and base calling. As a result, methods tested on simulated data score nearly perfectly [18], which has presumably encouraged the field to rely on real data alone for evaluations; for example, the Assemblathon 1 [19] used simulations, while GAGE [20] and Assemblathon 2 [21] in the following years used real data alone.

Table 1 lists the features and input for each of the simulators compared in this study. The two main simulation components are the coverage model and the quality and base-call model. Coverage in ART and BEAR is modelled uniformly, while pIRS and NEAT introduce a GC bias by comparing the coverage across the binned reference [22, 23]. This procedure is close to the Loess model described by Benjamini and Speed [24], who show that their alternative model based on the GC of individual fragments results in superior predictions. Furthermore, the bias from the sequences *flanking* the start and end of fragments [25] are not taken into account by any simulator. Finally, ART, pIRS, and BEAR draw DNA fragment lengths from a user-defined Gaussian distribution, while NEAT uses the empirical distribution from the input bam file.

For the qualities and base-calls, ART draws from empirical distributions of position-dependent qualities and introduces substitution errors [26–28] according to the probability given by the quality values. InDels are inserted based on four user-specified rates for insertion/deletion in the first/second read. In contrast, pIRS, NEAT, and BEAR draw quality values from a non-homogeneous Markov chain, where the quality depends on the last quality and the position in the read. pIRS then chooses a base call from a learned distribution depending on the quality, position, and reference base, while the inserted InDels depend only on the position in the read. NEAT instead follows a decision tree, where the occurrences of errors (substitution and InDels) only depends on the quality and the substituted nucleotide only on the reference base, while the length and nucleotides of InDels

Table 1 Overview of modelled features for the compared simulators

	ART	pIRS	NEAT	BEAR	ReSeq
Coverage	Parameters	Parameters	Parameters	Parameters	Mapped reads/ Parameters
GC bias	–	Mapped reads	Mapped reads	–	Mapped reads
Flanking bias	–	–	–	–	Mapped reads
Fragment length	Parameters	Parameters	Mapped reads	Parameters	Mapped reads
Reference-sequence bias	–	–	–	Hard-coded/(Reads)	Mapped reads
Base qualities	Reads	Mapped reads	Reads	Reads	Mapped reads
Sequence qualities	–	–	-	–	Mapped reads
Substitution errors	Hard-coded	Mapped reads	Hard-coded	Reads	Mapped reads
Systematic errors	–	–	–	–	Mapped reads
InDel errors	Parameters	Mapped reads	Hard-coded	Reads	Mapped reads
Variants	–	Two genomes	Vcf	–	Vcf

Parameters: Manually selected parameters. Reads: Learned from raw reads. Mapped reads: Learned from mapped reads. Hard-coded: Cannot be changed. BEAR's reference-sequence bias estimation from reads is design for metagenomics

have constant probabilities. It is worth mentioning that in the current version of NEAT, only qualities can be trained, but the base-call and InDel distributions rely on pretrained values. Specifically for metagenomics, BEAR's error model is learned from duplicated reads using DRISEE [29] and therefore does not require a reference. Its parameters are obtained from exponential regression on the substitution rates by nucleotide and position and on the InDel rates by position. A peculiar design choice of BEAR is to replace quality values at error positions with qualities generated by the error model instead of simply having error rates depend on the quality values. Notably, neither systematic, sequence-specific errors [30, 31] nor the relationship between quality and fragment length [32] are included by any simulator.

The ability for users to train simulators on real data is another important feature, because profiles require constant updates to changes in sequencers, chemistries, etc., and even without technology changes, developer-provided profiles are not always accurate for a given use case due to differences in genome contexts or fragmentation method (see Results). None of the simulators mentioned can be fully trained on real data, instead relying at least partially on user-provided parameters or hard-coded models (Table 1).

Real Sequence Reproducer provides well-tested functionality to estimate the necessary parameters from real data mapped to a reference. Based on these estimates, it produces synthetic data with a k-mer spectrum matching real data without ever directly using k-mer information (see below). Requiring a reference is not a big constraint, since one is needed for the simulation anyways and furthermore, with a modest penalty in accuracy, the ReSeq parameters can be estimated from a de novo assembly generated from the reads.

We show that ReSeq outperforms all competitors in terms of delivering a realistic simulation and therefore lays the methodological groundwork for accurate benchmarking of genomics tools.

Results

ReSeq consists of three parts: statistics calculation, probability estimation, and simulation (Fig. 1). Statistics calculation extracts the necessary information from the mapped reads and the corresponding reference. Afterwards, probability estimation combines the extracted matrices into distributions. Finally, the simulation step draws from those distributions.

For the statistics calculation, a file with variants can be specified, such that their positions in the reference are excluded from the statistics. Adapters can optionally be specified and are automatically detected otherwise.

The simulation produces synthetic data matching the calculated statistics and estimated biases. The reference provided can but does not need to be the same as the one used during the statistics calculation. To impose a clear separation, we will refer to the reference we simulate from as the template. To simulate single-end reads, the second read file can simply be ignored; however, paired-end Illumina data are still required for the statistics calculation. To properly handle coverage variations for sex chromosomes, mitochondria, or metagenomics, the simulation optionally takes a reference-bias file (not necessary if template and reference are identical). To simulate diploid and polyploid genomes or pooled sequencing, variants can be specified. To simulate bisulfite sequencing, allele-specific methylation values can be defined in an extended bed graph format with multiple

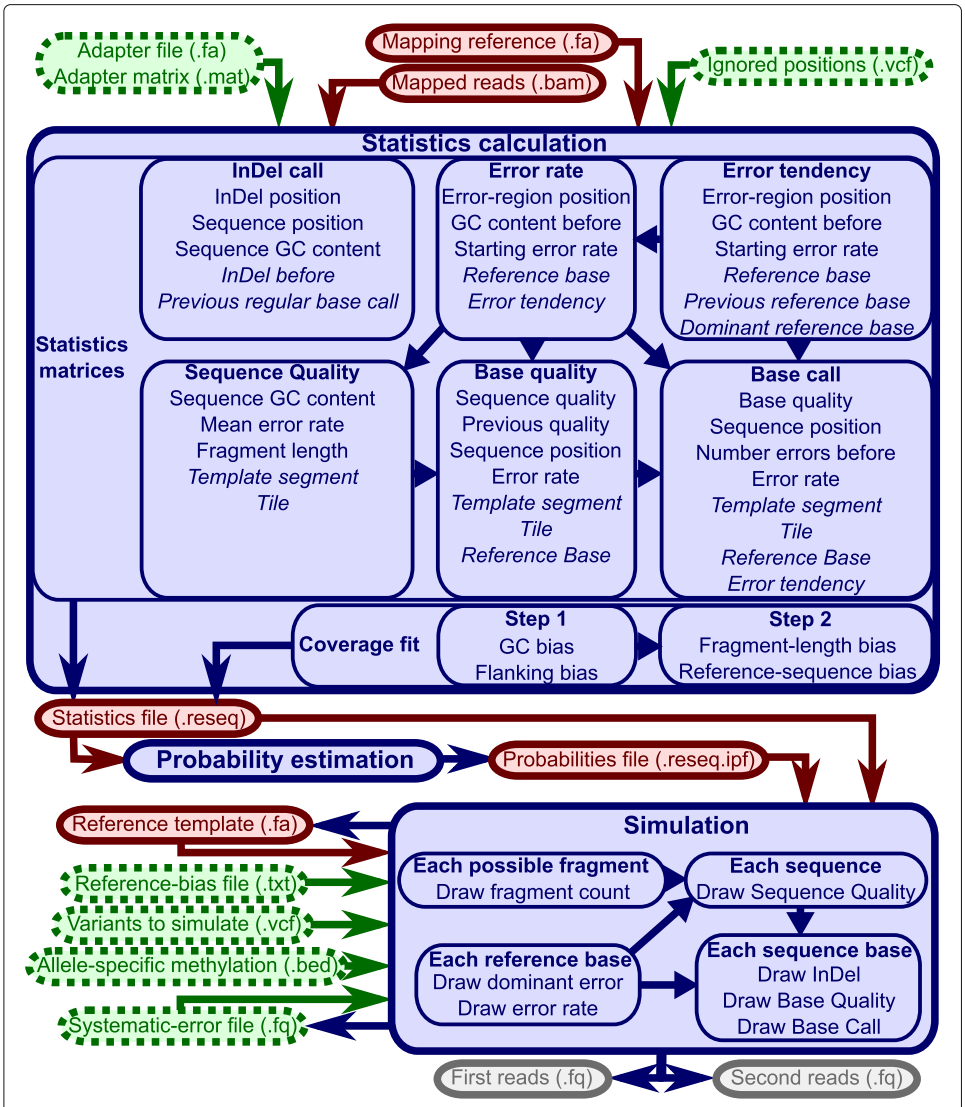

Fig. 1 ReSeq overview. The three parts of ReSeq (solid, blue) with its mandatory (solid, red) and optional (dotted, green) inputs. The italic entries for the statistics matrices are independent dimensions in the matrix, while the normal entries are reduced to two-dimensional interactions

score columns. However, we focus here on monoploid and diploid genomes and will not include simulations of bisulfite sequencing. For comparisons, we use ten representative datasets from different species, different Illumina machines and different adapters (Table 2). Additionally, we included a dataset from BGI [1, 2] to investigate whether Illumina simulators can also handle this related technology.

Generating qualities and base calls

To simulate quality values and base calls, ReSeq fills six matrices during the statistics calculation (Fig. 1): insertions and deletions, systematic error rates at each reference position, systematic error tendencies at each reference position, sequence qualities, base qualities, and base calls. The matrices are used to query the probability of the variable of interest conditional on all other variables in the matrix. For example, in a matrix containing

Table 2 Used datasets

Identifier	Sequencer	Adapter	Species	Reference	Accession ID **Sample ID**
Ec-Hi2000-TruSeq	HiSeq 2000	TruSeq	*Escherichia coli*	ASM584v2	SRR490124
Ec-Hi2500-TruSeq	HiSeq 2500	TruSeq	*Escherichia coli*	ASM584v2	SRR3191692
Ec-Hi4000-Nextera	HiSeq 4000	Nextera	*Escherichia coli*	ASM584v2	Ecoli1_L001
Bc-Hi4000-Nextera	HiSeq 4000	Nextera	*Bacillus cereus*	ASM782v1	Bcereus1_L001
Rs-Hi4000-Nextera	HiSeq 4000	Nextera	*Rhodobacter sphaeroides*	ASM1290v2	Rsphaeroides1_L001
At-HiX-TruSeq	HiSeq X Ten	TruSeq (unknown barcode)	*Arabidopsis thaliana*	TAIR9.171	ERR2017816
Mm-HiX-Unknown	HiSeq X Ten	Unknown	*Mus musculus*	GRCm38.p6	ERR3085830
Hs-HiX-TruSeq	HiSeq X Ten	TruSeq	*Homo sapiens*	GRCh38.p13	ERR1955542
Hs-Nova-TruSeq	NovaSeq 6000	TruSeq	*Homo sapiens*	GRCh38.p13	PRJEB33197
Ec-Mi-TruSeq	MiSeq	TruSeq	*Escherichia coli*	ASM584v2	DRR058060
At-BGI	BGISEQ-500	BGISEQ	*Arabidopsis thaliana*	TAIR9.171	PRJNA562949

Adapters labeled as unknown are not listed in the Illumina and BGI adapter overview [33, 34]

the base quality BQ, previous quality PQ, and sequence position SP, we would query $p(BQ|PQ, SP)$ for all BQ, which is a normalized slice of the matrix. These probabilities would then be used to draw the base quality.

However, due to the amount of variables included in each of these statistics, we cannot directly use large (sparse) matrices. For example, storing the quality values of Ec-Hi2000-TruSeq would require a matrix with $4.6 \cdot 10^9$ entries. Therefore, ReSeq only retains two-dimensional margins of the matrices. For the base-quality values, this means storing 10 two-dimensional margins for each template segment, tile, and reference base combination: BQ - sequence quality (SQ), BQ -PQ, BQ - SP, BQ - error rate(ER), SQ - PQ, SQ - SP, and so forth. This removes the higher-dimensional (3+) effects from the distributions, yet still provides a reasonable approximation (Additional file 1: Figure S1). The new set of marginal matrices has only around $3.0 \cdot 10^5$ combined entries. This saves considerable computer memory, but more importantly prevents sparsity, because sufficient observations are required to sample accurate probability distributions in the absence of an analytical description. Using the full matrix, the conditional probability of a base quality $p(BQ|SQ, PQ, SP, ER)$ would require many observations for every variable combination. In the reduced representation, the sampling only requires many observations for every two-dimensional combination of variables (i.e. SQ - PQ, SQ - SP, PQ - SP, etc.). Thus, the method requires much smaller input datasets.

Comparison of quality values and error rates

Here, we check whether the quality values and error rates show the typical patterns over the read length on all eleven datasets (Table 2). Additionally, we test Ec-Hi2500-TruSeq-asm, where the simulation profiles are trained on a non-optimized assembly built from

Ec-Hi2500-TruSeq that is highly fragmented and has many duplications (QUAST [35] report in Additional file 2: 61992 contigs, N50 of 402 and covering 98.9% of the *E. coli* reference with every covered reference base being represented, on average, 2.553 times in the assembly). Thus, comparing the results of Ec-Hi2500-TruSeq-asm and Ec-Hi2500-TruSeq gives insight on how well simulators can build profiles from a fragmented assembly (e.g., by filtering). In contrast, using the assembly as simulation template would not allow similar insight, because a template can only be improved by changing it, which affects all simulators equally. Therefore, we continue using the *E. coli* reference as simulation template.

In general, the simulated mean quality values by position resemble those of real data (Additional file 1: Figure S2, S3), except for BEAR, which produces reads of varying length and has for most datasets and positions, average quality scores 5 to 10 Phred units lower than the real data. A likely explanation for the strong deviation is BEAR's design choice to replace quality values at error positions with qualities generated by the error model instead of having error rates depend on the quality values. For ART, we observe quality values consistently one Phred unit higher than the real data (Additional file 1: Figure S3b–e, g–i, k–l). For pIRS and ReSeq, deviations appear with increased position (Additional file 1: Figure S3d, g–i), which is likely caused by training the quality values only on mapped reads. For Hs-Nova-TruSeq, all simulators have a more pronounced decrease in quality compared to real data, which is a result of the preqc filtering (Additional file 1: Figure S3j). Moreover, in Rs-Hi4000-Nextera, ReSeq experiences a strong drop in the second read's quality (Additional file 1: Figure S4d) that is absent from the first read (Additional file 1: Figure S4c). It is also worth mentioning that pIRS uses the same quality distribution for first and second reads, despite the clear differences in real data (Additional file 1: Figure S4). Finally, the mean quality values for Ec-Hi2500-TruSeq-asm and Ec-Hi2500-TruSeq are the same, as expected, since the qualities are independent of the reference.

The mean error rates by position are also well reproduced in the simulations (Additional file 1: Figure S5, S6), except for BEAR, since it uses DRISEE for training where increased error rates have already been observed [36]. The applied exponential regression seems to amplify the increased error rates for higher positions (Additional file 1: Figure S5d–e, g–h, l). At-HiX-TruSeq, Mm-HiX-Unknown, Hs-HiX-TruSeq, and At-BGI (Additional file 1: Figure S5g-i,l) are outside of BEAR's defined metagenomic use-case, but this does not generally affect performance. On another note, Ec-Hi2000-TruSeq, Ec-Hi2500-TruSeq, and Ec-Mi-TruSeq highlight the weakness of the hard-coded error models used by ART and NEAT (Additional file 1: Figure S6a-c,k). Such models require well-calibrated quality values (ART) or identical calibrations to their training set (NEAT). Especially in Ec-Hi2000-TruSeq, the quality values are not well-calibrated: while the Phred quality score of 2 predicts a 63% error rate, the observed rate after mapping is only 15%. Therefore, ART has strongly inflated error rates in this dataset, which could be solved by recalibrating the quality values, but then the quality values simulated from ART would be as inaccurate as the error rate is. Finally, ART's and NEAT's error rates increase sharply for Ec-Hi2500-TruSeq-asm compared to Ec-Hi2500-TruSeq, which seems to be an artifact from the reference-free error estimation, because increasing the coverage removes the error-rate differences completely for ART and mostly for NEAT (Additional file 1: Figure S7).

Systematic errors

The most important new feature introduced by ReSeq is the representation of systematic errors. In older HiSeq2000 data, the systematic behavior of errors is very pronounced (Fig. 2a,b), but also on the newer HiSeq4000, systematic errors are still present (Fig. 2c, d). Errors appear bundled at some positions and are strand dependent, which rules out variants as the cause. Until now, no model exists that predicts sequence-specific errors for a given position in a read based on all nucleotides in the same read preceding this position. Therefore, ReSeq distributes systematic errors randomly over the template by drawing error tendencies and rates for each template base on each strand. The five possible values for the error tendency are the four nucleotides and no tendency (i.e., random error). Error tendencies and rates can be stored and loaded to conserve the errors between multiple simulation runs, because real systematic errors are also very conserved, as highlighted in Fig. 2e by comparing Ec-Hi4000-Nextera (Ecoli1_L001) with technical replicates from the same library run on different lanes (Datasets Ecoli1_L002 and Ecoli1_L003) and from separately prepared libraries sequenced on the same lane (Datasets Ecoli2_L001 and Ecoli3_L001).

Figure S8 (Additional file 1) shows that ReSeq manages to reproduce the distribution of systematic errors well, except for *Arabidopsis thaliana* (Additional file 1: Figure S8g, l), with its extreme coverage difference between the chromosomes and chloroplast.

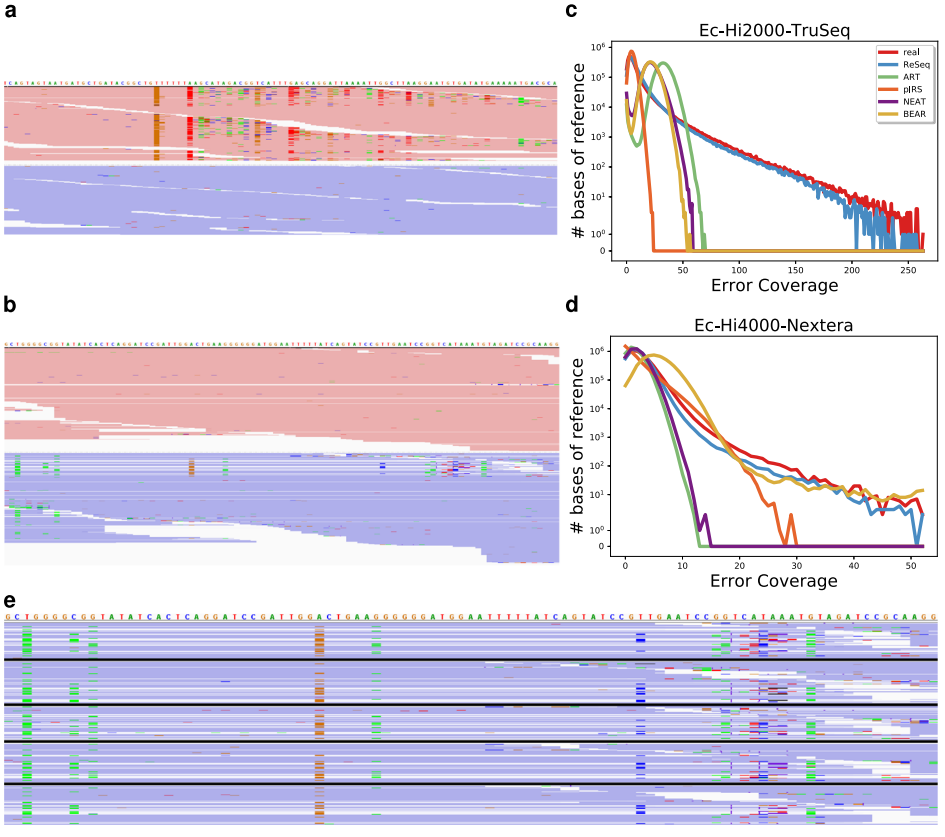

Fig. 2 Systematic errors in Illumina data. **a**, **c** Screenshots from the Integrative Genomics Viewer [37, 38] for Ec-Hi2000-TruSeq (**a**) and Ec-Hi4000-Nextera (**c**). The forward strand is colored red and the reverse strand blue. The bright colors mark substitution errors. **b**, **d** Amount of errors for all positions in the reference. **e** Section from c with the same section from four technical replicates of Ec-Hi4000-Nextera (Ecoli1_L001): Ecoli1_L002, Ecoli1_L003, Ecoli2_L001, Ecoli3_L001. The datasets are separated by thick black lines

Heterozygous variants in diploid samples (Additional file 1: Figure S8h–j) and fragmented references (Additional file 1: Figure S8c) also lead to reduced similarity to real data.

Coverage model

The coverage model defines the probabilities of where simulated reads start and end in the simulated genome (template). It works on fragment sites, which are defined by their reference sequence, start position, fragment length, and strand. Sites have been shown to be a good choice to estimate GC bias [24]. Since in our model duplications are fragments falling on the same site, the strand was added to make duplications strand-specific. During the statistics calculation, ReSeq determines all possible sites in a genome and counts the read pairs mapping to them. Sites, where the true counts are likely to deviate from the observed counts, e.g., repeat regions, are removed by looking at clusters of low mapping quality (see the "Methods" section). During the simulation, repeats do not require special consideration as long as they are part of the simulation template.

The model takes four different sources of bias into account: GC and flanking sequence in a first step and fragment length and reference sequence in a second step. The flanking bias arises from the nucleotides *flanking* the start and end of the fragment as a result of the fragmentation process. We observed that their effect on the coverage is especially pronounced if enzyme digestion is used (Additional file 1: Figure S9). The GC bias is due to PCR [23], while the fragment-length bias results from the fragmentation and size selection. Lastly, the reference-sequence bias represents the original abundance of the sequence in the sample.

In the first step of the coverage estimation, the GC and flanking bias are fit to the sites and their counts. Modeling the counts at each site with a Poisson would only account for statistical duplications. Therefore, we use a negative binomial to additionally account for PCR and optical duplications. Since coverage biases arise from different processes, we assume independence and write the mean μ_n of the negative binomial as a product:

$$\mu_n = \tilde{N} b_{seq}(seq_n) b_{len}(len_n) b_{GC}(GC_n) b_{start,n} b_{end,n}$$

with $\tilde{N}$ as the genome-wide normalization and the different b as the biases for this site.

The normalization parameter is of no further interest after the fit, leaving 128 relevant parameters. The GC bias b_{GC} (Fig. 3a) is binned by percent into 101 bins that are described by a natural cubic spline with six knots, i.e. six degrees of freedom.

The flanking bias (Fig. 3b) has 120 parameters: one for each nucleotide across each of thirty positions (10 bases before the fragment to 20 bases within). The flanking bias at the start and end of the fragment use the same parameters due to their similarity (Additional file 1: Figure S9), with the end reverse complemented to keep the meaning of positions relative to the fragment. Since the best way of combining biases from individual positions in the flank is a priori unknown, we tested the two simple options of a product and a sum on datasets fragmented using mechanical forces (Ec-Hi2000-TruSeq) and enzymes (Ec-Hi4000-Nextera). As seen in Fig. 3c, the predictions from the summation model nicely fit to the observed count means, using $2 \cdot 10^5$ sites per bin.

To properly account for duplications, the dispersion has two parameters, α and β, leading to the following mean-variance relationship:

$$\sigma_n^2 = \mu_n + \alpha \mu_n + \beta \mu_n^2.$$

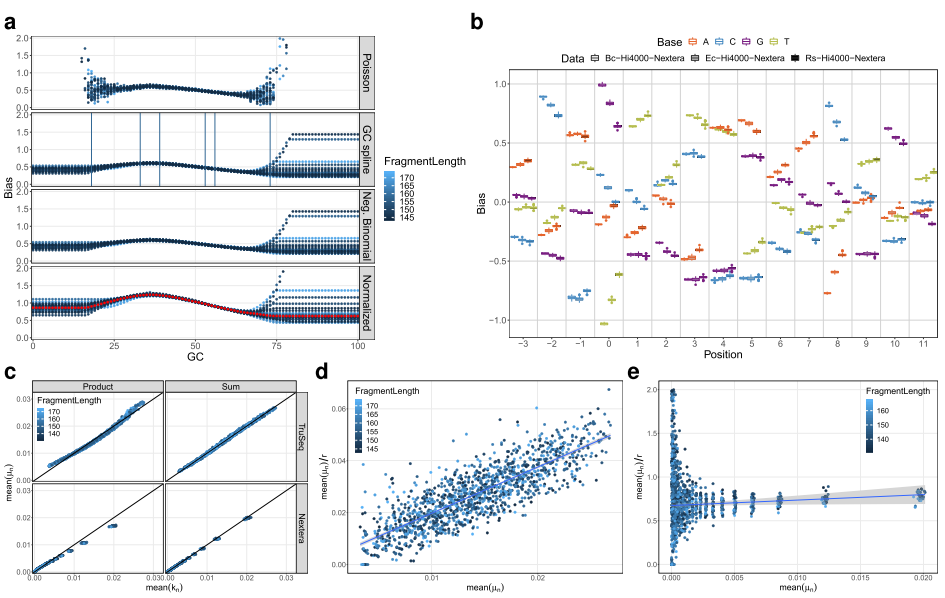

Fig. 3 Coverage model. **a** GC bias for Ec-Hi2000-TruSeq at the four steps of the bias fit for 30 fits with different fragment lengths. The red dots in the normalized panel are the median value and represent the final result. The horizontal lines in the GC spline panel are the chosen knots for one example fragment length. **b** Flanking bias. The effect of nucleotides in the genome relative to the fragment start or end with position 0 being the start/end. Negative positions are outside of the fragment. Only position − 3 to 11 are shown from the total model that includes position − 10 to 19. Each box summarizes 30 to 40 fits with different fragment lengths. The boxes are arranged around their true position for improved readability. The three datasets are all created with Nextera adapters. **c** Comparison of combining the different positions in the flanking bias by a product or a sum. Each dot is one bin of $2 \cdot 10^5$ fragment sites for one of 30 fits with different fragment length. The fragment sites are ordered by their predicted mean counts μ_n before binning. The *x*-axis is the mean of observed counts in the bin. The *y*-axis is the mean of predicted mean counts. For the sum the dots scatter around the identity, while for the product a curve is visible. **d**, **e** The observed counts k_n for the bins defined in c are fitted with a negative binomial with constant dispersion *r* for Ec-Hi2000-TruSeq (**d**) and Ec-Hi4000-Nextera (**e**). While Ec-Hi2000-TruSeq shows a significant slope and nearly no y-intercept, Ec-Hi4000-Nextera shows the exact opposite

Figure 3d, e, where α is the y-intercept and β is the slope, demonstrates that both parameters are needed to properly simulate all datasets and duplication types, even if single datasets need only one of the parameters (Additional file 1: Figure S10a) [39].

After the other biases have been fitted, ReSeq estimates the reference-sequence and fragment-length biases by iteratively adjusting them to match the observed coverage.

To test the biases obtained by ReSeq, we use Bc-Hi4000-Nextera, Ec-Hi4000-Nextera, and Rs-Hi4000-Nextera, which should have similar biases; and indeed, the flanking biases do not vary much in those three datasets (Fig. 3b), despite the different median GC content of the underlying genomes (35%, 51%, and 69%). Furthermore, we clearly reproduce previous findings for Nextera adapters [25], where the biases between a nucleotide and its complement are very similar if mirrored around position 4 (e.g., A at 5 and T at 3). The GC biases for the three datasets are compared in Figure S11 (Additional file 1), where the spread between different fragment lengths highlights the lower confidence in biases based on fewer sites. Bc-Hi4000-Nextera and Ec-Hi4000-Nextera look very similar, except for low-confidence, GC-rich fragments. Rs-Hi4000-Nextera looks somewhat different, but its high-confidence region is poorly accessible by the other two datasets. The reduced occurrence of AT-rich fragments (AT dropout) described in the literature [25] is

also observed with the exception of Rs-Hi4000-Nextera, which has little power to detect AT dropout due to its high median GC content. As another test, the biases were estimated on simulated, uniformly distributed data, which results in the expected flat profile (Additional file 1: Figure S12). The minor deviations for GC biases at the edges are due to low amounts of available sites. Overall, the above tests show that this part of the coverage model delivers accurate and consistent results.

Comparison of coverage distributions

Figure 4 shows ReSeq's improvements over the uniform distribution (ART and BEAR) or the sliding window approach (pIRS and NEAT), in terms of base-coverage distributions. No other simulator shows consistently a good accordance with real data. In Ec-Hi2500-TruSeq-asm (Fig. 4c), the median coverage provided as coverage parameter to ART, pIRS, and NEAT is not a good estimator for the real coverage, because the duplicated regions and the many very short hard-to-map contigs in the assembly reduce the median coverage from 1011 to 13. Therefore, ART and pIRS do not have an expected peak, whereas NEAT's

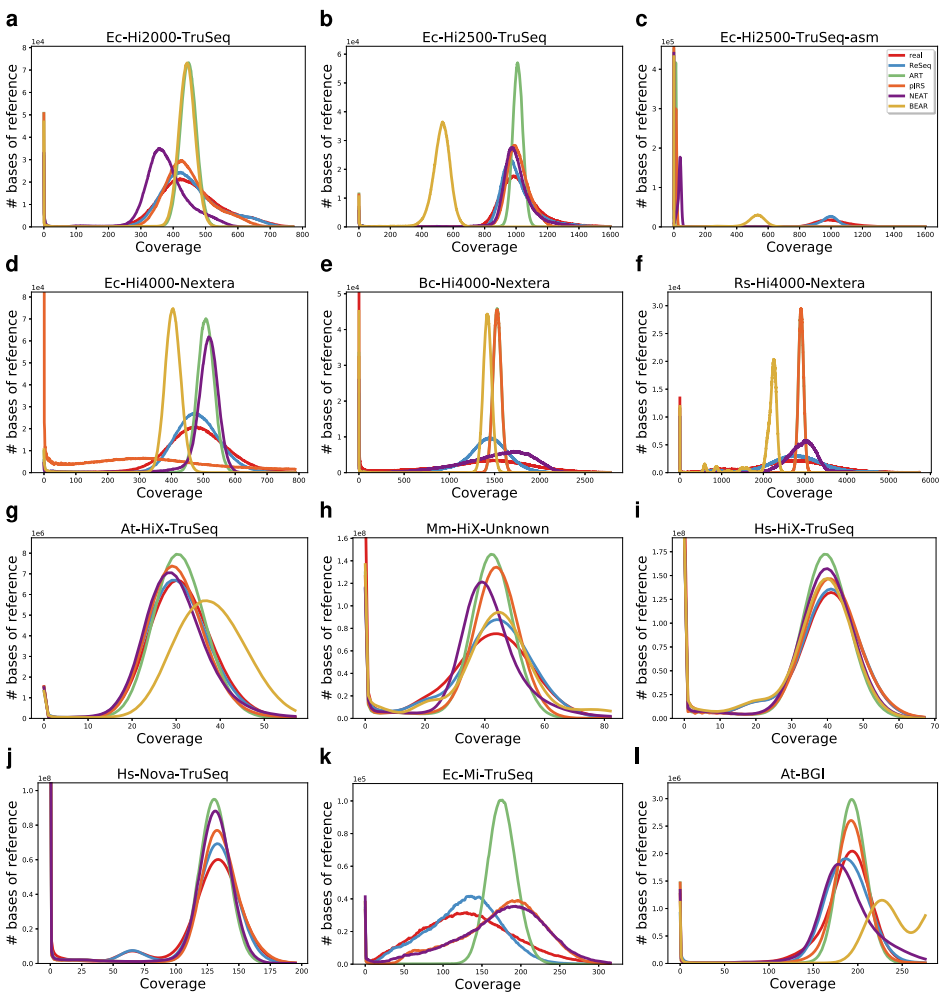

Fig. 4 Base-coverage distribution. Note that **j** BEAR was omitted due to excessive runtimes. **k** BEAR was omitted, because it did not have a single proper pair mapped

peak is at a very low coverage. ReSeq extracts the coverage itself and is therefore not directly affected. BEAR requires the number of reads and is therefore also not affected, but still does not show a peak at the correct coverage. In Figure S13 (Additional file 1), the median coverage estimated from the mapping to the reference (Ec-Hi2500-TruSeq) is provided to all simulators except BEAR. Since this is the only coverage parameter for ART, it performs the same as in Ec-Hi2500-TruSeq. In contrast, pIRS and NEAT do not manage to simulate a coverage that remotely resembles real data, because their GC-bias estimation is not robust to fragmented references.

ReSeq's negative binomial also captures the number of duplicated read pairs well for most datasets (Fig. 5), despite a decrease in performance with increasing size and complexity of the genome. A particular case are human samples, where the simulated duplication numbers exceed the highest real ones, but only around 0.1% of the fragments are affected for the most severe case in Hs-HiX-TruSeq (Fig. 5i). Other simulators do not handle duplications and thus do not resemble real data (Fig. 5d–j, l), except for datasets with low levels of duplication.

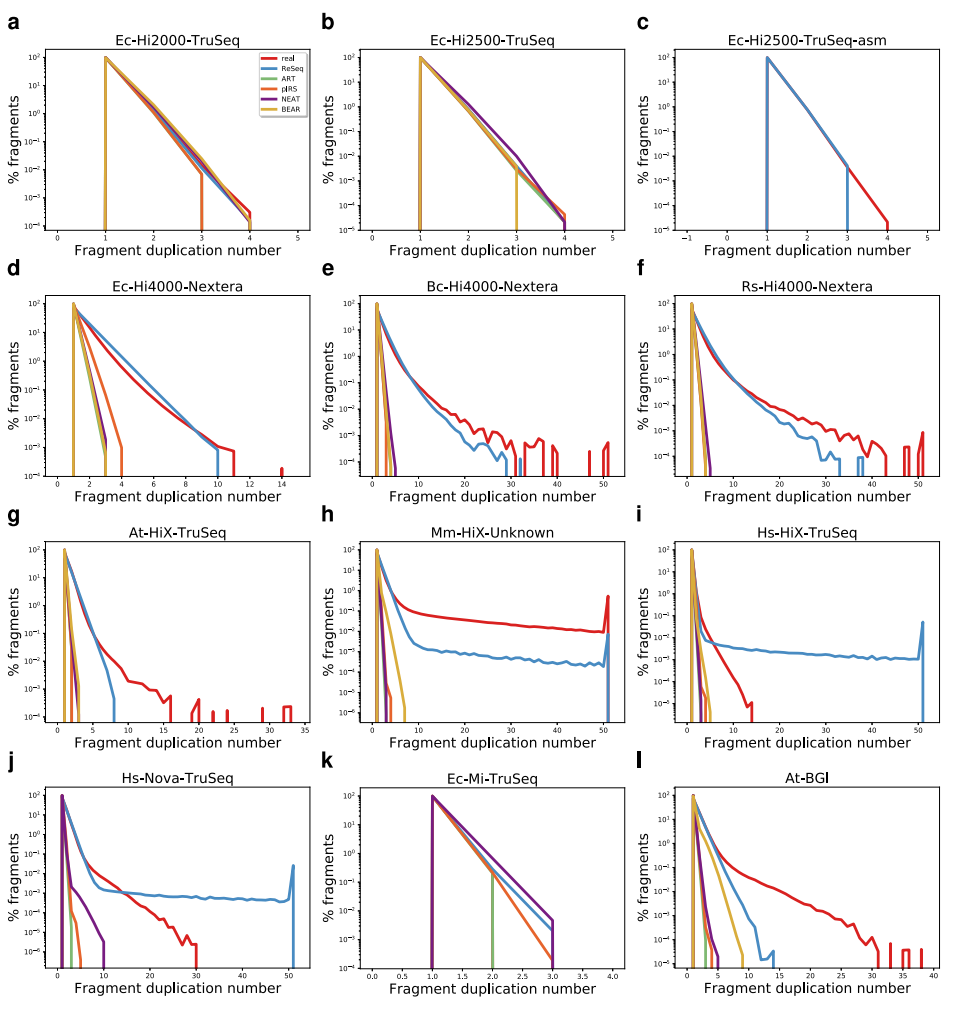

Fig. 5 Fragment duplication. The spike at 51 is an artifact of the counting that treats everything above 50 as 51. Note that **c** ART, pIRS, NEAT, and BEAR are omitted due to low coverage. **j** BEAR was omitted due to excessive runtimes. **k** BEAR was omitted, because it did not have a single proper pair mapped

Furthermore, ReSeq replicates the fragment-length distribution accurately (Additional file 1: Figure S14) for two reasons: (i) it does not parameterize the distribution with a Gaussian, which prevents ART, pIRS, and BEAR from capturing the long tail (Additional file 1: Figure S14a–c, h–j, l); (ii) it is the only simulator that implements adapters, which allows for fragment lengths below the read length (Additional file 1: Figure S14d–f, k). BEAR also allows for shorter fragment lengths, but does so by trimming the read length and thereby prevents benchmarks that require adapters to be present.

As a further test, we can again use the technical replicates of Ec-Hi4000-Nextera (Ecoli1_L001). The correlation of the base coverage between Ecoli1_L001, Ecoli1_L002, Ecoli1_L003, Ecoli2_L001, and Ecoli3_L001 can be seen as a gold standard of how much the base coverage of simulations should be correlated to real data. We calculated the three pairwise Spearman correlations of replicates from the same lane and from the same library, respectively, and compared them to the correlations of three independent simulations based on Ec-Hi4000-Nextera. For the simulations, we calculated the pairwise Spearman correlations with themselves and the three correlations with Ec-Hi4000-Nextera. Table 3 lists the averages of the calculated coverage correlations. For each entry, one of the three corresponding correlations is plotted in Figure S15 (Additional file 1). The correlations highlight three major points. First, ReSeq strongly increases the correlation between simulation and real data because, despite its high variance, the whole coverage distribution follows the real trend, which is not visible for other simulators. Second, independent simulations from ReSeq approach the correlatedness of real data, but do not populate the low- and high-coverage regions present in real data. Only pIRS performs similarly, but with a correlation of 1, nearly all randomness seems to be removed. Third, despite the major improvements, the correlation between ReSeq and real data is still low and further improvements are possible.

Comparison of k-mer spectra

A good high-level summary statistic to represent a dataset of genomic reads is the k-mer spectrum, which shows the systematic properties of errors (exponential decrease at low frequencies) and the coverage distribution (shape and position of peaks) (see Sohn and Nam Figure 4 [40]). Figure 6 (linear scale) and Figure S16 (log scale, Additional file 1) display the 51-mer spectra of the datasets and their simulations. The 31-mer and 71-mer spectra (Additional file 1: Figure S17, S18) are qualitatively the same and the conclusions drawn in this section are not specific for $k = 51$. The first row (Fig. 6a–c) is based on high-coverage *E. coli* datasets sequenced on the older HiSeq 2000 and 2500 with TruSeq

Table 3 Average (pairwise) Spearman correlations of base coverage

	Lane	Library	Real	Simulation
Real	0.91	0.92	-	-
ReSeq	-	-	**0.23**	**0.81**
ART	-	-	0.01	0.00
pIRS	-	-	0.12	**1.00**
NEAT	-	-	0.10	0.26
BEAR	-	-	− 0.00	0.05

The bold numbers highlight the simulators, which are the closest to the correlatedness of real data in the given category. Lane: Ec-Hi4000-Nextera (Ecoli1_L001) and replicates Ecoli1_L002 and Ecoli1_L003. Library: Ec-Hi4000-Nextera (Ecoli1_L001) and replicates Ecoli2_L001 and Ecoli3_L001. Real: Ec-Hi4000-Nextera and three simulations. Simulation: Three simulations of Ec-Hi4000-Nextera

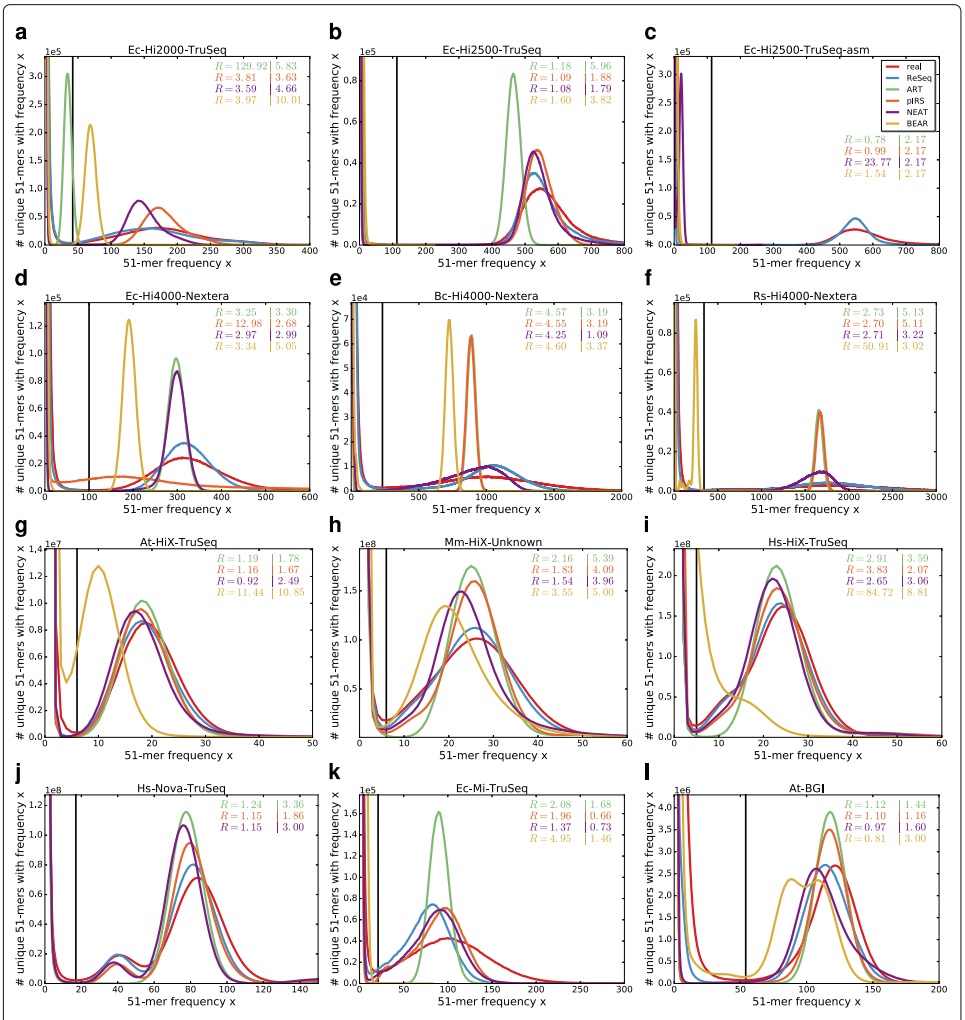

Fig. 6 51-mer spectra of real and simulated data. The shape and position of peaks reflects the coverage distribution, while the exponential decrease at low frequencies is defined by systematic errors. The black lines show the minimum between the exponential decrease and signal peak for real data. The first value of *R* gives the sum of relative deviations for frequencies (x) up to the minimum. The second value of *R* gives the sum of absolute deviations for frequencies larger than the minimum. Both values are stated relative to ReSeq. When the signal peak of the simulation does not exist or starts before the minimum in real data the values lose their interpretability. This happens for ART (**a**, **c**), pIRS (**c**, **d**, **l**), NEAT (**a**, **c**-**e**, **l**), and BEAR (**b**, **c**, **f**–**g**, **i**, **k**–**l**)

adapters and includes the simulations trained on the fragmented assembly instead of the reference (Fig. 6c). The second row's bacteria datasets (Fig. 6d-f) are all produced in a single HiSeq 4000 run using Nextera adapters (enzyme fragmentation). They are ordered from left to right by coverage ranging from high (508x) to very high (2901x) and by genome complexity: single sequence (*E. coli*), two sequences where one is not present in the data (*B. cereus*) and multiple sequences including plasmids of varying abundance (*R. sphaeroides*). The plasmids cause multiple smaller peaks in the 51-mer spectrum with lower frequencies than the main peak (Additional file 1: Figure S16f). Additionally, especially in Bc-Hi4000-Nextera (Additional file 1: Figure S16e), we observe variants in the bacteria populations that increase the 51-mer counts between the systematic errors and the signal peak. Although NEAT and ReSeq accept Vcf files, which allows non-diploid variants to be specified, we do not include them in the simulation. In the third row,

we have low-coverage (31x-43x), diploid datasets sequenced on HiSeq X Ten machines with TruSeq or related adapters. Here, we do include the variants in the simulations, which results in a second peak in the 51-mer spectrum at half the frequency of the main peak (Fig. 6h,i). Furthermore, the three genomes contain sequences with differences in abundance: in At-HiX-TruSeq (Additional file 1: Figure S16g), we observe a second peak at higher frequencies stemming from the mitochondria; in Mm-HiX-Unknown (Fig. 6h), complex changes in the NIH3T3 cell line [41], including copy-number variations, broaden the signal peak; and, in Hs-HiX-TruSeq (Fig. 6i), the X and Y chromosomes strengthen the heterozygous peak at half the coverage of the main peak. The fourth row is a mix of sequencers (NovaSeq, MiSeq, BGISEQ) run on different species (*H. sapiens*, *E. coli*, *A. thalina*) with medium coverage(131x-194x). The diploid species were simulated with variants.

BEAR does not produce a signal peak that matches the real signal's shape or position in any of the tested datasets. ART is consistently underperforming, for the coverage distribution as well as for the systematic errors. Its error rate inflation for Ec-Hi2000-TruSeq is causing the strong peak shift in the 51-mer spectrum (Fig. 6a). Notably, pIRS struggles with datasets with Nextera adapters: for Ec-Hi4000-Nextera, it results in a rather flat peak and an overabundance of high-frequency 51-mers (Fig. 6d), while it maintains a uniform coverage distribution for the other Nextera datasets (Fig. 6e-f).

ReSeq compares favorably for the coverage distributions of all datasets, except Ec-Mi-TruSeq (Fig. 6a–j, l), and for the systematic errors of all datasets, except At-HiX-TruSeq and At-BGI (Fig. 6a–f, h–k, Additional file 1: Figure S16a–f, h–k). Notably, ReSeq reproduces the coverage peak better on data with TruSeq adapters (Fig. 6a, b, g–j), compared to Nextera adapters (Fig. 6d–f). Furthermore, for all three HiSeq X Ten samples and the BGISEQ sample (Fig. 6g–i, l), ReSeq slightly underestimates the coverage. This could be manually corrected by specifying a parameter. For Ex-Mi-TruSeq, it is more complicated, because the coverage distribution fits (Fig. 4k), but the k-mer peak does not (Fig. 6k). A likely reason are large differences between the reference and the genome underlying the data (for instance, a low mapping rate of 65%). Using the higher medium coverage instead of ReSeq's own estimation improves the 51-mer spectrum, but reduces the similarity to real data for the base-coverage distribution (Additional file 1: Figure S19). Finally, the exponential decrease is poorly reproduced in the low and medium coverage datasets (Additional file 1: Figure S16g–l). We already saw that the systematic errors are harder to replicate for diploid samples (Additional file 1: Figure S8g–j, l); additionally, the extremes of the coverage peak, which are missing in simulations (Additional file 1: Figure S15c), overlap more with the exponential decrease than in high coverage datasets.

Since for Ec-Hi2500-TruSeq-asm (Fig. 6c), the median coverage provided (as coverage parameter to ART, pIRS, and NEAT) is a poor estimator, the median coverage estimated from the mapping to the reference is provided to all simulators except BEAR for Figure S20 (Additional file 1). For ART, the improved coverage removes nearly all performance losses compared to Ec-Hi2500-TruSeq, which is not surprising, since only InDel rates and the fragment-length parameters are taken from the assembly. Even though its performance is still low, ART improves relative to pIRS and NEAT, because these two do not cope well with the fragmented assembly and produce rather flat and broad peaks. ReSeq's coverage model also suffers from using the assembly instead of the reference, but can still maintain a general resemblance to the real data.

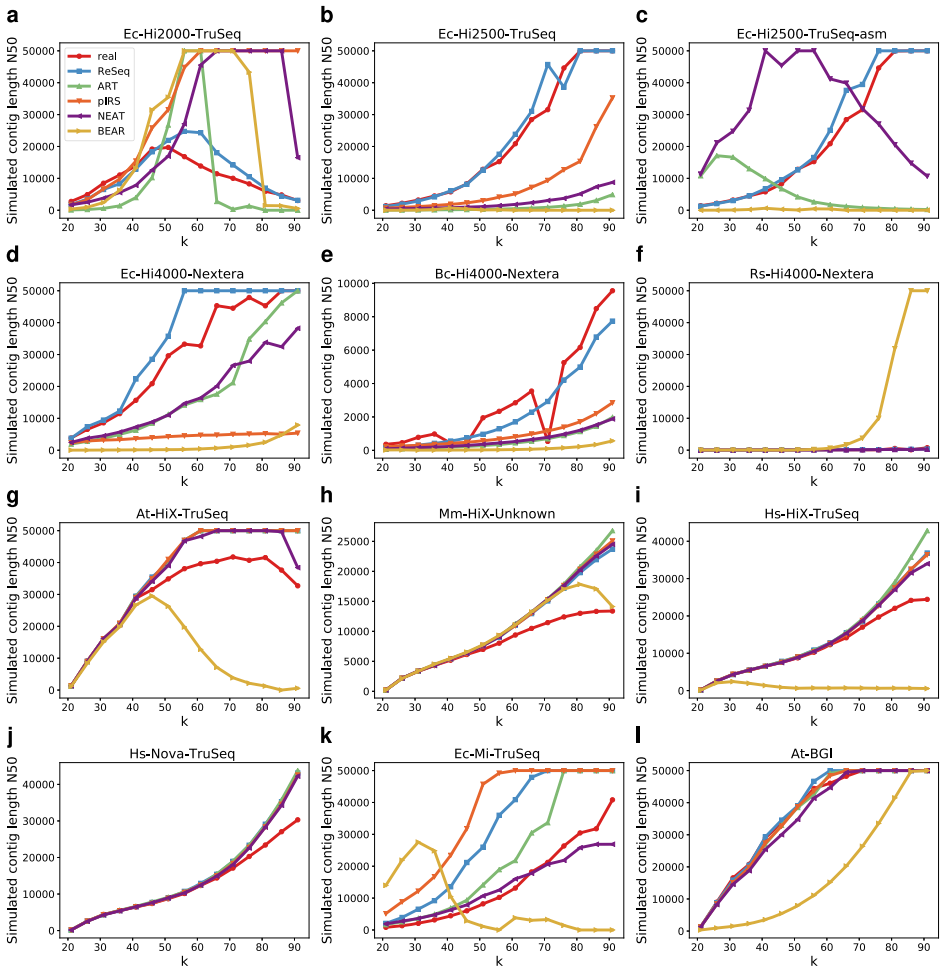

Fig. 7 Simulated contig N50 for real and simulated data. **c** preqc crashes for pIRS. **f** preqc crashes for pIRS and ART

Comparison of assembly continuity

To check whether the improved representation in the k-mer spectrum has consequences for applications, we ran the sga preqc module on all datasets and their simulations (Fig. 7). The preqc module estimates the N50 (length of the shortest contig still necessary to cover 50% of the genome) of short-read assemblies for different k-mer lengths. Due to crashes, Ec-Hi2500-TruSeq-asm is missing pIRS and Rs-Hi4000-Nextera is missing pIRS and ART.

For many datasets, ReSeq follows real data better than the other simulators (Fig. 7a–e). In the others, namely the low- and medium-coverage HiSeq X Ten and NovaSeq datasets (Fig. 7g–j), all simulators except BEAR perform equally, because the missed systematic nature of errors and undetected diploid variants prevent ReSeq from delivering the same performance. Only in Ec-Mi-TruSeq does ReSeq perform worse than NEAT and ART, due to the discrepancy between mapped coverage and k-mer coverage. Setting the coverage parameter to the median coverage as for the other simulators mitigates this effect (Additional file 1: Figure S21). For Rs-Hi4000-Nextera, BEAR deviates drastically from real data and other simulators (Fig. 7f). Figure S22 (Additional file 1) shows the same plot

without BEAR, but due to fluctuations in the real N50 values, no strong conclusions can be drawn.

In Ec-Hi2500-TruSeq-asm (Fig. 7c), the coverage-parameter estimate from the assembly strongly affects the N50 values of ART, pIRS, and NEAT. Providing the coverage estimate from the reference to all simulators except BEAR (Additional file 1: Figure S23), mostly reverts the drastic changes in N50. pIRS's difference to real data only slightly increases, compared to using the reference for the complete training (Ec-Hi2500-TruSeq: Fig. 7b), while NEAT does not show the unexpected peak of N50 around $k = 50$ anymore, but does not restore the slight resemblance to the real data it had in Ec-Hi2500-TruSeq, and ART fully recovers the increase in N50 toward high k that has at least the same tendency as real data. Noticeably, ReSeq's N50 values stay close to the real ones no matter how it was trained.

Comparison of cross-species simulations

An important property of trained profiles is that not only can they be used to simulate the original dataset, but also datasets with alternative templates. To test training and simulation across species, we split six of the datasets into two groups with identical sequencer and fragmentation combinations. The datasets were simulated using the median coverage, template, and variants from the simulated datasets; all other parameters and profiles were provided from another dataset within the group. The resulting 51-mer spectra are compared to those of the simulated datasets in Figure S24 (linear scale, Additional file 1) and Figure S25 (log scale, Additional file 1). In the HiSeq 4000 group with enzyme fragmentation, we used the profiles from Ec-Hi4000-Nextera to simulate Bc-Hi4000-Nextera and Rs-Hi4000-Nextera and the profiles from Rs-Hi4000-Nextera to simulate Ec-Hi4000-Nextera. The results highlight the difficulty of applying profiles across genomes with very different GC content (Additional file 1: Figure S24a-c). In the HiSeq X Ten group with mechanical fragmentation and genomes with similar GC content (Additional file 1: Figure S24d-f), we used the profiles from Mm-HiX-Unknown to simulate At-HiX-TruSeq and Hs-HiX-TruSeq and the profiles from Hs-HiX-TruSeq to simulate Mm-HiX-Unknown.

Similar to the case of simulating the original datasets (Fig. 6), BEAR does not create a signal peak that matches the real signal's shape or position in any of the datasets, except At-HiX-TruSeq (Additional file 1: Figure S24). ART is unaffected by the profile change, but still shows low performance due to the uniformly distributed coverage. pIRS's performance on cross-species simulations is hard to judge on the Nextera datasets, because pIRS already underperforms when reproducing the original Nextera datasets. For example, pIRS on Ec-Hi4000-Nextera results in a rather flat peak (Fig. 6d) and simulating other datasets from its profile results in no visible peak (Additional file 1: Figure S24b,c). However, for TruSeq adapters and genomes with similar median GC content, pIRS seems to be unaffected by using profiles trained on a different species (Fig. 6g–i, Additional file 1: Figure S24d-f)). This is not the case for NEAT, where in the HiSeq 4000 group, minor peaks appear on the low-frequency side of the main peak (Additional file 1: Figure S24c, S25a,c), and in the HiSeq X Ten group, an asymmetric signal peak with a pronounced tail is visible in the simulations based on the Mm-HiX-Unknown profiles (Additional file 1: Figure S24d,f). This is likely caused by other sources of bias being incorporated into the GC bias. ReSeq is missing the abundance information for the

reference sequences and therefore does not correctly simulate the plasmids in Rs-Hi4000-Nextera (Additional file 1: Figure S24c), the mitochondrial genome in At-HiX-TruSeq (Additional file 1: Figure S25d), the copy-number variations in Mm-HiX-Unknown (Additional file 1: Figure S24e), and the X and Y chromosome in Hs-HiX-TruSeq (Additional file 1: Figure S24f). This demonstrates how ReSeq nicely separates the individual biases and produces results (Additional file 1: Figure S24) resembling the original datasets (Fig. 6). On Mm-HiX-Unknown, we demonstrate that given the abundance information, ReSeq again produces a coverage peak as good as the one from the simulation trained on the original dataset (Additional file 1: Figure S26, Fig. 6h).

The assembly continuity seems to be unaffected by not incorporating abundance information and most simulations perform as well or even slightly better (Additional file 1: Figure S27b, d–f) than with the profile from the original datasets (Fig. 7e, g–i). The exception are simulating Ec-Hi4000-Nextera with the profile from Rs-Hi4000-Nextera and vice versa. For Ec-Hi4000-Nextera, ReSeq, ART, and NEAT show a decrease in similarity to the real data and only pIRS and BEAR improved, but they did not have any resemblance to real data for the original profile (Additional file 1: Figure S27a, Fig. 7d). For Rs-Hi4000-Nextera, the general trend is unclear, because only ReSeq and BEAR manage to create artificial datasets that can be evaluated with the preqc module in both cases (Additional file 1: Figure S27c, S22, Fig. 7f). ReSeq's performance drops strongly in relative terms, but the absolute difference in N50 remains low. On the other hand, BEAR changes from extremely high N50 values for the original profile to a flat N50 distribution for the Ec-Hi4000-Nextera profile.

Looking at more detailed statistics, the quality values and error rates remain mostly accurate (except for BEAR), but reflect the differences between datasets (Additional file 1: Figure S28, S29, S30, S31). The systematic errors are reproduced by ReSeq also for cross-species simulations, but the performance for the HiSeq 4000 datasets with strongly varying median GC content (Additional file 1: Figure S32a-c) is reduced compared to the simulation of the original datasets (Additional file 1: Figure S8d–f). However, for the HiSeq X Ten datasets with similar median GC content, the performance does not change (Additional file 1: Figure S32d–f, S8g–i). Finally, the base-coverage distribution (Additional file 1: Figure S33) highlights again the effect of the missing abundance information for ReSeq and NEAT's issue applying the GC bias across species.

Comparison of computation requirements

Besides better representation of real data, computational requirements are also important. Figure S34 (Additional file 1) shows the total CPU time, the elapsed time and the maximum amount of memory used for the training and simulation steps of each simulator for the eleven datasets. ReSeq lies on the high side of CPU time for training and in the intermediate region for simulation (Additional file 1: Figure S34a, d). Noticeably, parts of ReSeq's CPU requirements scale with genome size instead of number of reads, which leads to worse performance in lowly-covered genomes.

Due to ReSeq's effective parallelization, its elapsed times are low for this benchmark with 48 virtual CPUs (Additional file 1: Figure S34b,e). In contrast, the single-threaded processes implemented in perl or python have strikingly high elapsed times. This is well visible in Hs-HiX-TruSeq and applies to the training of pIRS (over a week), NEAT (several

days), and BEAR (half a week) as well as the simulation of NEAT (close to 2 weeks) and BEAR (several weeks). For the simulation, NEAT provides a tool to split the template and later merge the reads, which can be used for manual parallelization with multiple instances to reduce the elapsed time to 10 h in this case.

Memory requirements remain mostly below 10GB for training (Additional file 1: Figure S34c). The exceptions are pIRS needing around 20GB to train on human-sized genomes and ReSeq requiring around 150GB for training on these datasets. However, ReSeq's memory consumption during training strongly depends on the amount of CPUs used (Additional file 1: Figure S35). Therefore, the memory requirements can be reduced substantially by a mixed setup, where only a single thread is used for the statistics calculation and multiple threads are used for the probability estimation. Thus, if needed, ReSeq can still fallback to time and memory requirements that are in the same range as the simulators using single-threaded training scripts. For simulations (Additional file 1: Figure S34f), the memory requirements stay also below 10 GB, except for the multiple instances approaches that require on human-sized genomes 35–90 Gb (ART) or 120Gb (NEAT). Another exception is BEAR that needs close to 14 GB for Rs-Hi4000-Nextera and around 100 GB for Mm-HiX-Unknown and Hs-HiX-TruSeq. In typical systems with 2–8 GB per core, BEAR's high-memory requirements may block many CPUs additional to the 2 used ones, which could lead to effective CPU times much higher than the values in Figure S34d (Additional file 1).

Example: genuine comparison of short-read mappers

After comparing ReSeq to other simulators, we demonstrate its use in an example of a simulation study, where the performance of two popular mapping algorithms, bwa and bowtie2, is compared. For the simulation, we trained ReSeq either on bowtie2 or bwa mappings to verify that performance is not biased by the mapper used for training.

As a first quality check, we compare mapping statistics between simulated and real data for three datasets (Fig. 8a–c). In real data, the number of unmapped pairs and single reads is higher compared to the simulation, which is most pronounced in Ec-Hi2500-TruSeq (Fig. 8b), where the mapping of some reads is prevented by deviations between the reference genome and the true genome underlying the data. In contrast, simulated reads are drawn directly from the reference template and thus do not exhibit mapping issues caused by genome deviations. Furthermore, the bowtie2-based Ec-Hi2000-TruSeq simulation (Fig. 8a) compared to the bwa-based simulation and real data has less unmapped reads for bowtie2 and less soft-clipping, which could be a sign of underestimated adapter content, but the truth data from the simulations show that the difference in adapter content can only account for less than 1/6 of the difference in unmapped reads. Finally, the number of insertions and deletions varies in several cases (Fig. 8a–c), but for the general comparison of mappers here, the differences are too low to influence the result.

As a next step, we compare the mapping-quality distributions between simulated and real data (Fig. 8d–f) and observe that the mapping rates increase in mostly identical steps, when mapping quality requirements are lowered. The overall shifts in mapping rates are an alternative representation of the different percentages of unmapped reads discussed in the previous paragraph. A prominent feature not explained by the general shifts can be seen in Ec-Hi2000-TruSeq (Fig. 8d), where the difference in mapping rate between

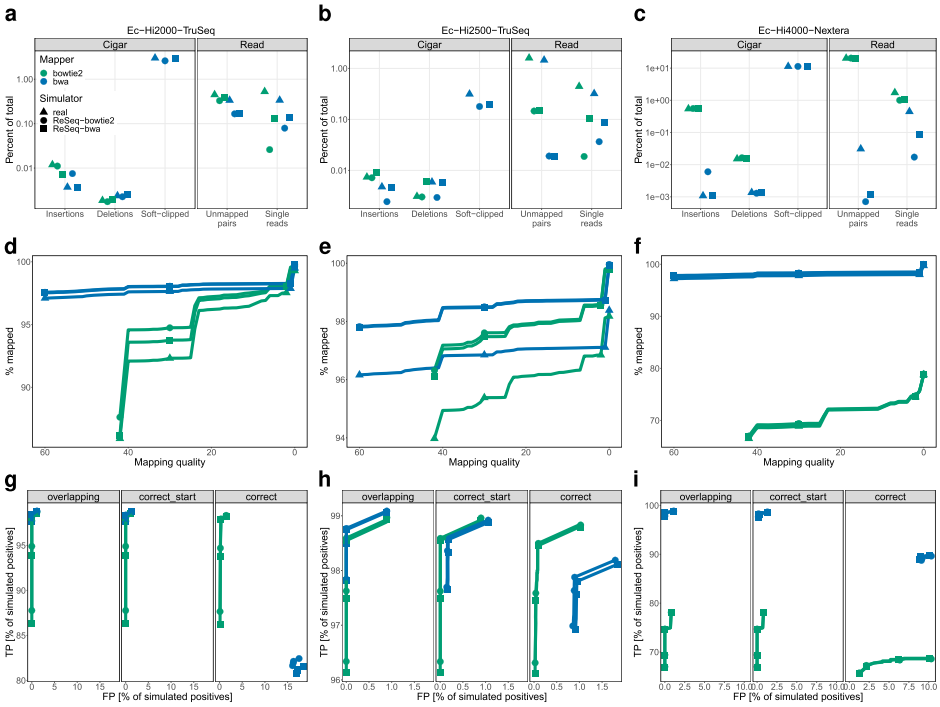

Fig. 8 Comparison between bwa and bowtie2 on *E. coli*. Each dataset has simulations trained on mappings from bwa and bowtie2. **a**–**c** Statistics of mapping outcomes. Bowtie2 does not clip and therefore has no soft-clipped bases. **d**–**f** Mapping quality distribution. Mapping qualities are cumulative, i.e., all mapping qualities at the given score or higher. Markers are only shown for mapping qualities 0,2,30,42 for bowtie and 0,1,30,60 for bwa. **g**–**i** Mapping accuracy for all mapping quality thresholds. Markers are only shown for mapping qualities 0,2,30,42 for bowtie and 0,1,30,60 for bwa. Positives are mapped reads, which fulfill the correctness criteria (TP) or do not (FP). Overlapping: True and mapped positions overlap independent of strand. Correct_start: Perfect match of start position and strand. Correct: Perfect match of start and end positions and strand. **a**, **d**, **g** Ec-Hi2000-TruSeq. **b**, **e**, **h** Ec-Hi2500-TruSeq. **c**, **f**, **i** Ec-Hi4000-Nextera

the two simulations is much more pronounced for high-quality mappings compared to low-quality mappings. In light of the observed decrease in soft-clipped bases (Fig. 8a), the mapping rate differences can be explained by a lower number of simulated reads with many errors ($>$ 10) for the bowtie2-based simulation compared to the bwa-based (Additional file 1: Figure S36). We observe that bwa handles dense errors by clipping the reads, while bowtie2 reduces the mapping quality.

Finally, we can use the truth from the simulations to calculate true positives (TP) and false positives (FP) of read mapping (Fig. 8g-i). In Ec-Hi2000-TruSeq (Fig. 8g), we observe strong differences in TPs between simulations, which are due to the different amount of reads with many errors (Additional file 1: Figure S36) and we use the bwa-based simulation.

Overall, we observe that in the case of strong adapter presence, bwa is the better choice due to its soft-clipping capability (Fig. 8i), although bowtie2 allows better control over FPs with adequate mapping quality thresholds in case both read ends are required to map perfectly. Ec-Hi4000-Nextera also highlights how important adapter simulation is to get an adequate mapping comparison (Additional file 1: Figure S37). In the datasets with lower adapter content (Fig. 8g, h), the recommended mapper depends on the downstream application. If perfect mapping is required, bowtie2 is the better choice, since bwa

often clips too much. If only overlaps with the true position are needed, for example for counting in intervals, bwa slightly outperforms bowtie2. If having a correct start position is sufficient, the mappers perform approximately the same with a minimal preference for bwa if reads with many errors are frequent and a minimal preference for bowtie2 otherwise.

For Ec-Hi4000-Nextera we already mentioned, how different the benchmarking result would be not using ReSeq and thus not simulating adapters (Additional file 1: Figure S37), but mapping qualities suggest that this is not the only case (Additional file 1: Figure S38). Also for Ec-Hi2000-TruSeq, Bc-Hi4000-Nextera, Rs-Hi4000-Nextera, and Ec-Mi-TruSeq, other simulations do not resemble real data (Additional file 1: Figure S38a,d-f,k). Furthermore, both *A. thaliana* datasets seem especially hard to reproduce (Additional file 1: Figure S38g,l). The mapping qualities in the cross-species simulations (Additional file 1: Figure S39) show largely the same results.

Discussion

ReSeq's advancements in faithfully reproducing real data (Fig. 9) can improve many benchmark studies. Short-read error-correction methods could previously not been adequately tested on simulations, because all methods scored nearly perfectly [18]. Many error-correction methods are based on splitting erroneous and correct k-mers at the spectrum's minimum that occurs between the signal peak and the exponential decrease at low k-mer frequencies. ReSeq improves the shape of the exponential decrease by including systematic errors and broadens the signal peak with the extended coverage model. The similarity to real k-mer spectra will drastically increase the difficulty for error-correction methods. Compared to real data, the simulation has the advantage of knowing the true sequence for every read and does not require to derive it by mapping to a reference genome, which can create biases due to repeat regions, ploidy, and deviations between data and reference. However, simulations do not have the full complexity of real data and we recommend a combined evaluation that compares the results from simulations and real data, because it has the potential to hint at inconsistencies caused by deriving the ground truth or imperfectly mimicking real data and to roughly estimate remaining uncertainty in the benchmark.

The situation for assembler benchmarks is similar. We have shown above that ReSeq simulates datasets with realistic N50 values and, in contrast to real data, we can be certain that the dataset is perfectly represented by the reference used to evaluate the correctness of the assembly.

In the case of mapping benchmarks, real data with ground truth is incredibly laborious to produce and in comparison to other simulators, ReSeq includes adapters and reads with a large number of errors, which increases the faithfulness of the results. Since variant calling is performed on mapped reads, these improvements will also translate to variant-caller benchmarks. Additionally, the inclusion of systematic errors adds one source of false positive variant calls that is not present for other simulators.

Despite all the advancements of ReSeq, no simulation is perfect and neither are the ground truth estimates from real data. Therefore, it is very important to assess the scope of a benchmark, clarifying under which circumstances the study results are relevant for real-life applications. The scope of simulations can be tested and defined with comparisons between real and simulated data on statistics accessible for both (e.g.,

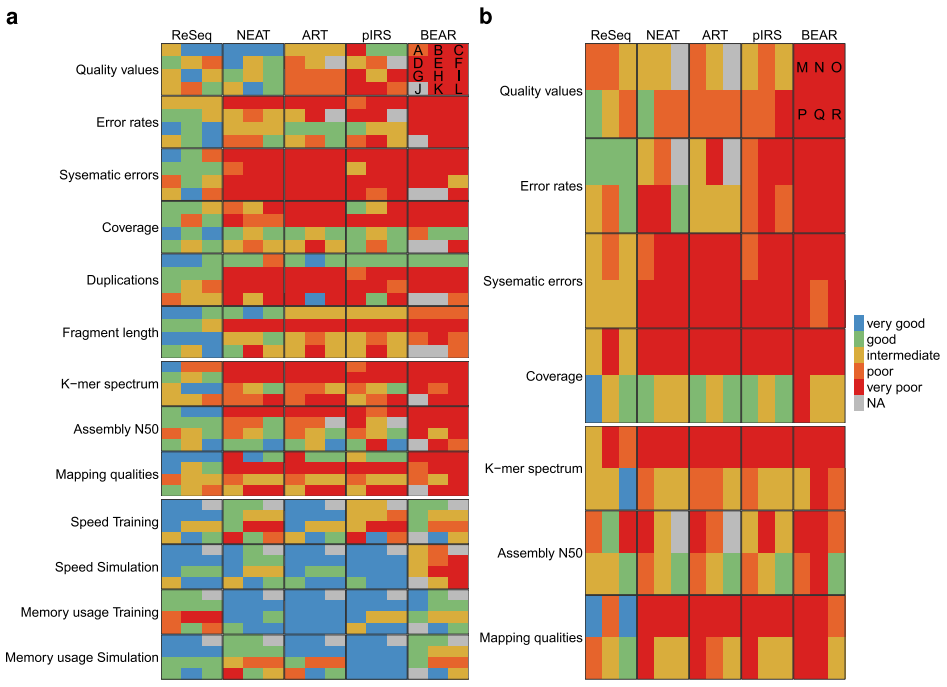

Fig. 9 Overview of benchmarking results. The scoring is based on absolute distances, mean quotients and correlation (see the "Methods" section). **a** Simulations of datasets from their own profiles: **a** Ec-Hi2000-TruSeq. **b** Ec-Hi2500-TruSeq. **c** Ec-Hi2500-TruSeq-asm. **d** Ec-Hi4000-Nextera. **e** Bc-Hi4000-Nextera. **f** Rs-Hi4000-Nextera. **g** At-HiX-TruSeq. **h** Mm-HiX-Unknown. **i** Hs-HiX-TruSeq. **j** Hs-Nova-TruSeq. **k** Ec-Mi-TruSeq. **l** At-BGI. **b** Simulations of datasets from profiles created on another species. The pairings are given in the following notation: "simulated dataset ← training dataset": (M) Ec-Hi4000-Nextera ← Rs-Hi4000-Nextera (N) Bc-Hi4000-Nextera ← Ec-Hi4000-Nextera (O) Rs-Hi4000-Nextera ← Ec-Hi4000-Nextera (P) At-HiX-TruSeq ← Mm-HiX-Unknown (Q) Mm-HiX-Unknown ← Hs-HiX-TruSeq (R) Hs-HiX-TruSeq ← Mm-HiX-Unknown

mapping rates). Our example study, comparing short-read mappers, is limited in three ways: (i) mapped reads do not vary from the reference, (ii) the reference is completely assembled and does not have many repeat regions, and (iii) InDel simulations might need improvements for InDel specific benchmarks. In a more complete study, the scope could be extended by using references with different levels of continuity, repetitiveness, and divergence from the simulated datasets. If InDels are of interest, a more detailed investigation of the differences between real and simulated data would be needed. For example, examining the InDel distribution over the genome could highlight regions where the reference varies from the true genome of the real data. Larger variations can cause distortions in the mapping and induce false InDels into reads, which are then translated into the simulation profiles. An improved reference would reduce the number of false InDels and thereby increase concordance between simulations.

For optimal simulations with ReSeq, the reference genome and variant calls provided for training should be of high-quality and closely match the used reads. Homozygous variants are preferred directly in the training genome instead of the variant file and an adequate representation of heterozygosity is important to correctly mimic the assembly challenge. Furthermore, over 100-fold coverage is advised to capture systematic errors well. Since adapters are learned from the data, the provided reads should not be trimmed. This requires running the Illumina basecaller with deactivated adapter trimming, which

is generally advised, because keeping raw data untrimmed allows to later modify the trimming step that can affect analysis results [42, 43]. Therefore, many but not all public datasets provide untrimmed data.

During the simulation step, the provided template and variants can be synthetic. However, care should be taken that they mimic real genomes in terms of repetitiveness and heterozygosity. Furthermore, the GC content of the template should not deviate strongly from the training reference since GC biases are only accurate in the trainable range.

The only drawback of ReSeq compared to the other simulators are the resource requirements. Due to the increase in required memory with used cores, this comes down to required CPU time. We optimized the code and replaced many time-consuming calculations with estimations, but further optimizations and estimations could be implemented.

The biggest potential gain in representing Illumina genomic sequencing data could be a predictive model for the systematic errors. This would adjust for differences of systematic-error rates across genomes and could increase sensitivity for lowly-covered genomes. In the absence of a predictive model, it is most important to improve low-coverage datasets by handling the strong dependence of systematic-error rate on the coverage. The dependence is based on the discrete nature of errors and leads in combination with unequal coverage over the genome to inaccuracies, which become exacerbated when higher coverages are simulated from low-coverage statistics.

The coverage model could be extended by taking into account interactions between bases for the flanking bias. Especially for enzyme cutting, the effect of a motif likely varies from the sum of the effects of each base. Also, the effect of stretches of GC instead of only the overall GC content could be taken into account, as done for RNA [44].

Finally, additional data types could be supported. For targeted sequencing methods, ReSeq's coverage model could be extended to capture and reproduce the coverage distribution on and around the targets, including off-target effects and target-dependent enrichments. Defining targets in a general form would allow to reuse the extension to simulate exons and introns for RNAseq. However, this requires overlapping targets, which could be handled with equivalence classes. For methods such as 10x Genomics, barcodes, empty droplets, doublets, and index misassignment could all be added. Combining barcodes and a coverage-model extension for overlapping targets would allow to simulate single-cell RNAseq datasets at the read level. Metagenomics and allele-specific methylation are already supported but would profit from a separate simulator comparison.

Conclusion

ReSeq improves the faithfulness of simulated data for all tested datasets. To achieve this, we solved three major challenges. First, we developed a coverage model that can be trained on complete large genomes. Second, we included systematic errors into the simulation. Third, we efficiently represented the important statistics, such that memory requirements remain constrained and the parameters can still be learned from a single real dataset.

Furthermore, ReSeq provides an easy-to-use training of all required models. No manual choice of parameters is needed, which simplifies usage over a wide range of genomes,

Illumina machines, and DNA preparations. The results from the At-BGI dataset suggest that sequencers from BGI can also be successfully simulated. Additionally, ReSeq is more robust to fragmented references during the profile generation compared to pIRS and NEAT.

The simulation of eleven diverse datasets showed the importance of choosing good training data that fit the desired simulation in terms of genomic GC distribution, preprocessing (PCR, fragmentation), and sequencing machine. Therefore, hard-coded profiles should be avoided.

ReSeq and all of its code are available [45].

Methods

Adapter detection

ReSeq determines adapters automatically from the 10-mers of all unmapped parts of the reads, which in case of unmapped reads are the whole read or otherwise, the read segments that exceed the fragment length. ReSeq collects the abundance as starting 10-mer and the total abundance of 10-mers in the unmapped parts. It does so separately for the first and second reads. If the starting 10-mer appears a second time within the unmapped part, the read is rejected and not taken into account for the 10-mer analysis, therefore removing low-complexity regions.

To find the adapters, the starting 10-mer with the highest abundance is used as a seed and extended separately in both directions with the 10-mer that has the highest total abundance of the four possible choices (i.e., added nucleotides). The extension stops when the most abundant 10-mer has no longer an abundance five times higher than the next highest abundance or the 10-mer is already present in the adapter. ReSeq generally filters 10-mers consisting of a single repeated nucleotide and trims the poly-A tail at the end of the adapters. Figure S40 (Additional file 1) demonstrates the procedure for the first (of the paired-end) reads of Ec-Hi2000-TruSeq.

The automatic detection might fail in case of low adapter presence and does not handle more than one adapter for the first or the second read. For those cases, adapters can be specified. The adapter sequences are provided in FASTA format and a corresponding matrix defines the valid adapter pairings for first and second read. For the common Nextera XT v2 and single TruSeq adapters [33], the sequences and matrix are provided with ReSeq. After the adapters have been determined or specified, ReSeq uses the code from skewer [46] to find adapters in unmapped and low quality reads. The seqan library [47] is used for storing and reading from disk and general sequence handling.

Handling of Ns

For the statistics calculation ReSeq ignores positions containing an N in the reference. At the beginning of the simulation, ReSeq replaces regions of 1 to 99 consecutive Ns in the template with random bases. Regions with at least a hundred Ns are replaced with a repeated 4-mer made up from the two bases following and the two bases preceding the region. This prevents replacing the non-assembled parts of the genome with easy-to-assemble random sequences. To use the same bases for the randomly replaced Ns over multiple simulation runs, the replacement can be stored in a FASTA file previous to the simulation.

Iterative proportional fitting

To save memory and to reduce the size of the required inputs for the statistics calculation, ReSeq stores only the two-dimensional margins for each statistic instead of the whole matrix. To use the marginal matrices as a probability distribution during the simulation, they need to be converted back to a single matrix. This is done during the probability estimation step by applying the iterative proportional fitting (ipf) procedure [48], because no formula exists that can combine the full set of two-dimensional margins into a single matrix. Formulas can only satisfy at most a number of two-dimensional margins equal to the dimension of the matrix [49]. The ipf procedure adjusts the values in the matrix such that they fulfill one given margin at a time, looping over all given margins. For each positions *ij* in the currently handled margin *s* all contributing matrix elements m_{ijkln} are adjusted in the following way:

$$m_{ijkln} = m_{ijkln} \frac{s_{ij}}{\sum_k \sum_l \sum_n m_{ijkln}}$$

Repeating those adjustments many times for all margins has been shown to converge toward the solution [48].

In its original form, the iterative proportional fitting stores the full matrix, which potentially requires TBs of memory. However, the fitting can be nicely combined with a data reduction technique described by Fienberg [50]. There, a matrix element is stored as the sum of average differences from the lower dimensional average. For a two-dimensional matrix, the full description of element m_{ij} is:

$$m_{ij} = [1] + [A]_i + [B]_j + [AB]_{ij},$$

where [1] is the average value of the whole matrix, $[A]_i$ is the average value of row *i* minus the overall average [1], $[B]_j$ is the average value of column *j* minus the overall average [1], and $[AB]_{ij}$ is the difference of the element *ij* from the average predicted by [1], $[A]_i$ and $[B]_j$. The full representation does not reduce memory, since the matrix $[AB]$ is of the same size as the original matrix *m*. However, for our four and five-dimensional matrices, truncating the representation at two dimensions drastically reduces the storage requirements. This truncation corresponds to the two-dimensional margins used for the iterative proportional fitting. ReSeq includes the overall average [1] and the one-dimensional vectors $[A]_i, [B]_j, \ldots, [E]_n$ into the two-dimensional matrices. Furthermore, the exponential function is applied to guarantee positive numbers. Thereby, the representation of a five dimensional matrix is reduced to the following combination of ten two-dimensional matrices:

$$m_{ijkln} = e^{[AB]_{ij} + [AC]_{ik} + \cdots + [CE]_{kn} + [DE]_{ln}}$$

Stored probabilities and conditions

Figure 1 lists all dimensions of the six statistics matrices. The matrices are used to query the probabilities along the first dimension, conditional on all other dimensions. The probabilities for the five possible systematic error tendencies (random error and the four nucleotides) depend on the position in a systematic-error region (see the "Distinguishing systematic and random errors" section), the GC content before the given base, the systematic error rate at the first base in an error region, the reference base, the

previous reference base, and the most frequent nucleotide within the last five bases (dominant reference base). If multiple nucleotides have the same frequency within the last five bases, ReSeq chooses the one that is the closest to the current base as dominant reference base. The GC content is calculated in a window from half a read length before the base up to the base. The error-region position is cut into bins of ten bases with the first bin going from position one to ten. The position of zero means either the position where the error region starts, or a position outside of error regions (i.e. no dependence on previous systematic errors). The probabilities for the systematic error rate at each template position depend on the same variables, except that the previous and dominant reference base are replaced by the dominant error.

The sequence quality is based on the GC content of the read, its mean systematic error rate, fragment length, tile, and whether it is the first or second read(template segment). Fragment lengths are binned into bins of ten for this purpose.

The four insertion probabilities (one for each nucleotide) and the deletion probability depend on the position in the current insertion or deletion, the position in the read, the GC content of the error-free read, and the last regular base call. ReSeq also models whether the read position directly before the potential insertion or deletion is itself an insertion or deletion (InDel before), since this means that the InDel will be an extension of the current InDel.

Quality values are assigned based on the drawn sequence quality, previous base quality, read position, systematic error rate, template segment, tile, and reference base.

The regular base calls are determined from the base quality, read position, number of errors already in the read, systematic error rate, template segment, tile, reference base, and dominant error.

In the case of soft-clipped bases in the mapping ReSeq extends the reads to its full length for statistic calculations, except if adapters are detected. This removes the bias introduced by soft-clipping based on low quality, but introduces spurious errors in the rare case of InDel errors in the extended part.

Distinguishing systematic and random errors

To learn the distribution of systematic errors in the genome, they need to be distinguished from non-systematic (i.e., random) errors first. For this purpose, ReSeq determines the most abundant substitution error and its count for every position (reference base, strand). Then the probability is calculated that at least this many errors are observed under the null hypothesis (a *P* value). We assume random errors (the null hypothesis) to have equal probabilities for every base in non-overlapping windows of the genome and to convert bases with equal probability to the three possible erroneous nucleotides. Under this simplification, the number of conversions into a specific nucleotide at a given position follows a Poisson distribution with a mean μ equal to the coverage N_c at the given position times the median error rate r_e divided by three: $\mu = N_c \cdot \frac{r_e}{3}$. The median error rate is calculated in bins of 10^4 reference bases (i.e. $2 \cdot 10^4$ individual positions due to the separation of the two strands).

After the *P* value calculation, we apply the Benjamini-Hochberg procedure [51] and call errors systematic if their adjusted *P* value is less than 0.05. If both strands are reported as systematic, there is potentially a variant at this reference base. Therefore, reference bases are ignored for all statistics, if at this position the most abundant error of the

forward strand matches the complemented most abundant error of the reverse strand. When ReSeq detects a non-variant systematic error, the position on this strand gets assigned the most abundant substitution error as systematic error tendency and its frequency as error rate. Furthermore, the position is set as the start of an systematic-error region with a length of one read length. The error region is defined by the error rate of the systematic error starting it. If another systematic error is detected within the error region it is considered to be caused from the same trigger sequence as the first systematic error, except if it has a higher error rate in which case the earlier error region stops and a new error region begins.

Removing low-quality sites

To filter low-quality sites, the start position of reads on the forward strand and the end position of reads on the reverse strand are considered. Start positions of low-quality reads (mapping quality < 2) are connected if less than 10 other reads start between them. Unconnected positions are likely caused by the read and not the site and thus are treated as high-quality. Connected start positions instead highlight low-quality regions that in addition to all positions between them, are ignored for the coverage model. Similarly, we ignore sites ending on or between connected end positions of low-quality reads.

Bias fit selection

To limit the memory requirements of the GC and flanking bias estimation, sites are separated into combinations of reference sequence and fragment length. From all possible combinations, ReSeq selects a small fraction and performs one fit each. The selection only includes combinations, where the sequence is part of the longest reference sequences covering 80% of the genome (L80) and the fragment length is one of the thirty most abundant fragment lengths for this reference sequence. How many of the thirty most abundant fragment lengths in the selection are fitted depends on the number of reference sequences in the L80. For a single reference sequence, all thirty are fitted. For twenty and more sequences, only five fits per sequence are equally distributed over the thirty most abundant fragment lengths. For 2–19 sequences, the number of fits per reference sequence is gradually reduced. For example, the single sequence of *E. coli* results in thirty fits, while the four (of seven) sequences of *A. thaliana* in the L80, result in $4 \cdot 15 = 60$ fits.

Negative binomial model

The coverage is modelled with a negative binomial leading to the following full likelihood over all sites n:

$$\prod_n \binom{k_n + r_n - 1}{k_n} \left(1 - \frac{\mu_n}{\mu_n + r_n}\right)^{r_n} \left(\frac{\mu_n}{\mu_n + r_n}\right)^{k_n}$$

where k_n is the observed count, μ_n is the mean, and r_n is the dispersion at site n. The mean is written as a product:

$$\mu_n = \tilde{N} b_{seq}(seq_n) b_{len}(len_n) b_{GC}(GC_n) b_{start,n} b_{end,n}$$

with $\tilde{N}$ as the genome-wide normalization and the different b as the biases for this site. We cannot determine b_{seq} and b_{len}, because they are identical for all sites n within a fit.

Thus, those biases are absorbed into the normalization:

$$\mu_n = N b_{GC}(GC_n) b_{start,n} b_{end,n}$$

This model could alternatively be described as a generalized linear model with a logarithm link. However, the sheer size of the necessary matrix of explanatory variables makes it computationally prohibitive. Furthermore, the flanking bias uses the logit link function internally.

For the flanking bias, we tested two choices to combine the individual positions: sum and product. If we choose the product of the positions p, the flanking bias at the start results in:

$$b_{start,n} = \prod_p b_{f,p}(start_{n,p})$$

In contrast, the sum over the positions can become negative, so we apply twice the inverse logit. Since the center of the logit is approximately linear, this should keep the summation behavior:

$$b_{start,n} = \frac{2}{1 + e^{-\sum_p b_{f,p}(start_{n,p})}}$$

For the GC bias, we use an exponential to prevent negative values and thus keep the canonical link function:

$$b_{GC}(GC_n) = e^{b_{GC,spline}(GC_n)}$$

Applying the inverse logit, similar to the flanking bias, improves the speed or convergence rate for some datasets, but generally performs worse (Additional file 1: Figure S41).

Finally, the dispersion is described with two parameters, α and β:

$$r_n = \frac{\mu_n}{\alpha + \beta \mu_n}.$$

The log-likelihoods are maximized with the limited-memory Broyden-Fletcher-Goldfarb-Shanno algorithm (L-BFGS) [52, 53] implemented in the NLopt package [54] due to its low number of required likelihood calculations. To further improve convergence and speed, ReSeq performs the fit in three steps (Fig. 3a). It starts by fitting a Poisson, which allows to analytically calculate the optimal GC bias for each bin. In a second step, the individual GC bins are connected using the cubic natural spline, while the spline's knots are greedily adjusted to achieve the highest maximum likelihood. In the last step, ReSeq maximizes the likelihood for the full negative-binomial model with the knots fixed to the greedily determined values. Afterwards, the normalized biases and dispersion from all converged reference sequences and fragment length combinations are merged using a median to be robust against outliers.

GC knot adjustment

In the second step of the GC and flanking bias fit for the *Coverage model*, the GC bias is fitted with a spline, which requires defining the six knot positions. The highest and lowest knot are fixed at the highest and lowest GC bin. GC bins without at least one counted fragment in the real data cannot host a knot. Beyond the outer knots, the GC bias is assumed to remain constant. For the inner four knots, ReSeq tries to find the ideal distribution. First, the four knots are distributed to have an equal spacing according to the number of sites. Then ReSeq maximizes the Poisson likelihood with the flanking biases

fixed to the values determined in the first step of the GC and flanking bias fit (Poisson without GC spline). It additionally performs this maximization for all knot distributions that can be reached from the starting distribution by moving only a single of the inner knots, while maintaining knot order. From the performed fits, ReSeq greedily chooses the knot distribution with the highest likelihood as the new starting distribution and repeats this procedure until convergence.

GC biases renormalization

After the third step of the GC and flanking bias fit, the 101 GC biases are renormalized to remove the normalization parameter and make them comparable across all combinations of reference sequence and fragment length. For this, ReSeq only takes the best-represented GC biases into account to calculate the renormalization factor, which prevents the highly-variable bins from disturbing the normalization. The best-represented GC biases are those with the most sites that cover together at least 80% of the total number of sites. After the normalization, the mean of those best-represented biases equals 1.

Parameter combination with medians

The flanking biases and the dispersion parameters from the individual fits for each combination of reference sequence and fragment length are combined with a standard median. In contrast, the GC biases need a weighted median, because not all fits contain a high amount of information about every GC bin.

The unnormalized weight of a given GC bias in a fit depends on the amount of sites $l_{sites,GC}$ with the corresponding GC content and equals to $\sqrt{l_{sites,GC}} - 1$. To account for the connection between the biases due to the spline, a fraction of the weight is also added to the other bins depending on the distance:

$$w_{GC} = \sqrt{l_{sites,GC}} - 1 + \sum_{d=1}^{GC} \frac{\sqrt{l_{sites,GC-d}} - 1}{10^d} + \sum_{d=1}^{100-GC} \frac{\sqrt{l_{sites,GC+d}} - 1}{10^d}$$

Those weights are then normalized such that the sum over all GC bins equals 1 in each fit, which makes sure that all fits are equally important.

Reference-sequence and fragment-length bias estimation

After the other biases have been fitted, ReSeq estimates the reference-sequence and fragment-length biases. It starts by summing flanking and GC biases over all sites for selected combinations of fragment length and reference sequence $b_{sum,ref,len}$. Additionally, the counts of mapped pairs for each reference sequence, C_{ref}, and each fragment length, C_{len}, are obtained. The reference-sequence bias b_{ref} and the fragment-length bias b_{len} are then iteratively adjusted until convergence to follow the equations:

$$b_{ref} = \frac{C_{ref}}{\sum_{len} b_{len} b_{sum,ref,len}}$$

$$b_{len} = \frac{C_{len}}{\sum_{ref} b_{ref} b_{sum,ref,len}}$$

Corrected reference sequence and fragment length counts

To minimize biases, ReSeq only counts read pairs that pass all filters (mapping quality, forward-reverse order, maximum fragment length). Furthermore, the read pairs must not fall on a site excluded due to low-quality mappings. To avoid a distortion of the counts from the last filter, ReSeq scales the reference-sequence counts from the high-quality part (that remains after filtering) to the full sequence.

To properly handle short fragment lengths, ReSeq searches for adapters, whose presence frequently leads to no or low-quality mappings and subsequent exclusion from the analysis. Hence, if the scan detects adapters, ReSeq will recover otherwise excluded unmapped and low-quality reads, incorporating the pairs into the fragment-length distribution.

Bias sum estimation

Summing up the bias over all sites is an essential part of ReSeq. It is necessary to calculate the fragment-length biases during the statistics calculation and to normalize the biases during the simulation. To speed up the bias sum, ReSeq calculates in both use cases the sum only every 20th fragment length (the samples) and estimates the other fragment lengths. The sampling starts at the first fragment length with non-zero counts and stops as soon as it encounters a zero count, except if this would exclude sampling positions with at least 10 counts. The estimation of the non-sampled fragment length is different in the statistics calculation and the simulation due to different requirements.

In the statistics calculation, ReSeq first obtains the fragment-length biases for the sampled lengths. After this fit, ReSeq extends the estimates to all biases by using the ratio of counts over bias from the samples. This ratio is almost constant except for the shorter fragment length. Therefore, it is much easier to describe by a natural spline and requires less sampling than the biases itself. ReSeq estimates the parameters of the natural spline by a non-linear least square fit to the samples using the L-BFGS algorithm [52, 53] for minimization. ReSeq starts by assigning a knot to every sample and then gradually removes the knots for which the sum over the squared residuals $S = \sum_i r_i^2$ increases the least, until only 6 knots are left. Occasionally, ReSeq performs a greedy search for the optimal knot positions during the reduction, which is similar to the one described above in *GC knot adjustment*. Finally, ReSeq uses the optimized spline to calculate the bias estimates for all fragment lengths.

In the simulation, ReSeq only requires the total bias sum over the whole genome and all fragment lengths. Thus, the precision of individual values is secondary and another approach is chosen, where ReSeq applies a spline to interpolate the ratio of the bias sum over the fragment-length bias using every sample as a knot. From the ratios, the bias sums are retrieved and subsequently summed over all fragment lengths. Additionally, the simulation requires the maximum bias for each fragment length, which ReSeq estimates by using the maximum ratio over all samples multiplied with the respective fragment-length bias.

Reproducibility of simulator comparison

Except for the parts that are specifically mentioned in this paragraph, everything for this paper was run through snakemake [55] with the scripts available on GitHub [56]. Figure 3 and Figures S1, S10, S11, S40 and S41 (Additional file 1) represent intermediate values

from ReSeq. The output of the intermediate values requires additional calculations and disk writing in performance-critical functions. Therefore, it is triggered by hard-coded flags and not by parameters to allow the compiler to remove unnecessary code and prevent speed-loss during normal use, when the additional outputs are not needed. Due to the hard-coded flags, the creation of the intermediate values is not included in the snakemake pipeline. Instead, we added the csv files containing the intermediate values to the git repository. Moreover, the computing requirements were manually extracted from the timing files and inserted into a csv file included in the git repository. The IGV [37, 38] screenshots included in Fig. 2 are also in the repository, so that all plotting scripts can be called. We compared the newest versions of the simulators (Additional file 1: Table S1) and all other software and their versions are stated in Table S2 (Additional file 1) [57–60]. The datasets with sample IDs Ecoli1_L001, Bcereus1_L001, Rsphaeroides1_L001, Ecoli1_L002, Ecoli1_L003, Ecoli2_L001, and Ecoli3_L001 were downloaded as raw data from Illumina BaseSpace. We ran the conversion from bcl to fastq format locally with disabled adapter trimming.

Simulations

The references for each dataset were corrected with pilon [61] before subsequent trainings, simulations, and evaluations. If not stated otherwise, all reads were mapped with bowtie2 [62] in its default end-to-end alignment mode that does not allow for soft-clipped bases. For diploid organisms, freebayes [63] called the variants and we filtered out reported variants with qualities below 10. For simulators requiring coverage as a parameter, we calculated the median from the coverage summary of samtools stats [64]. Since BEAR only accepts the number of reads as a parameter, we set it to match the total number of reads in the real dataset. Additionally, BEAR requires the sequence abundances, where we provided the number of mapped reads divided by the total number of mapped reads or in case of the cross-species simulations, the sequence length divided by the total genome length. The insertion and deletion rates provided to ART were obtained from parsing the CIGAR strings in the bam files. With M, I, D being the number of matches, insertions, and deletions in the CIGAR strings, respectively, the insertion rate is $\frac{I}{M+D}$ and the deletion rate is $\frac{D}{M+D}$. The fragment lengths obtained from the mapped reads were fitted with a Gaussian to retrieve the fragment length parameters. In the case of Bc-Hi4000-Nextera, Ec-Hi4000-Nextera, and Rs-Hi4000-Nextera, the fragment length was not Gaussian distributed, but followed an exponential decay. Therefore, we set the mean to the lowest value accepted by all simulators, which is the read length plus one, and manually fitted the variance for the second half of the Gaussian. The length of sliding windows were set for all datasets to the specified fragment length.

Evaluation of simulator performance

To evaluate performance, we counted 51-mers with Jellyfish [65], except for the human and mouse datasets, where memory consumption skyrocketed and kmc [66] using disk storage was applied instead. The preqc module of sga [67] calculated in a reference-free way the mean quality values and error rates by position as well as the simulated contig length N50. We modified the preqc plotting script to adjust it to the general style. Further plots were obtained with ReSeq using bowtie2 [62] mappings. In case of simulated datasets without adapters, the evaluation with ReSeq requires to specify TruSeq adapters

as a decoy. To calculate the coverage correlation, we counted the base coverage with samtools depth [64]. For the short-read assembly, we called sga [68] with default parameters, except for the ropebwt algorithm for indexing. The assembly statistics were calculated by QUAST [35] with the minimum contig length set to zero.

Speed and memory benchmarks

We benchmarked the speed and memory usage with GNU time. The single-threaded processes were run on a machine with four AMD Opteron(tm) Processor 6376 with a total of 64 cores and 512GB of memory operating Ubuntu 16.04.9. Due to their single threaded nature, other programs were running in parallel to get a realistic use case. The multi-threaded processes ran on single cluster nodes with 384 GB memory and two Intel Xeon Gold 6126 resulting in 48 vCPUs. All 48 possible threads were provided to the processes.

To report CPU time, we summed user and system time. We summed CPU times of all processes in a category, as well as elapsed times for multi-threaded processes. For single-threaded processes, we summed the elapsed times only if the processes were dependent and needed to be run in series. Otherwise, we took the maximum. For maximum memory, we report the maximum resident set size. The memory maxima were summed if single-threaded processes are run in parallel and the maximum was taken if they are run in series. For multi-threaded processes, we always took the maximum memory.

For the multi-instance approaches with NEAT and ART, we ran 48 parallel instances on the cluster nodes and calculated the times and memory requirements as for single-threaded processes. For ART the coverage of each instance was adjusted, while for NEAT the dedicated splitting functionality was used.

Summary scoring

The scores in Fig. 9 are based on three metrics:

1. The sum of the absolute differences between the real and simulated values ($abs = \sum_x |real(x) - sim(x)|$).
2. The mean quotient of the real and simulated values with the higher value as the denominator, so that the score goes from 0 to 1:

$$quot = mean_x \begin{pmatrix} \frac{real(x)}{sim(x)} & real(x) < sim(x) \\ \frac{sim(x)}{real(x)} & real(x) \geq sim(x) \end{pmatrix}$$

3. The Spearman correlation *cor* between real and simulated values.

For ART, pIRS, and NEAT, we evaluate the simulations of Ec-HiSeq2500-TruSeq-asm with the median coverage from Ec-HiSeq2500-TruSeq and for ReSeq the simulation of Ec-Mi-TruSeq with the median coverage. In all other cases, the default simulations are used.

Quality values are scored based on the absolute differences as very good ($abs < 20$), good ($abs < 53$), intermediate ($abs < 100$), poor ($abs < 500$), or very poor otherwise. pIRS is penalized by one category for not separating quality values of first and second reads.

Error rates are normalized by the mean error rate of the real data and then separated based on the absolute distance into very good ($abs < 20$), good ($abs < 35$), intermediate ($abs < 65$), poor ($abs < 100$), or very poor otherwise. pIRS is penalized by one category for not separating error rates of first and second reads.

For systematic errors, coverage, duplications and fragment length, the values used in the metrics are divided by the total to make them independent of genome size or number of reads ($y(x) = \frac{\tilde{y}(x)}{\sum_k \tilde{y}(k)}$).

The systematic error distributions are truncated when the real values fall below 10 counts or when the lowest coverage threshold to include 99% of the genome is reached. The coverage threshold removes repeats and plasmids, which would otherwise dominate the metrics. Based on all three metrics, the systematic errors are scored as very good ($abs < 0.25$, $quot > 0.6$, $cor > 0.99$), good ($abs < 0.5$, $quot > 0.4$, $cor > 0.97$), intermediate ($abs < 0.7$, $quot > 0.2$, $cor > 0.95$), poor ($abs < 0.85$, $quot > 0.1$, $cor > 0.85$), or very poor otherwise.

The coverage is separated based on the absolute difference into very good ($abs < 0.1$), good ($abs < 0.35$), intermediate ($abs < 0.62$), poor ($abs < 0.9$), or very poor otherwise.

The duplication distributions are truncated when the real values reached zero and scored based on the mean quotients as very good ($quot > 0.75$), good ($quot > 0.25$), intermediate ($quot > 0.05$), poor ($quot > 0.01$), or very poor otherwise. Simulators with duplication values exceeding the values of real data are penalized by one category if less than 0.01% of the fragments are affected or two categories otherwise.

The fragment length is separated based on the absolute difference into very good ($abs < 0.01$), good ($abs < 0.05$), intermediate ($abs < 0.2$), poor ($abs < 0.6$), or very poor otherwise. BEAR is penalized by a reduction of one category for the trimming of reads, which violates the fixed read length of real data.

For the k-mer evaluation, the 51-mer spectrum is split at the minimum between the exponential decrease and the signal peak in real data indicated in Fig. 6. Everything up to the minimum is evaluated with the mean quotient and correlation, while everything after the minimum is divided by the total number of 51-mers in this part and then scored based on the absolute differences. The scores are very good ($abs < 0.2$, $quot > 0.75$, $cor > 0.98$), good ($abs < 0.37$, $quot > 0.6$, $cor > 0.9$), intermediate ($abs < 0.52$, $quot > 0.3$, $cor > 0.75$), poor ($abs < 0.75$, $quot > 0.05$, $cor > 0.48$), or very poor otherwise.

The amount of reads for all mapping qualities are normalized by the total number of reads and then categorized based on the absolute difference as very good ($abs < 0.05$), good ($abs < 0.1$), intermediate ($abs < 0.25$), poor ($abs < 0.4$), or very poor otherwise.

The simulated N50 spectra have only 15 points and their values fluctuate considerably for some datasets. Additionally, the range of values varies substantially between datasets. Thus, the above metrics do not separate the quality of simulations well and instead we use the count of simulated N50 values that are between the minimum and maximum of the real N50 value and their 1, 2, or 4 neighboring N50 values on both sides. Missing neighbours for the first and last points are replaced with a value twice the first/last value minus the second/second last value. The given scores are very good ($count_1 \geq 13$, $count_2 \geq 14$, $count_4 \geq 15$), good ($count_1 \geq 5$, $count_2 \geq 10$, $count_4 \geq 12$), intermediate ($count_2 \geq 5$,

$count_4 \geq 9$), poor ($count_4 \geq 6$), or very poor otherwise. Simulations with a consistently wrong direction of change in N50 over at least 4 points are reduced by one category.

The speed is evaluated based on the elapsed times t and separated into very good ($t < 1$h), good ($t < 4$h), intermediate ($t < 24$h), poor ($t < 72$h), or very poor otherwise. If they exist, the threaded or multi-process versions of the simulators are used.

The memory consumption *mem* was scored as very good (*mem* < 1GB), good (*mem* < 8GB), intermediate (*mem* < 32GB), poor (*mem* < 128GB), or very poor otherwise. Consistent with the speed evaluation, we use the threaded or multi-process versions of the simulators if they exist.

Supplementary Information

The online version contains supplementary material available at https://doi.org/10.1186/s13059-021-02265-7.

Additional file 1: Supplementary figures, tables and formulas.
Additional file 2: QUAST assembly report.
Additional file 3: ReSeq Manual.
Additional file 4: Review history.

Acknowledgments
The authors thank members of the Robinson Lab at the University of Zurich for valuable feedback and the Functional Genomics Center Zurich for providing helpful datasets.

Peer review information
Barbara Cheifet was the primary editor on this article and managed its editorial process and peer review in collaboration with the rest of the editorial team.

Review history
The review history is available as Additional file 4.

Authors' contributions
Stephan Schmeing (SMG) conceived the study. MDR supervised the study. SMG developed the simulator and its methods and performed the benchmarking. SMG and MDR wrote the manuscript. All authors read and approved the final manuscript.

Authors' information
Twitter handles: @StephanSchmeing (Stephan Schmeing); @markrobinsonca (Mark D. Robinson).

Funding
This work was supported by the University Research Priority Program (URPP) *Evolution in Action* of the University of Zurich. This work made use of infrastructure provided by S3IT (www.s3it.uzh.ch), the Service and Support for Science IT team at the University of Zurich.

Availability of data and materials
Only public data were used during this study. The datasets with accession IDs SRR490124, SRR3191692, ERR2017816, ERR3085830, ERR1955542, PRJEB33197 (ERR3518653, ERR3519640, ERR3522379, ERR3522985, ERR3528464, ERR3528894, ERR3530147, ERR3530148, ERR3556082, ERR3556092, ERR3559869, ERR3561121, ERR3561123, ERR3561210, ERR3561217, ERR3561247), DRR058060, and PRJNA562949 (SRR10037570, SRR10037571, SRR10037572, SRR10037573, SRR10037578, SRR10037579) can be downloaded from the European Nucleotide Archive: https://www.ebi.ac.uk/ena
The datasets with sample IDs Ecoli1_L001, Bcereus1_L001, Rsphaeroides1_L001, Ecoli1_L002, Ecoli1_L003, Ecoli2_L001, and Ecoli3_L001 are part of the "HiSeq 4000: Nextera XT (B.cereus, E.coli, R.sphaeroides)" project in Illumina's BaseSpace Sequence Hub: https://basespace.illumina.com
ReSeq and all of its code are available under the MIT License at: https://github.com/schmeing/ReSeq [45].
The snakemake pipeline and custom scripts used for this publication are available under https://github.com/schmeing/ReSeq-paper [56].
The frozen ReSeq and pipeline versions used for the preparation of the manuscript are available on Zenodo [69]. The manual of the frozen ReSeq version is provided as Additional file 3.

Ethics approval and consent to participate
Ethics approval is not applicable for this work.

Competing interests
The authors declare that they have no competing interests.

Received: 17 July 2020 Accepted: 7 January 2021
Published online: 19 February 2021

References

1. Patch AM, Nones K, Kazakoff SH, Newell F, Wood S, Leonard C, Holmes O, Xu Q, Addala V, Creaney J, Robinson BW, Fu S, Geng C, Li T, Zhang W, Liang X, Rao J, Wang J, Tian M, Zhao Y, Teng F, Gou H, Yang B, Jiang H, Mu F, Pearson JV, Waddell N. Germline and somatic variant identification using BGISEQ-500 and HiSeq X Ten whole genome sequencing. PLoS ONE. 2018;13(1):0190264.
2. Jeon SA, Park JL, Kim JH, Kim JH, Kim YS, Kim JC, Kim SY. Comparison of the MGISEQ-2000 and Illumina HiSeq 4000 sequencing platforms for RNA sequencing. Genomics Inform. 2019;17(3):32.
3. Robinson MD, Vitek O. Benchmarking comes of age. Genome Biol. 2019;20(1):205.
4. Mangul S, Martin LS, Hill BL, Lam AK, Distler MG, Zelikovsky A, Eskin E, Flint J. Systematic benchmarking of omics computational tools. Nat Commun. 2019;10(1):1393.
5. Li S, Tighe SW, Nicolet CM, Grove D, Levy S, Farmerie W, Viale A, Wright C, Schweitzer PA, Gao Y, Kim D, Boland J, Hicks B, Kim R, Chhangawala S, Jafari N, Raghavachari N, Gandara J, Garcia-Reyero N, Hendrickson C, Roberson D, Rosenfeld J, Smith T, Underwood JG, Wang M, Zumbo P, Baldwin DA, Grills GS, Mason CE. Multi-platform assessment of transcriptome profiling using RNA-seq in the ABRF next-generation sequencing study. Nat Biotechnol. 2014;32(9):915–25.
6. Krusche P, Trigg L, Boutros PC, Mason CE, De La Vega FM, Moore BL, Gonzalez-Porta M, Eberle MA, Tezak Z, Lababidi S, Truty R, Asimenos G, Funke B, Fleharty M, Chapman BA, Salit M, Zook JM, the Global Alliance for Genomics and Health Benchmarking Team. Best practices for benchmarking germline small-variant calls in human genomes. Nat Biotechnol. 2019;37:555–60.
7. Zook JM, Chapman B, Wang J, Mittelman D, Hofmann O, Hide W, Salit M. Integrating human sequence data sets provides a resource of benchmark snp and indel genotype calls. Nat Biotechnol. 2014;32:246–51.
8. Zook JM, Catoe D, McDaniel J, Vang L, Spies N, Sidow A, Weng Z, Liu Y, Mason CE, Alexander N, Henaff E, McIntyre ABR, Chandramohan D, Chen F, Jaeger E, Moshrefi A, Pham K, Stedman W, Liang T, Saghbini M, Dzakula Z, Hastie A, Cao H, Deikus G, Schadt E, Sebra R, Bashir A, Truty RM, Chang CC, Gulbahce N, Zhao K, Ghosh S, Hyland F, Fu Y, Chaisson M, Xiao C, Trow J, Sherry ST, Zaranek AW, Ball M, Bobe J, Estep P, Church GM, Marks P, Kyriazopoulou-Panagiotopoulou S, Zheng GXY, Schnall-Levin M, Ordonez HS, Mudivarti PA, Giorda K, Sheng Y, Rypdal KB, Salit M. Extensive sequencing of seven human genomes to characterize benchmark reference materials. Sci Data. 2016;3:160025.
9. Eberle MA, Fritzilas E, Krusche P, Källberg M, Moore BL, Bekritsky MA, Iqbal Z, Chuang H-Y, Humphray SJ, Halpern AL, Kruglyak S, Margulies EH, McVean G, Bentley DR. A reference data set of 5.4 million phased human variants validated by genetic inheritance from sequencing a three-generation 17-member pedigree. Genome Res. 2017;27:157–64.
10. Li H, Bloom JM, Farjoun Y, Fleharty M, Gauthier L, Neale B, MacArthur D. A synthetic-diploid benchmark for accurate variant-calling evaluation. Nat Methods. 2018;15:595–7.
11. Clevenger J, Chavarro C, Pearl SA, Ozias-Akins P, Jackson SA. Single nucleotide polymorphism identification in polyploids: a review, example, and recommendations. Mol Plant. 2015;8(6):831–46.
12. Escalona M, Rocha S, Posada D. A comparison of tools for the simulation of genomic next-generation sequencing data. Nat Rev Genet. 2016;17:459–69.
13. Alosaimi S, Bandiang A, van Biljon N, Awany D, Thami PK, Tchamga MSS, Kiran A, Messaoud O, Hassan RIM, Mugo J, Ahmed A, Bope CD, Allali I, Mazandu GK, Mulder NJ, Chimusa ER. A broad survey of DNA sequence data simulation tools. Brief Funct Genomics. 2020;19(1):49–59.
14. Huang W, Li L, Myers JR, Marth GT. Art: a next-generation sequencing read simulator. Bioinformatics. 2012;28:593–4.
15. Hu X, Yuan J, Shi Y, Lu J, Liu B, Li Z, Chen Y, Mu D, Zhang H, Li N, Yue Z, Bai F, Li H, Fan W. pirs: profile-based illumina pair-end reads simulator. Bioinformatics. 2012;28:1533–5.
16. Stephens ZD, Hudson ME, Mainzer LS, Taschuk M, Weber MR, Iyer RK. Simulating next-generation sequencing datasets from empirical mutation and sequencing models. PLoS ONE. 2016;11(11):e0167047.
17. Johnson S, Trost B, Long JR, Pittet V, Kusalik A. A better sequence-read simulator program for metagenomics. BMC Bioinformatics. 2014;15 Suppl 9:S14.
18. Molnar M, Ilie L. Correcting illumina data. Brief Bioinforma. 2015;16:588–99.
19. Earl D, Bradnam K, John JS, Darling A, Lin D, Fass J, Yu HOK, Buffalo V, Zerbino DR, Diekhans M, Nguyen N, Ariyaratne PN, Sung W-K, Ning Z, Haimel M, Simpson JT, Fonseca NA, Birol İ, Docking TR, Ho11 IY, Rokhsar DS, Chikhi R, Lavenier D, Chapuis G, Naquin D, Maillet N, Schatz MC, Kelley DR, Phillippy AM, Koren S, Yang S-P, Wu W, Chou W-C, Srivastava A, Shaw TI, Ruby JG, Skewes-Cox P, Betegon M, Dimon MT, Solovyev V, Seledtsov I, Kosarev P, Vorobyev D, Ramirez-Gonzalez R, Leggett R, MacLean D, Xia F, Luo R, Li Z, Xie Y, Liu B, Gnerre S, MacCallum I, Przybylski D, Ribeiro FJ, Yin S, Sharpe T, Hall G, Kersey PJ, Durbin R, Jackman SD, Chapman JA, Huang X, DeRisi JL, Caccamo M, Li Y, Jaffe DB, Green RE, Haussler D, Korf I, Paten B. Assemblathon 1: a competitive assessment of de novo short read assembly methods. Genome Res. 2011;21:2224–41.
20. Salzberg SL, Phillippy AM, Zimin A, Puiu D, Magoc T, Koren S, Treangen TJ, Schatz MC, Delcher AL, Roberts M, Marçais G, Pop M, Yorke JA. Gage: a critical evaluation of genome assemblies and assembly algorithms. Genome Res. 2012;22(3):557–67.
21. Bradnam KR, Fass JN, Alexandrov A, Baranay P, Bechner M, Birol I, Boisvert S, Chapman JA, Chapuis G, Chikhi R, Chitsaz H, Chou W-C, Corbeil J, Fabbro CD, Docking TR, Durbin R, Earl D, Emrich S, Fedotov P, Fonseca NA, Ganapathy G, Gibbs RA, Gnerre S, Godzaridis É, Goldstein S, Haimel M, Hall G, Haussler D, Hiatt JB, Ho IY, Howard J, Hunt M, Jackman SD, Jaffe DB, Jarvis ED, Jiang H, Kazakov S, Kersey PJ, Kitzman JO, Knight JR, Koren S, Lam T-W, Lavenier D, Laviolette F, Li Y, Li Z, Liu B, Liu Y, Luo R, MacCallum I, MacManes MD, Maillet N, Melnikov S, Naquin D, Ning Z, Otto TD, Paten B, Paulo OS, Phillippy AM, Pina-Martins F, Place M, Przybylski D, Qin X, Qu C, Ribeiro FJ, Richards S, Rokhsar DS, Ruby JG, Scalabrin S, Schatz MC, Schwartz DC, Sergushichev A, Sharpe T, Shaw TI, Shendure J, Shi Y, Simpson JT, Song H, Tsarev F, Vezzi F, Vicedomini R, Vieira BM, Wang J, Worley KC,

Yin S, Yiu S-M, Yuan J, Zhang G, Zhang H, Zhou S, Korf IF. Assemblathon 2: evaluating de novo methods of genome assembly in three vertebrate species. GigaScience. 2013;2(1):10.
22. 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(5):R51.
23. Aird D, Ross MG, Chen W-S, Danielsson M, Fennell T, Russ C, Jaffe DB, Nusbaum C, Gnirke A. Analyzing and minimizing PCR amplification bias in illumina sequencing libraries. Genome Biol. 2011;12(2):R18.
24. Benjamini Y, Speed TP. Summarizing and correcting the GC content bias in high-throughput sequencing. Nucleic Acids Res. 2012;40(10):e72.
25. Kia A, Gloeckner C, Osothprarop T, Gormley N, Bomati E, Stephenson M, Goryshin I, He MM. Improved genome sequencing using an engineered transposase. BMC Biotechnol. 2017;17(1):6.
26. Schirmer M, D'Amore R, Ijaz UZ, Hall N, Quince C. Illumina error profiles: resolving fine-scale variation in metagenomic sequencing data. BMC Bioinformatics. 2016;17:125.
27. Ma X, Shao Y, Tian L, Flasch DA, Mulder HL, Edmonson MN, Liu Y, Chen X, Newman S, Nakitandwe J, Li Y, Li B, Shen S, Wang Z, Shurtleff S, Robison LL, Levy S, Easton J, Zhang J. Analysis of error profiles in deep next-generation sequencing data. Genome Biol. 2019;20(1):50.
28. Pfeiffer F, Gröber C, Blank M, Händler K, Beyer M, Schultze JL, Mayer G. Systematic evaluation of error rates and causes in short samples in next-generation sequencing. Sci Rep. 2018;8(1):10950.
29. Keegan KP, Trimble WL, Wilkening J, Wilke A, Harrison T, D'Souza M, Meyer F. A platform-independent method for detecting errors in metagenomic sequencing data: DRISEE. PLoS Comput Biol. 2012;8(6):1002541.
30. Nakamura K, Oshima T, Morimoto T, Ikeda S, Yoshikawa H, Shiwa Y, Ishikawa S, Linak MC, Hirai A, Takahashi H, Altaf-Ul-Amin M, Ogasawara N, Kanaya S. Sequence-specific error profile of illumina sequencers. Nucleic Acids Res. 2011;39(13):e90.
31. Meacham F, Boffelli D, Dhahbi J, Martin DI, Singer M, Pachter L. Identification and correction of systematic error in high-throughput sequence data. BMC Bioinformatics. 2011;12:451.
32. Tan G, Opitz L, Schlapbach R, Rehrauer H. Long fragments achieve lower base quality in illumina paired-end sequencing. Sci Rep. 2019;9(1):2856.
33. Illumina Adapter Sequences (1000000002694 V14). https://support.illumina.com/content/dam/illumina-support/documents/documentation/chemistry_documentation/experiment-design/illumina-adapter-sequences-1000000002694-14.pdf. Accessed 06 Jan 2021.
34. Oligos and Primers for BGISEQ/DNBSEQ/MGISEQ Library Preparation. https://en.mgitech.cn/Download/download_file/id/71. Accessed 06 Jan 2021.
35. Gurevich A, Saveliev V, Vyahhi N, Tesler G. QUAST: quality assessment tool for genome assemblies. Bioinformatics. 2013;29(8):1072–5.
36. Eren AM, Morrison HG, Huse SM, Sogin ML. DRISEE overestimates errors in metagenomic sequencing data. Brief Bioinforma. 2014;15(5):1072–5.
37. Robinson JT, Thorvaldsdóttir H, Winckler W, Guttman M, Lander ES, Getz G, Mesirov JP. Integrative genomics viewer. Nat Biotechnol. 2011;29:24–6.
38. Thorvaldsdóttir H, Robinson JT, Mesirov JP. Integrative Genomics Viewer (IGV): high-performance genomics data visualization and exploration. Brief Bioinform. 2013;14:178–92.
39. Wingett S. Illumina patterned flow cells generate duplicated sequences. https://sequencing.qcfail.com/articles/illumina-patterned-flow-cells-generate-duplicated-sequences/. Accessed 06 Jan 2021.
40. Sohn JI, Nam JW. The present and future of de novo whole-genome assembly. Brief Bioinform. 2018;19(1):23–40.
41. Leibiger C, Kosyakova N, Mkrtchyan H, Glei M, Trifonov V, Liehr T. First molecular cytogenetic high resolution characterization of the NIH 3T3 cell line by murine multicolor banding. J Histochem Cytochem. 2013;61(4):306–12.
42. Liao Y, Shi W. Read trimming is not required for mapping and quantification of RNA-seq reads. bioRxiv. 2019. https://doi.org/10.1101/833962.
43. Sturm M, Schroeder C, Bauer P. SeqPurge: highly-sensitive adapter trimming for paired-end NGS data. BMC Bioinformatics. 2016;17:208.
44. Love MI, Hogenesch JB, Irizarry RA. Modeling of RNA-seq fragment sequence bias reduces systematic errors in transcript abundance estimation. Nat Biotechnol. 2016;34(12):1287–91.
45. Schmeing S. ReSeq. GitHub. https://github.com/schmeing/ReSeq. Accessed 06 Jan 2021.
46. Jiang H, Lei R, Ding S-W, Zhu S. Skewer: a fast and accurate adapter trimmer for next-generation sequencing paired-end reads. BMC Bioinformatics. 2014;15:182.
47. Reinert K, Dadi TH, Ehrhardt M, Hauswedell H, Mehringer S, Rahn R, Kim J, Pockrandt C, Winkler J, Siragusa E, Urgese G, Weese D. The seqan c++ template library for efficient sequence analysis: a resource for programmers. J Biotechnol. 2017;261:157–68.
48. Fienberg SE. An iterative procedure for estimation in contingency tables. Ann Math Stat. 1970;41(3):907–17.
49. Lewis PM. Approximating probability distributions to reduce storage requirements. Inf Control. 1959;2(3):214–25.
50. Fienberg SE. The analysis of multidimensional contingency tables. Ecology. 1970;51(3):419–33.
51. Benjamini Y, Hochberg Y. Controlling the false discovery rate: a practical and powerful approach to multiple testing. J R Stat Soc Ser B (Methodol). 1995;57(1):289–300.
52. Nocedal J. Updating quasi-Newton matrices with limited storage. Math Comput. 1980;35(151):773–82.
53. Liu DC, Nocedal J. On the limited memory bfgs method for large scale optimization. Math Program. 1989;45:503–28.
54. Johnson SG. The NLopt nonlinear-optimization package. https://github.com/stevengj/nlopt. Accessed 06 Jan 2021.
55. Köster J, Rahmann S. Snakemake–a scalable bioinformatics workflow engine. Bioinformatics. 2012;28(19):2520–2.
56. Schmeing S. ReSeq-paper. GitHub. https://github.com/schmeing/ReSeq-paper. Accessed 06 Jan 2021.
57. Li H. A statistical framework for SNP calling, mutation discovery, association mapping and population genetical parameter estimation from sequencing data. Bioinformatics. 2011;27(21):2987–93.
58. Quinlan AR, Hall IM. BEDTools: a flexible suite of utilities for comparing genomic features. Bioinformatics. 2010;26(6):841–2.
59. H. L. Aligning sequence reads, clone sequences and assembly contigs with BWA-MEM. arXiv 1303.3997v2. 2013.

60. Heng L, Yu C, Li Y, Lam TW, Yiu SM, Kristiansen K, Wang J. SOAP2: an improved ultrafast tool for short read alignment. Bioinformatics. 2009;25(15):1966–7.
61. Walker BJ, Abeel T, Shea T, Priest M, Abouelliel A, Sakthikumar S, Cuomo CA, Zeng Q, Wortman J, Young SK, Earl AM. Pilon: an integrated tool for comprehensive microbial variant detection and genome assembly improvement. PLoS ONE. 2014;9(11):112963.
62. Langmead B, Salzberg SL. Fast gapped-read alignment with Bowtie 2. Nat Methods. 2012;9(4):357–9.
63. Garrison E, Marth G. Haplotype-based variant detection from short-read sequencing. arXiv 1207.3907. 2012.
64. Li H, Handsaker B, Wysoker A, Fennell T, Ruan J, Homer N, Marth G, Abecasis G, Durbin R. The Sequence Alignment/Map format and SAMtools. Bioinformatics. 2009;25(16):2078–9.
65. Marçais G, Kingsford C. A fast, lock-free approach for efficient parallel counting of occurrences of k-mers. Bioinformatics. 2011;27(6):764–70.
66. Kokot M, Dlugosz M, Deorowicz S. KMC 3: counting and manipulating k-mer statistics. Bioinformatics. 2017;33(17):2759–61.
67. Simpson JT. Exploring genome characteristics and sequence quality without a reference. Bioinformatics. 2014;30(9):1228–35.
68. Simpson JT, Durbin R. Efficient de novo assembly of large genomes using compressed data structures. Genome Res. 2012;22(3):549–56.
69. Schmeing S. ReSeq simulates realistic illumina high-throughput sequencing data. Zenodo. 2021. https://doi.org/10.5281/zenodo.4420862.

Publisher's Note