CROMqs: an infinitesimal successive refinement lossy compressor for the quality scores

I. Ochoa, A. No, M. Hernaez, and T. Weissman
Department of Electrical Engineering
Stanford University, Stanford CA 94305
Email: {iochoa, albertno, mhernaez, tsachy}@stanford.edu

***Abstract*—Massive amounts of sequencing data are being generated thanks to advances in sequencing technology and a dramatic drop in the sequencing cost. Much of the data are comprised of nucleotides and the corresponding quality scores that indicate their reliability. The latter are more difficult to compress and are themselves noisy. As a result, lossy compression of the quality scores has recently been proposed to alleviate the storage costs. Further, it has been shown that lossy compression, at some specific rates, can achieve a performance on variant calling similar to that achieved with the lossless compressed data.**

We propose CROMqs, a new lossy compressor for the quality scores with the property of "infinitesimal successive refinability". This property allows the decoder to decompress the data iteratively without the need of agreeing with the encoder on a specific rate prior to compression. This characteristic is particularly amenable in practice, as in most cases the appropriate rate at which the lossy compressor should operate can not be established prior to compression. Further, this property can be of interest in scenarios involving streaming of genomic data. CROMqs is the first infinitesimal successive refinement lossy compressor for the quality scores in the literature, and we show that it obtains a comparable rate-distortion performance to previously proposed algorithms. Moreover, we also show that CROMqs achieves a comparable performance on variant calling to that of the lossless compressed data.

I. Introduction

Recent advances in Next Generation high-throughput Sequencing (NGS) [1] have revolutionized biomedical sciences, marking the beginning of a new era for biological research. It is now possible to identify genomic changes that predispose individuals to debilitating diseases or make them more responsive to certain therapies and emerging treatments [2]. Timely discovery and knowledge mining in this area of biological research is largely enabled by massive raw data sets of NGS machines. Moreover, the decrease in the sequencing costs is expected to drastically increase the volume of genomic data to be stored and processed in the near future [3].

In this paper we focus on the raw NGS data, which is stored in the widely accepted FASTQ format. It consists of both the nucleotide sequences (called "reads") and per-base quality scores that indicate the level of confidence in the readout of these sequences. The latter are generally represented in the *Phred scale*, given by $Q = \lceil -10 \log_{10} P \rceil$, with P being the probability that the corresponding nucleotide is in error.[1]

Both short-term managing and long-term storage of this data represent a huge burden as scientists have to expand their data storage solutions, which are costly and space demanding. As such, recent research has focused on read and quality score compression, to allow for more efficient storage and fast exchange of this data.

Quality scores have proven to be more difficult to compress than the reads, due in part to their higher variance and larger alphabet. For example, as shown in [4], losslessly compressed quality scores may occupy up to 6 times more than the compressed reads. Moreover, there is evidence that quality scores are corrupted by some amount of noise introduced during sequencing, mainly due to the use of inaccurate models to estimate the probabilities of error [5]. Thus, whereas lossless compression is preferred for the reads, lossy compression of quality scores has emerged as a natural candidate to boost compression performance (see [6]–[8] and references therein).

Traditionally, lossy compressors have been analyzed in terms of their rate-distortion performance. Such analysis provides a yardstick for comparison of lossy compressors of quality scores that is oblivious to the multitude of downstream applications. However, the data under consideration is used for biological inference, and thus it is important to also analyze the effect on the subsequent analysis. Variant calling, which is used for medical decision making, is one of the most important analysis performed in practice. Variant calling seeks for the existing variants (differences) between the sequenced genome and that of a reference genome. A methodology for analyzing the effect of lossy compressors of quality scores on variant calling was presented in [9]. Note that extensive studies, like the one provided in [9], demonstrate that lossy compression of the quality scores can lead to comparable performance – or even superior – to that of the lossless compressed data in some cases (e.g., for some rates).

These results demonstrate that lossy compression operating at the right rate can reduce the size of the genomic files with comparable performance to that of the lossless case. However, we may not be able to establish the appropriate rate at the lossy compressor prior to the analysis. In addition, current lossy compressors for the quality scores require the user to specify the desired rate prior to compression. Unfortunately, this means that in order to find the most adequate rate, one should compress the data several times at different rates, perform the analysis on each of the reconstructed files, and

[1] In the FASTQ file they are stored as an ASCII character ranging (generally) from 33 to 73, which corresponds to the *Phred+33* scale.

978-1-5090-1090-5/16/$31.00 ©2016 IEEE

based on the results, decide on the most appropriate rate.

As the data grows, this process becomes extremely inefficient. To partially alleviate this issue, in this paper we propose CROMqs, a new lossy compressor for the quality scores. CROMqs allows the user to compress and decompress the data only once, without the need of knowing the rate ahead of time, while being able to analyze the effect of the lossy compressor at several rates. Specifically, we propose a lossy compressor with the property of *infinitesimal successive refinability*,[2] based on the work presented in [10]. Informally, this means that the encoder of the lossy compressor can compress the data at a high rate (only once), while allowing the decoder to decompress the data iteratively, reducing the distortion (equivalently increasing the rate) at every step. Thus the decoder can generate a reconstructed set of quality scores, with decreased distortion, at every decompression step.

To the best knowledge of the authors, this is the first lossy compressor for quality scores with this property. Moreover, we show that the proposed lossy compressor CROMqs achieves a rate-distortion performance comparable to or better than the previously proposed algorithms, while obtaining a variant calling performance comparable to that achieved with the losslessly compressed data. Specifically, we consider mean squared error distortion for the rate-distortion analysis, and the methodology and data proposed in [9] for the variant calling analysis, which comprises several variant calling tools and datasets.

II. Problem description

We consider the lossy compression of N quality score sequences of length n, presented for example in a FASTQ file. We denote the set of quality score sequences as $\boldsymbol{Q} = \{\boldsymbol{Q}_i\}_{i=1}^N$, with $\boldsymbol{Q}_i = [Q_i(1), Q_i(2), \ldots, Q_i(n)]$. We further denote the reconstructed quality scores by $\widehat{\boldsymbol{Q}} = \{\widehat{\boldsymbol{Q}}_i\}_{i=1}^N$, with $\widehat{\boldsymbol{Q}}_i = [\widehat{Q}_i(1), \widehat{Q}_i(2), \ldots \widehat{Q}_i(n)]$.

We are interested in designing *an infinitesimal successive refinement* lossy compressor for the quality scores. This implies that the decoder can reconstruct the quality score sequences with only partial messages. More precisely, assume the encoder compresses the quality scores with rate R, thus producing an encoded sequence of nNR bits. Then, for any first fraction $\nu \in (0,1)$ of the encoded bits, the decoder can reconstruct a set of quality scores that achieve the same distortion as if the encoder had compressed the quality scores at rate νR. Thus, the encoder and the decoder do not have to agree on the rate prior to compression.

We analyze the proposed algorithm in terms of its rate-distortion performance. In particular, we measure the overall distortion D under Mean Square Error (MSE). That is,

$$\begin{aligned} D &= \frac{1}{N}\sum_{i=1}^{N} D(i) \\ &= \frac{1}{N}\sum_{i=1}^{N}\frac{1}{n}\sum_{j=1}^{n}(Q_i(j) - \widehat{Q}_i(j))^2. \end{aligned}$$

Our goal is to design a lossy compressor that minimizes the MSE between the original quality scores and its reconstructions at any given rate. In addition, we are interested in analyzing if the variant calling performance obtained with the reconstructed quality scores is comparable to that obtained with the lossless compressed ones. To perform this analysis, we use the methodology and data presented in [9], which will be introduced in Section IV.

III. Proposed Scheme

In this section we describe CROMqs, the proposed lossy compressor for the quality scores. CROMqs is based on CROM, the infinitesimal successive refinement lossy compressor introduced in [10]. We first review CROM, and then introduce our scheme.

A. CROM – Coding with Random Orthogonal Matrices

CROM is an iterative lossy compressor, which achieves the rate-distortion function of a Gaussian source under MSE for any stationary ergodic source. The main steps of CROM for compressing a source sequence X^n are described in Algorithm 1. It uses orthogonal matrices $\{A_i\}_{i=1}^{L_n}$ and scalars $\{\alpha_i\}_{i=1}^{L_n}$, with L_n being the total number of iterations.

Algorithm 1 CROM

```
Set $\mathbf{X}^{(1)} = A_1 X^n$.
for $i = 1$ to $L_n$ do
    Let $m_i = \arg\max \mathbf{X}^{(\mathbf{i})}$.
    Let $\mathbf{U}^{(\mathbf{i})} = (U_1^{(i)}, U_2^{(i)}, \cdots, U_n^{(i)})$ where
        $U_j^{(i)} = \begin{cases} \sqrt{\frac{n-1}{n}} & \text{if } j = m_i \\ -\sqrt{\frac{1}{n(n-1)}} & \text{otherwise.} \end{cases}$   (1)
    Let $\mathbf{X}^{(\mathbf{i+1})} = A_{i+1}(\mathbf{X}^{(\mathbf{i})} - \alpha_i \mathbf{U}^{(\mathbf{i})})$.
end for
Send $(m_1, m_2, \ldots m_{L_n})$.
```

At each iteration, the encoder performs the following operations. First, it finds the index m_i of the maximum element in $\mathbf{X}^{(\mathbf{i})}$, which requires a description of only $\log n$ bits. Then, the encoder constructs the vector $\mathbf{U}^{(\mathbf{i})}$, which is roughly 1 at the position of the maximum element (i.e., m_i), and roughly zero at all other positions. Then, it subtracts $\mathbf{X}^{(\mathbf{i})}$ and $\mathbf{U}^{(\mathbf{i})}$ with appropriate scaling. The final step is to multiply by the orthogonal matrix A_{i+1}. Informally, orthogonal matrices mix the source so that the output has similar statistical properties to Gaussian random vectors. This permits to keep applying the

[2] This property will be formally defined in Section III-B.

scheme iteratively. Based on the extreme value theory [11], the decoder can guess the value of the maximum element closely, and thus it suffices to describe only the index of the maximum element. The decoding can be done in a reverse manner. The encoder and the decoder share the matrices A_i's and scalars α_i's, so it can recover $\mathbf{X^{(i)}} = A_{i+1}^T \mathbf{X^{(i+1)}} + \alpha_i \mathbf{U^{(i)}}$.

Note that at each iteration the encoder stores the index of the maximum element, which requires $\log n$ bits. Thus if the encoder performs L iterations, the corresponding rate is $R = \frac{L \log n}{n}$ bits per symbol. At this rate, it achieves distortion $D_G\left(\frac{L \log n}{n}\right)$, where D_G is the distortion-rate function of a memoryless Gaussian source. If the decoder only has messages from the first $L'(< L)$ iterations, it can achieve $D_G\left(\frac{L' \log n}{n}\right)$. Recall that this is the distortion that CROM would achieve with rate $R' = \frac{L' \log n}{n}$. Since the messages from the first L' iterations take an effective rate of $R' = \frac{L' \log n}{n}$, we can conclude that CROM has the desired property of *infinitesimal successive refinability*.

This scheme achieves the rate distortion function of memoryless Gaussian sources. That is, CROM is optimal when the source is memoryless Gaussian. In addition, it is computationally efficient compared to other optimal schemes.

B. Proposed lossy compressor (CROMqs)

In this section we describe CROMqs, which lossily compresses quality scores. To the best of our knowledge, there are no known statistics of the quality score sequences. Thus, under the lack of known statistics, as argued in [6], it is natural to assume that the quality score sequences are identically and independently distributed (i.i.d.) as a multivariate Gaussian with mean $\boldsymbol{\mu_Q}$ and covariance matrix $\Sigma_{\boldsymbol{Q}}$. This is justified by the fact that, given a vector source with a particular covariance structure, the Gaussian multivariate source is the least compressible. Further, a code designed under the Gaussian assumption will perform at least as well on any other source of the same covariance [12]. Note that due to the high correlation between the quality scores, the covariance matrix $\Sigma_{\boldsymbol{Q}}$ is not diagonal in general. Given this assumption, the main steps of CROMqs are the following:

- Extract a set of subsequences, distributed as i.i.d. Gaussian, using the Singular Value Decomposition (SVD).
- Compute the appropriate rate for each subsequence using a water-filling technique.
- Compress each subsequence with CROM, with the previously computed rate.

Next we describe each of these steps in more detail.

First, we perform the following operation on each of the quality score sequences: $\boldsymbol{Q}_i' = V^T(\boldsymbol{Q}_i - \boldsymbol{\mu_Q})$, where V is extracted from the Singular Value Decomposition (SVD) $\Sigma_{\boldsymbol{Q}} = VSV^T$.[3] As a result of this operation, the new set of sequences $\{\boldsymbol{Q}_i'\}_{i=1}^N$ are i.i.d. distributed as $\mathcal{N}(0, S)$, where S is the diagonal matrix containing the singular values of $\Sigma_{\boldsymbol{Q}}$. We denote $S = \mathrm{diag}(\sigma_1^2, \ldots, \sigma_n^2)$. That is, for any position $l \in \{1, \ldots, n\}$, a subsequence $\{Q_i'(l)\}_{i=1}^N$ represents a sequence of i.i.d. symbols drawn from a $\mathcal{N}(0, \sigma_l^2)$ distribution. With this transformation, the problem is of compressing n Gaussian source sequences of length N, each with possibly different variances.

Since each subsequence is i.i.d. Gaussian with possibly a different variance, we need to assign the appropriate rate for each subsequence, such that the obtained distortion is the same in each of them. To find the appropriate rates we use a water-filling technique. In theory, CROM achieves the Gaussian distortion rate function $D(R) = \sigma_l^2 e^{-2R}$ as the block length N goes to infinity. However, we cannot achieve the theoretical optimum distortion with a finite block length N. Moreover, for computational reasons (which will become apparent below), we split each subsequence in blocks of length $N' = 65,536$. For this block length, CROM achieves roughly a distortion function given by $D(R) = \sigma_l^2 e^{-1.4R}$. Thus, we choose the rate R_l for each subsequence such that the associated distortions $\sigma_l^2 e^{-1.4R_l}$ are all equal. The number of iterations of CROM for each subsequence is given by $L_l = \frac{N' R_l}{\log N'}$.

After this computation, we are ready to apply CROM with a large enough rate on each subsequence $\{Q_i'(l)\}_{i=1}^N$, for all $1 \leq l \leq n$. However, recall that CROM requires uniformly randomly generated orthogonal matrices $\{A_i\}_{i=1}^{L^n}$ to guarantee the best performance with high probability. Moreover, uniformly drawn matrices are dense with high probability, and this becomes a major bottleneck for the implementation of CROM. The reason is that dense matrices are hard to store, and multiplication between dense matrices is computationally very expensive. Thus, instead of uniform random orthogonal matrices, we use structured orthogonal matrices as described in [10, Section V]. Recall that we compress subsequences of length $N' = 65,536$ at each time, and thus the constructed matrices are of size $65,536 \times 65,536$. Matrices are constructed in a systematic manner, so that it is enough to store only $65,536 \times 8$ real numbers to describe each matrix. In addition, the constructed matrices are sparse, which means that matrix multiplication can be performed much faster.

In theory, CROM should use a different orthogonal matrix at each iteration. However, if we construct an enough number of matrices which are sufficiently diverse, it can be shown that the performance of CROM is not degraded by much. Thus we generated 50 matrices, and used them iteratively. That is, the first matrix is used again in the 51st iteration, and so on. Note that we can use the same set of matrices for each subsequence, since we are compressing each subsequence independently.

Once the encoding part of the lossy compressor is finished, the decoder can sequentially reconstruct different sets of sequences $\{\widehat{\boldsymbol{Q}}_i'\}_{i=1}^N$, with decreasing distortion, thanks to the infinitesimal successive refinability. Finally, note that the decoder needs to further perform the following operation to reconstruct the quality scores:

$$\widehat{\boldsymbol{Q}}_i = \lceil V\widehat{\boldsymbol{Q}}_i' + \boldsymbol{\mu_Q} \rceil, \text{ for } 1 \leq i \leq N. \qquad (2)$$

[3] Note that this transformation is analogous to the Karhunen–Loève transform.

IV. Results

To assess the performance of CROMqs, we perform two distinct analysis. In particular, we consider the rate-distortion performance and the effect of the proposed lossy compressor on variant calling. We use the same datasets for both analysis, which we introduce next. The remaining of the section corresponds to the results of each of the performed analysis.

A. Datasets

We use data from the *H. Sapiens* individual *NA12878*, as suggested in [9]. In particular, we consider the datasets ERR174324 and ERR262997, which correspond to a 15×-coverage pair-end WGS (Whole Genome Sequenced) dataset and a 30×-coverage pair-end WGS dataset, respectively. For each of them we extracted the chromosome 20. The number of reads is approximately 9 millions for the 15× dataset and 20 millions for the 30× dataset, and in both files the read length is 101.

B. Rate-distortion performance

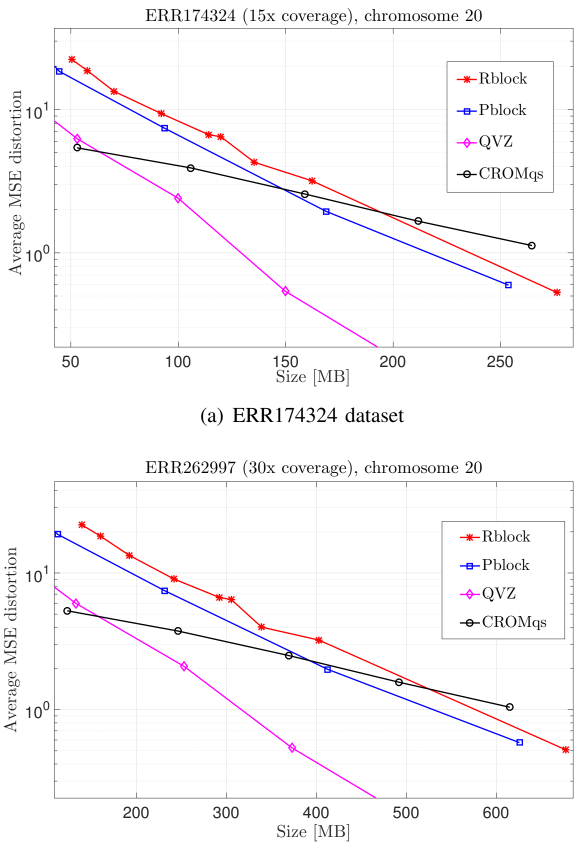
As stated above, we focus on the rate-distortion performance for MSE distortion. We compare the performance of the proposed algorithm CROMqs with that of the state-of-the-art lossy compressors for quality scores. In particular, we consider the lossy compressors Rblock and Pblock, presented in [7], as well as QVZ [8].

Figure 1 shows the results on rate-distortion for the two considered datasets, as a function of the size (in MB) versus the average MSE distortion. For reference, lossless compression provides a total size of 250 MB (ERR174324) and 600 MB (ERR262997). As it can be observed, the proposed algorithm CROMqs offers a comparable performance to the previously proposed algorithms in both analyzed datasets. In particular, for the smallest rate, we obtain the least distortion among the considered algorithms. For small rates, the performance is better than that of Rblock and Pblock, although QVZ obtains the best performance. For higher rates, CROMqs offers a performance comparable to that of Rblock and Pblock. The reason why CROMqs does not achieve the smallest distortion for very high rates is due to the transformation applied to the original quality scores. As a result of this transformation, CROMqs is not able to achieve zero distortion. That is, lossless compression is not achievable by CROMqs.

In summary, CROMqs shows a comparable performance in terms of rate-distortion to that of the state-of-the-art algorithms, while being the only lossy compressor for the quality scores providing the property of infinitesimal successive refinability.

C. Effect on variant calling

Next we show the effect that applying the proposed lossy compressor CROMqs in the quality scores has on variant calling. Recall that variant calling is the process of finding the existing variants (differences) between a sequenced genome and that of a reference genome. As stated above, we follow the methodology proposed in [9] for the analysis. More precisely, it considers the three most used variant calling pipelines in practice, that is, GATK [13], Samtools [14], and Platypus [15]. Thus we analyze the effect that replacing the quality scores with those reconstructed by CROMqs has on the output of these pipelines.



(a) ERR174324 dataset

(b) ERR262997 dataset

Fig. 1: Rate-distortion performance of the proposed lossy compressor CROMqs and the previously proposed algorithms Rblock, Pblock and QVZ, for the chromosomes 20 extracted from a) the ERR174324 dataset (15×-coverage) and b) the ERR262997 dataset (30×-coverage)

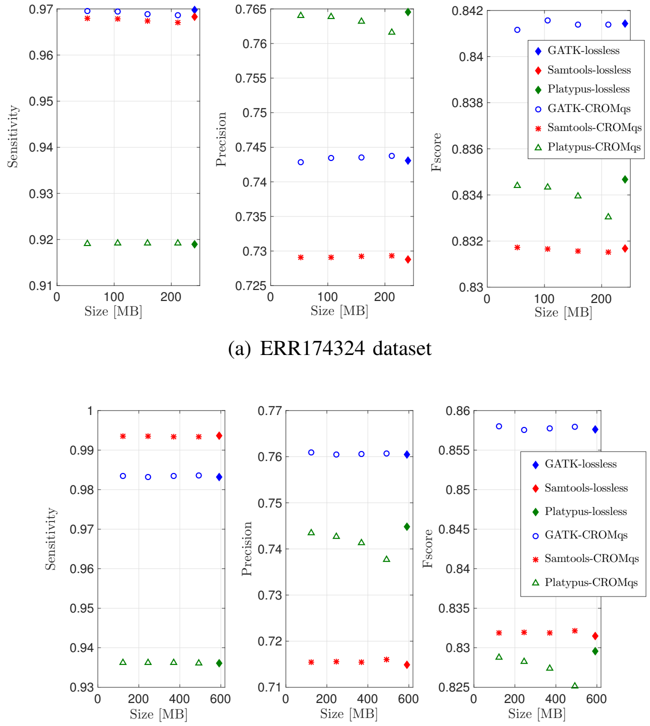
In particular, we are interested in analyzing the "correctness" of the variants found by each of the pipelines. For that, and as suggested in [9], we use the set of "true" variants provided by the Genome in a Bottle consortium (GIAB) [16], which are assumed to be correct for the data used in our study. This set of true variants allow us to compute the number of True Positives (T.P.); the set of found variants that are contained in the true set; the number of False Positives (F.P.), the set of found variants that are not contained in the true set; and the number of False Negatives (F.N.), the variants contained in the true set that were not found. We would like to obtain a large number of T.P., and a small number of F.P. and F.N. For the analysis, we then use the following metrics:

- Sensitivity: proportion of variants in the true set that are present in the VCF file, that is, T.P./(T.P. + F.N.).

- Precision: proportion of variants in the VCF file that are present in the true set, that is, T.P./(T.P. + F.P.).
- F-Score: harmonic mean of the sensitivity and the precision.

Note that we are interested in obtaining high values for all three metrics.

The results on variant calling when using CROMqs for lossily compressing the quality scores are shown in Figure 2. As it can be observed, applying CROMqs to the quality scores produces a small variability in the results (for both datasets), as compared to the lossless compressed data. However, this variability is much smaller than the variability that exist between the performance of the different variant calling pipelines. In addition, in some cases applying CROMqs produces a set of variants that increases the sensitivity, precision, and fscore simultaneously. This suggests that CROMqs could be used in practice to boost the compression performance without degrading the variant calling performance.



(a) ERR174324 dataset

(b) ERR262997 dataset

Fig. 2: Effect of CROMqs on variant calling, for the different tools GATK, Samtools, and Platypus, on the chromosome 20 of (a) the ERR174324 dataset (15×-coverage) and (b) the ERR262997 dataset (30×-coverage)

V. Conclusion

We proposed CROMqs, a new *infinitesimal successive refinement* lossy compressor for the quality scores. This property allows the decoder to reconstruct the quality scores iteratively, minimizing the distortion at each step, without the need of specifying the desired rate prior to compression. This is specially useful in practice, as in most cases the appropriate rate at which the lossy compressor should operate can not be established a priori. CROMqs is the first lossy compressor for quality scores with this property.

In addition, experimentation on real data shows that CROMqs achieves a rate-distortion performance comparable – and sometimes superior – to that of the previously proposed algorithms. Further, we analyzed the effect that applying CROMqs has on the subsequent analysis performed on the data. Specifically, we analyzed the effect on variant calling, one of the most important applications in practice, and showed that using CROMqs produces a set of variants that is very closed to that obtained with the lossless compressed data.

Acknowledgment

This work was partially supported by the Center for Science of Information (CSoI), a fellowship from the Basque Government, and an NIH grant with number 1 U01 CA198943-01.

References

[1] M. L. Metzker, “Sequencing technologies - the next generation,” *Nature Reviews Genetics*, vol. 11, no. 1, pp. 31–46, 2009.

[2] J. S. Berg, M. J. Khoury, and J. P. Evans, “Deploying whole genome sequencing in clinical practice and public health: meeting the challenge one bin at a time,” *Genetics in Medicine*, vol. 13, no. 6, 2011.

[3] W. KA, “DNA Sequencing Costs: Data from the NHGRI Genome Sequencing Program,” www.genome.gov/sequencingcosts, 2008.

[4] J. K. Bonfield and M. V. Mahoney, “Compression of fastq and sam format sequencing data,” *Plos ONE*, 2013.

[5] S. Das and H. Vikalo, “Onlinecall: fast online parameter estimation and base calling for illumina’s next-generation sequencing,” *Bioinformatics*, vol. 28, no. 13, pp. 1677–1683, 2012.

[6] I. Ochoa, H. Asnani, D. Bharadia, M. Chowdhury, T. Weissman, and G. Yona, “Qualcomp: a new lossy compressor for quality scores based on rate distortion theory,” *BMC bioinformatics*, vol. 14, no. 1, 2013.

[7] R. Cánovas, A. Moffat, and A. Turpin, “Lossy compression of quality scores in genomic data,” *Bioinformatics*, 2014.

[8] G. Malysa, M. Hernaez, I. Ochoa, M. Rao, K. Ganesan, and T. Weissman, “Qvz: lossy compression of quality values,” *Bioinformatics*, p. btv330, 2015.

[9] I. Ochoa, M. Hernaez, R. Goldfeder, T. Weissman, and E. Ashley, “Effect of lossy compression of quality scores on variant calling,” *Briefings in bioinformatics*, p. bbw011, 2016.

[10] A. No and T. Weissman, “Rateless lossy compression via the extremes,” *arXiv preprint arXiv:1406.6730*, 2014.

[11] B. Gnedenko, “Sur la distribution limite du terme maximum d’une serie aleatoire,” *Annals of mathematics*, pp. 423–453, 1943.

[12] A. Lapidoth, “On the role of mismatch in rate distortion theory,” *IEEE Trans. Inf. Theory*, vol. 43, no. 1, pp. 38–47, 1997.

[13] G. A. Auwera, M. O. Carneiro, C. Hartl *et al.*, “From fastq data to high-confidence variant calls: the genome analysis toolkit best practices pipeline,” *Current Protocols in Bioinformatics*, pp. 11–10, 2013.

[14] H. Li, B. Handsaker *et al.*, “The sequence alignment/map format and samtools,” *Bioinformatics*, vol. 25, no. 16, pp. 2078–2079, 2009.

[15] A. Rimmer, H. Phan *et al.*, “Integrating mapping-, assembly-and haplotype-based approaches for calling variants in clinical sequencing applications,” *Nature genetics*, vol. 46, no. 8, pp. 912–918, 2014.

[16] J. M. Zook, B. Chapman, J. Wang *et al.*, “Integrating human sequence data sets provides a resource of benchmark snp and indel genotype calls,” *Nature biotechnology*, vol. 32, no. 3, pp. 246–251, 2014.