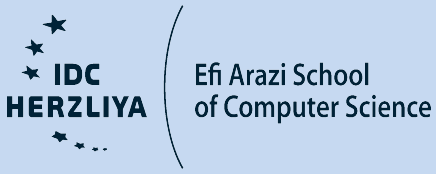


The Interdisciplinary Center, Herzliya
Efi Arazi School of Computer Science
M.Sc. program - Research Track

Applications of statistical and ML methods in molecular biology including synthetic DNA QC

by
Yoav Orlev

M.Sc. dissertation, submitted in partial fulfillment of the requirements for the M.Sc. degree, research track, School of Computer Science
The Interdisciplinary Center, Herzliya

This work was carried out under the supervision of Prof. **Zohar Yakhini** from the Efi Arazi School of Computer Science, The Interdisciplinary Center, Herzliya.

Applications of statistical and ML methods in Molecular Biology, including synthetic DNA QC

Yoav Orlev, Zohar Yakhini

May 2021

Abstract

SOLQC - Recent years have seen a growing number and a broadening scope of studies using synthetic oligo libraries for a range of applications in synthetic biology. As experiments are growing by numbers and complexity, analysis tools can facilitate quality control and help statistical assessment and inference. We present a novel analysis tool, called SOLQC, which enables fast and comprehensive analysis of synthetic oligo libraries. SOLQC takes as input the results of an NGS analysis performed on the library, by the user. SOLQC then provides statistical information such as the distribution of variant representation, different error rates and their dependence on sequence or library properties. SOLQC also produces graphical descriptions of the analysis results. The results are reported in a flexible report format. We demonstrate SOLQC by analyzing several literature libraries. In the context of these analyses we discuss the potential benefits and relevance of the different components of SOLQC.

Intrinsic Autoencoders - An important tools that serves life science to gain insights into the functionality of living cells is the analysis of gene expression in cells and populations. The results of gene expression profiling usually reside in a very high dimensional space, obstructing efficient and effective inference and down stream tasks such as classification. We believe that working in a

more adequate representation space can facilitate better results for downstream analysis tasks such as classification and clustering. As a step in this direction we investigate the intrinsic dimension of simple data. In particular, we show how autoencoders can be used to fully reconstruct data that resides in a manifold whose dimension is much smaller than that of the original ambient representation space.

Contents

1 SOLQC

1.1 Motivation and Background

Recent years have seen a growing number and a broadening scope of studies using synthetic oligo libraries for a range of applications in synthetic biology. As experiments are growing by numbers and complexity, analysis tools can facilitate quality control and help in assessment and inference. Background on DNA synthesis and on relevant literature can be found in our paper ([1] and appendix A) as well as [2] and [3].

1.2 Results

We present a novel analysis tool, called SOLQC, which enables fast and comprehensive analysis of synthetic oligo libraries, based on NGS analysis performed by the user. SOLQC provides statistical information such as the distribution of variant representation, different error rates and their dependence on sequence or library properties. SOLQC produces graphical descriptions of the analysis results. The results are reported in a flexible report format. We demonstrate SOLQC by analyzing literature libraries. We also discuss the potential benefits and relevance of the different components of the analysis.

1.3 Paper

Published in Bioinformatics, https://doi.org/10.1093/bioinformatics/btaa740. Also see full version in the appendix. [1]

1.4 Future Directions

1. In it's current state, running the tool with high volume NGS results (over 100M reads) can take several hours, which is unacceptably long. This is mainly due to the matching process.

A more efficient matching algorithm, designed for the specific task of aligning to a set of short variants/sequences, will significantly reduce this time consuming part of the process.

2. Application of the tool to more datasets.

3. Improved visualization and statistical analysis.

1.5 Appendix

See SOLQC full version below.

Bioinformatics
doi.10.1093/bioinformatics/xxxxxx
Advance Access Publication Date: Day Month Year
Manuscript Category

OXFORD

Subject Section

SOLQC : Synthetic Oligo Library Quality Control Tool.

Omer Sabary [*1], Yoav Orlev [*2], Roy Shafir [1,2], Leon Anavy [1], Eitan Yaakobi [1] and Zohar Yakhini [1,2]

[1]Computer Science Department, Technion, Haifa, 3200003, Israel.
[2]School of Computer Science, Herzliya Interdisciplinary Center, Herzliya, 4610101, Israel.
[*]The two first authors contributed equally to this work.

*To whom correspondence should be addressed.

Associate Editor: XXXXXXX

Received on XXXXX; revised on XXXXX; accepted on XXXXX

Abstract

Motivation: Recent years have seen a growing number and a broadening scope of studies using synthetic oligo libraries for a range of applications in synthetic biology. As experiments are growing by numbers and complexity, analysis tools can facilitate quality control and help in assessment and inference.
Results: We present a novel analysis tool, called **SOLQC**, which enables fast and comprehensive analysis of synthetic oligo libraries, based on NGS analysis performed by the user. SOLQC provides statistical information such as the distribution of variant representation, different error rates and their dependence on sequence or library properties. SOLQC produces graphical descriptions of the analysis results. The results are reported in a flexible report format. We demonstrate SOLQC by analyzing literature libraries. We also discuss the potential benefits and relevance of the different components of the analysis.
Availability: https://app.gitbook.com/@yoav-orlev/s/solqc/
Contact: omersabary@cs.technion.ac.il

1 Introduction

DNA synthesis technology has greatly developed over recent years and is holding a promise to enable a leap in using natural systems for various applications. For example, synthetic DNA is used for making protein therapeutics and drugs. Another application is in genome editing, wherein optimizing CRISPR-Cas9 systems and reagents is enabled by using libraries of synthetic DNA oligonucleotides. Synthetic DNA is particularly useful for screening of large guide-RNA libraries to optimize CRISPR-Cas9 based systems [19]. The use of synthetic DNA also enables the optimization of crops for efficient biofuel production [24]. In particular, synthetic DNA can be used to perform codon-optimization, directed evolution, enzyme libraries screens, and incorporation of non-natural amino acids to improve novel enzymatic activities in the biofuel industry [8]. Last but not least, synthetic DNA is also an attractive alternative for data storage media, see e.g. [20] and more details in section 1.1. With an information density orders of magnitude better than that of magnetic media and due to its highly robust chemical properties DNA can potentially efficiently store data for centuries. This progress in the use of synthetic DNA, as well as its potential to drive future applications, drives work focused on the optimization of manufacturing processes and of design cycles. A key to such work is monitoring the quality of synthetic DNA throughout the process, including at the hands of the end users.

Synthetic DNA libraries consisting of thousands of DNA sequences, often referred to as **variants**, have become a common tool in molecular biology. They allow for systematic, unbiased investigations for discovery biology, directed evolution for protein engineering, and in vitro molecular optimization to generate mutant proteins with novel properties. Sharon et al. used an oligonucleotide library (OL) to infer gene regulatory logic [23], Levy et al. used an OL to discover a bacterial insulation mechanism [18], and most recently Kotler et al. found links between differential functional impact to mutations in p53 using OLs [16].

The process of using OLs in such studies usually starts with a design file containing the DNA variants, which will be synthesized as millions of physical oligonucleotides (oligos). These oligos will typically be sequenced in one or more steps of the experimental process, producing results in the form of NGS output files (typically fastq). We refer to each

sequenced synthesized strand as a read. To validate and/or optimize the results of the different steps in an OL based study, it is important to quality control all components to make sure that the results stem from the biology and not from noise, from confounding interference of from other biases related to synthesis and to sequencing.

1.1 DNA Storage Systems

The recent progress in synthesis and sequencing technologies has paved the way for the development of a non-volatile data storage technology based upon DNA molecules. A DNA storage system consists of three important components. The first is DNA synthesis. The stage at which the strands or DNA molecules that encode the data are produced (those strands called input strands, or variants). In order to produce strands with acceptable error rates, in a high throughput manner, the length of the strands is typically limited to no more than 250 nucleotides [2]. The second part is a storage container with compartments. This container stores the DNA strands. No order is assumed for this stage. Finally, sequencing is performed to read back a representation of the strands (the output of this stage consists of strands called output strands or sequencing reads). A decoding process transforms the sequencing output back to digital data. The encoding and decoding stages are two processes, external to the storage system, that convert the user's binary data into strands of DNA in such a way that, even in the presence of errors, it will be possible to revert back and reconstruct the original binary data of the user.

One of the first experiments to store information in DNA was conducted by Clellan et al. in 1999, where they coded and recovered a message consisting of 23 characters [7]. Shortly after this, in 2000, Leier et al. have managed to successfully store three sequences of nine bits each [17]. A more significant progress, in terms of the amount of data stored successfully, was reported by Gibson et al. in 2010, demonstrating in-vivo storage of 1,280 characters in a bacterial genome [10]. The first large scale demonstrations of the potential of in vitro DNA storage were reported by Church et al. who recovered 643 KB of data [6] and by Goldman et al. who accomplished the same task for a 739 KB message [11]. Both of these pioneering groups did not recover the entire message successfully as no error correcting codes were used. Later, in [12], Grass et al. stored and recovered successfully 81 KB message, in an encapsulated media, and Bornholt et al. demonstrated storing a 42 KB message [4]. A significant improvement in volume was reported in [3] by Blawat et al. who successfully stored 22 MB of data. Recently, Erlich and Zielinski managed to store 2.11 MB of data with high storage density [9]. The largest volume of stored data is reported by Organick et al. in [20]. Organick et al. describe the encoding and decoding of 200 MB of data, an order of magnitude more data than previously reported. Yazdi et al. [28] developed a method that offers both random access and rewritable storage. Most recently, Anavy et al. [1] described how more data can be stored for less synthesis cycles. Their approach uses composite DNA letters. A similar approach, on a smaller scale, was reported in [5].

1.2 Synthetic Oligo Library (OL) Errors

The processes of synthesizing, storing, sequencing and handling oligonucleotides are all error prone. Each step in the process can independently introduce a significant number of errors:

1. Both the synthesis process and the sequencing process can introduce deletions, insertions, and substitution errors on each of the reads and/or synthesized strands.
2. Current synthesis methods can not generate one copy for each design variant. They all generate thousands to millions of non perfect copies. Each of these copies has a different distribution of errors. Moreover, there might be some variants with a significantly larger number of copies, while some variants may be not be represented at all. In other words, the representation and the error profile of variants in the library is not uniform.
3. The use of DNA for storage or of OLs for other applications typically involves PCR amplification of the strands in the DNA pool [13]. PCR is known to have a preference for some sequences over others, which may further distort the distribution of the number of copies of individual sequences and their error profiles [21, 22].

Most of the research on characterizing errors in synthetic DNA libraries has been done in the context of individual studies using synthetic DNA. Tian et al. showed in [26] that the rate of deletion is $1/100$ per position, insertion is $1/400$ per position, and the rate for substitution is $1/400$. Later, Kosuri and Church [15] noted that column-based oligo synthesis has total error rate of approximately $1/200$ or less for oligos of 200 bases, where the most dominant error is a single base deletion. In addition, they showed that high GC content, at more than 50% of the bases in the strand being G or C, can inhibit the assembly and lead to lost data. They also pointed-out that in OL synthesis, a synthesis method based on DNA microarrays, the error rates are usually higher than those for column-based synthesis. Recently, in [13], Heckel, Mikutis, and Grass, studied the errors in a DNA storage channel based upon three different data sets from the experiments in [9, 11, 13]. In their work they studied the deletion/insertion/substitution rate and how it is affected by filtering reads with incorrect length (compared to the designed length). In particular, when they considered only reads with the correct length, they showed, as expected, that the deletion rate has been significantly decreased in all of the data sets. They also investigated the conditional error probability for substitutions and found out that in [9] the most dominant substitution error was from G to T (20%), and in the rest of the experiments, the most dominant substitution error was from C to G (about 30-40%). They also examined the effect of the number of PCR cycles on the coverage depth, which is the distribution of the number of reads per each of the variants. They concluded that, since the efficiency of the PCR amplification on each of the strands is different, a larger number of PCR amplification cycles leads to a higher differences in the coverage depth distribution of the variants. Organick et al. also characterized the errors in their experiment [20]. First, they found that substitution was the most frequent error in the library, then deletion, and lastly insertion. Furthermore, they found that while deletions showed almost equal rates for all of the four bases, insertions were mostly associated with base G, and substitutions were mostly associated with base T. Lastly, they also examined the read error rates per position. It should be noted that substitution errors are most likely associated with sequencing and not with synthesis.

1.3 This Work

In this paper we describe SOLQC, a software tool that supports the statistical analysis and quality control of OLs. The tool is designed to enable and to facilitate individual labs obtaining information about DNA libraries and performing error analysis before or during experiments. We describe our methods and demonstrate the results of analyzing several libraries from the literature. The dissemination reflects only the authors' view and the EU Commission is not responsible for any use that may be made of the information it contains.

2 Materials and Methods

2.1 SOLQC Tool

In this section we present our software tool, called **SOLQC - Synthetic Oligo Library Quality Control**. This quality control tool generates a customized report which consists of several statistics and plots for a given input synthetic library. Detailed instructions to use the tool are given in Section 6.

The input to the SOLQC tool is the result of a sequencing reaction run on the library. It consists of the design variants and of all the sequenced reads. The input to the tool is provided using the following three files.

1. Design file: This file consists of the design variants that were synthesized and it has to be in a csv format. The tool also supports an IUPAC description [14] of the design.
2. NGS results file: This file is in fastq format and contains the NGS results.
3. Config file: Auxiliary file which consists of other details on the design variants such as information on the barcode etc.

The SOLQC tool is operated in the following order.

1. **Preprocessing**: The reads can be filtered such that only valid reads will be processed by the tool. The selection of valid reads can be configured by the user according to the sequence barcode and its length.

2. **Matching**: Each read is matched to its corresponding variant. The matching step can be done by different strategies as follows.

   - Barcode matching: If the library has a barcode assigned to each variant, the barcode will be used in order to match each read with a tunable tolerance in errors for the matching.
   - Edit distance [25]: The edit distance between an input read and, in principle, all the variants will be calculated, such that the variant with the smallest edit distance will be selected as the matched one.
   - Fast matching: The tool supports also faster matching using several approximations of the edit distance.

   Alternatively, this matching step can be done by the user in advance. In this case the matching between reads and variants is given by fourth input file (in csv format). The set of reads which are matched to the same variant form a *variant cluster*.

3. **Alignment**: Every read is aligned according to its matched variant and an error vector is computed which represents the location and error types at each position of the variant (with insertions handled separately). Fig. 1 demonstrates an example for the alignment step.

4. **Analysis**: The matched reads and their error vectors are used in order to create error characterization and data statistics for the library, as will be described in the sequel.

5. **Report generation**: The output of our tool is a report which consists of analysis results, as selected by the user, in a customizable format.

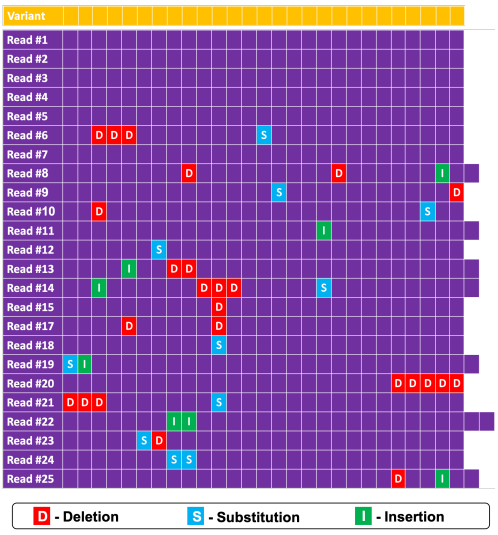


Fig. 1. An example of 25 reads (in purple) aligned to a variant of length 27 (in yellow). For each read, the locations of the deletions, substitutions, insertions are marked in red, blue, green, respectively. This alignment output forms the basis of the analysis performed by SOLQC.

2.2 Statistical QC Analysis for Synthetic DNA Libraries

In this section we describe and discuss the statistical analysis performed and supported by the SOLQC tool. These statistics are explained on actual data from the experiment in [9] by Erlich and Zielinski. The details of this experiment are summarized in Table 1. These statistical results are divided into two parts; The first one addresses the composition of the synthesized library (composition statistics) and the second one addresses the errors inferred from sequencing reads (error statistics). We sampled 1,689,319 reads out of the 15,787,115 reads of the library, and analyzed only reads with length at most 4 bases shorter or longer than the design's length, which is 152 (i.e., their base-length was between 148 and 156). Those reads were matched with their closest design variants using an approximation of the edit distance which calculated the edit distance between all reads and variants based upon the first 80 bases.

Table 1. Experiment by Erlich and Zielinski [9]

Data size	2.11 MB
Design length	152 bases
Number of variants	72000
Number of reads	15,787,115
Number of sampled reads	1,689,315
Number of filtered reads	1,427,781
Synthesis Technology	Twist Bioscience
Sequencing Technology	Ilumina miSeq V4

2.2.1 Composition statistics

1. **Symbol statistics** (Figs. 2 and 3). This plot presents, using a stacked-bar plot, the distribution of all bases in the library by their occurrence at any position both for the reads and for the design variants. This is demonstrated in Fig. 2 for the design variants and in Fig. 3 for the reads.
   - X-axis: The position (index) in the DNA variant or read.
   - Y-axis: The number of occurrences for each base type, scaled for the sequencing depth.
   - Description: In Fig. 2, for every position (index) in the variant, the number of occurrences of each of the four bases in all of the design variants is calculated. Similarly, in Fig. 3, the number in every position is calculated according to the actual reads.

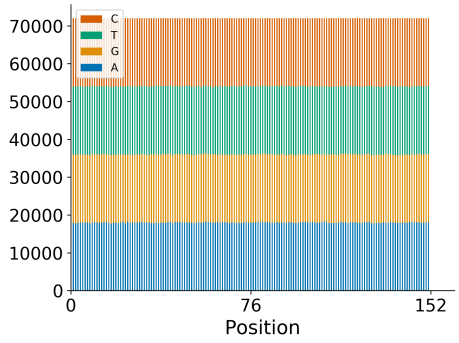


Fig. 2. Base distribution in the design variants (see 2.2.1.1).

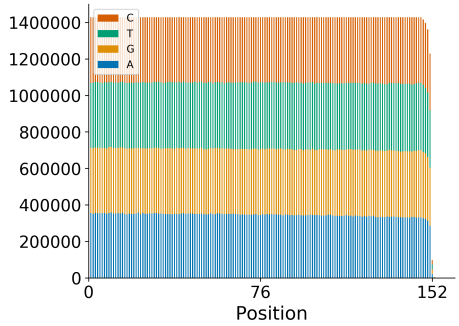


Fig. 3. Base distribution in the reads (see 2.2.1.1).

2. **Histogram of the cluster size per variant** (Figs. 4 and 5). The plot in Fig. 4 presents the histogram of the variant cluster size. That is: the number of filtered reads, per design variant.
   - X-axis: The size of a variant cluster, starting from the size of the smallest variant cluster among all the variants in the library and up to the largest variant cluster value.
   - Y-axis: The number of variants in the library that have a cluster of size x.
   - Description: According to the matching step, the cluster size for each of the design variants is calculated and the histogram is generated by counting the number of variants with a given cluster size. Note that the sum of the y values in this histogram is the number of variants in the experiment, which is 72,000 in [9].

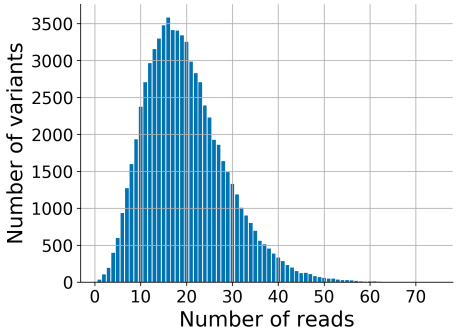


Fig. 4. Histogram of the number of filtered reads per variant (see 2.2.1.2).

This plot can also have a stratified version by 5 ranges of the GC-content of the design variants, as depicted in Fig. 5. To define the 5 values of the GC-content presented in the figure, the tool takes the minimal and maximal values of the GC content as designed in the library and partitions the range between them to 5 different subranges of equal size (in terms of range). The GC-content is presented by percentage.

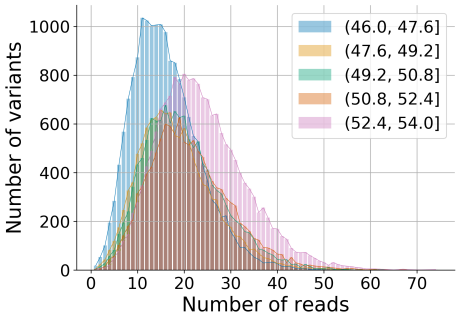


Fig. 5. Histogram of the number of filtered reads per variant, stratified by the GC-content (see 2.2.1.2).

3. **Sorted bar plot of the number of filtered reads per variant** (Figs. 6 and 7). The plot in Fig. 6 presents a sorted bar plot for the variant cluster sizes.
   - X-axis: The variant rank after sorting all variants in the library by their cluster size.
   - Y-axis: The cluster size of variant x.
   - Description: In this plot, after calculating the cluster size for each of the design variants, we sort them in a non-increasing order by the cluster size. Each variant is associated with a bar whose height corresponds to the variant cluster size. Hence, there are 72,000 bars, corresponding to the number of variants in [9].

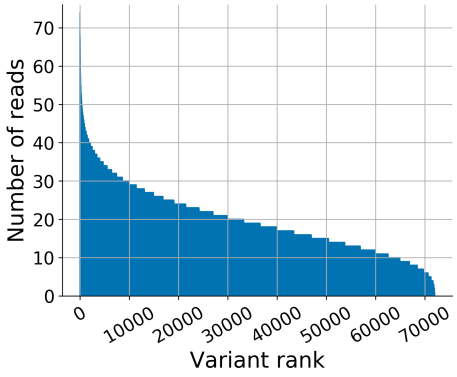


Fig. 6. Sorted bar plot of the number of filtered reads per variant (see 2.2.1.3).

We also plot, as shown in Fig. 7, a stratified version by 5 values of the GC-content of the variants. These 5 values of GC-content were defined by the tool as described in Fig. 5.

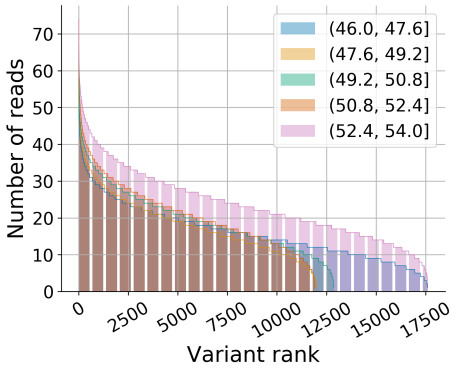


Fig. 7. Sorted bar plot of the number of filtered reads per variant, stratified by the GC-content (see 2.2.1.3).

4. **Histogram of the length of reads** (Fig. 8). This plot presents the distribution of the different lengths of all the reads.
   - X-axis: The length of the read.
   - Y-axis: The number of filtered reads found in the library of length x, presented in log-scale.
   - Description: This plot presents a histogram of the different lengths of all reads in the library.

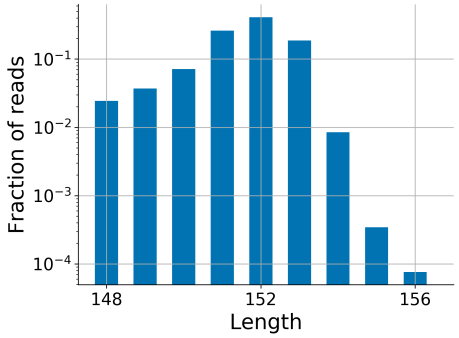


Fig. 8. Histogram of the length of reads (see 2.2.1.4). Note that the difference between 152 and 154 is two orders of magnitude.

2.2.2 Error statistics

1. **Total error rates** (Fig. 9). This plot presents the insertion, substitution, and deletion error rates as inferred from the reads in the library.
   - X-axis: Each bar presents the type of error, which can be one of the following: insertions, substitutions, single-base deletions, long deletions (deletions of more than one base), and total deletions (deletions of one or more bases).
   - Y-axis: The error rate, calculated as the ratio between the total number of errors of each type and the total number number of read bases. The plot is in log scale.
   - Description: After the alignment step, an error vector is calculated for each of the reads based upon its errors with respect to the matched variant. This error vector consists of the locations of the substitutions, insertions, and deletions in the read. See Fig. 1 for an example. For the error rates of insertions, substitutions, and deletions, we plot the ratio between the number of occurrences of each error type (in the entire sequencing data) and the total number of read bases expected in the library (number of filtered reads $\times$ design length[1]. For long deletions, we count each burst of at least two consecutive deletions as a single error, and then plot its ratio with the total number of read bases in the library. Lastly, the error rate of the single-base deletion is calculated is a similar way using the number of bursts of deletions of length 1.

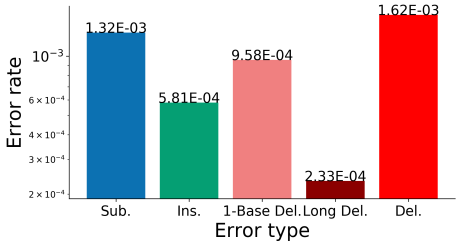


Fig. 9. Total error rates. (see 2.2.2.1)

[1] For example: the deletion rate in Fig. 1 is $24/(25 \times 27)$, which is calculated to be the ratio between the number of red squares (24) and the product of the number of rows (25) with the variant length (27).

2. **Error rate stratified by symbol** (Fig. 10). This plot presents by a heat map the symbol dependent, error distribution. Each square presents for each type of error, its error rate for the specific symbol. For insertions we address both the inserted symbol, and the symbol before the insertion. The x, y entry in the heat map is calculated to be the ratio between the number of type y errors of base x and the expected number of base x in the reads[2].

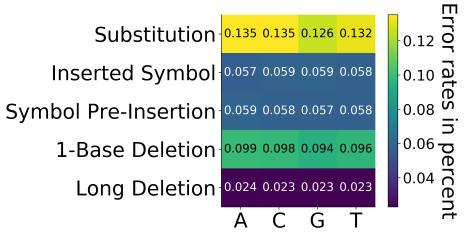


Fig. 10. Error rates stratified by symbol. Note that the numbers are in percents. For example the value of 0.024 for "A" long deletion, means that 0.024 percents of the occurrences of base A in the library creates long deletion error. (see 2.2.2.2)

3. **Error rate per position** (Fig. 11). This plot presents the error rate for every error type as it is reflected in a specific position of the strand.
   - X-axis: The position in the strand, from 5' to 3'; note that the phosphoramidite synthesis direction is 3' to 5'. It is important to emphasize that we report rates as calculated from the alignment results. These rates reflect both synthesis as well as sequencing errors. We expect substitution and insertion errors to be primarily due to sequencing. Long deletions primarily due to synthesis.
   - Y-axis: The error rates per position in all reads for single-base deletions, long deletions, substitutions, and insertions, presented in log scale.
   - Description: For every position between 0 (the first position, from 5' to 3') and 151 (the last position in [9]) and for each error type as described in Fig. 9, the tool calculates the error rate as the ratio between the number of errors of each type and the number of filtered reads.

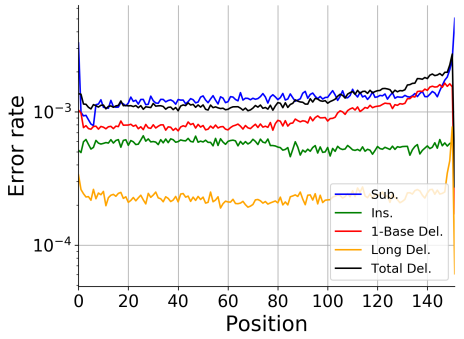


Fig. 11. Error rates by position (see 2.2.2.3). X-axis represents position counted from the 5' end of the designed variant. The Y-axis is log-scale.

4. **Deletion length distribution** (Fig. 12). This plot presents the distribution of the lengths of all deletions.
   - X-axis: Deletion length, which is the number of consecutive deleted bases.
   - Y-axis: The error rate for each length of burst of deletions with exactly x bases divided by the number of total bases in the library.
   - Description: The tool counts the number of deletion bursts of size exactly x bases, based on the alignment error vector. The error rate is then calculated as the ratio between this number and the expected number of bases in the reads.

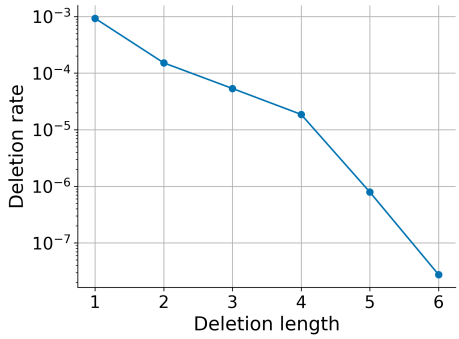


Fig. 12. Deletion length distribution (see 2.2.2.4).

5. **Cummulative distribution based upon the number of errors** (Fig. 13). This plot presents the percentage of reads in the library with x or less errors.
   - X-axis: Number of errors.
   - Y-axis: Percentage of reads with at most x errors.
   - Description: For a given number of errors x, the tool calculates the fraction of reads with at most x errors.

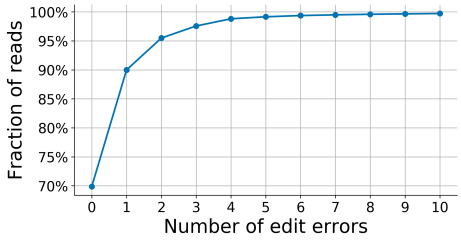


Fig. 13. Cummulative distribution based upon the number of errors (see 2.2.2.5). Note that 70% of the reads have neither sequencing nor synthesis error.

6. **GC-content error analysis** (Fig. 14). Error rates in a form of box plot based upon the GC-content. This plot depicts the reads error rates, grouped by the GC-content of their corresponding design variants. Each point represents the error rate of one of the reads in the library, with GC-content x. The box extends from the lower to upper quartile error rate of the reads with GC-content x, and plots green line at the median and green triangle at the mean value.

[2] The expected number of base x in the reads is calculated as the sum of the products of the number of base x in each of the design variants, and the number of reads matched to it.

Table 2. Synthetic DNA Libraries

	EZ-17 [9]	G-15 [12]	O-17 [20]	Y-16 [27]
Data storage size	2.11 MB	83 KB	200 MB (9.503066 MB)	3,633 bytes
Design length (bases)	152	158	150	880-1,060
Number of variants	72,000	4,991	607,150	17
Number of reads	15,787,115	3,312,235	62,879,612	6,660
Number of filtered reads	1,427,781	1,945,744	91,898	6,660
Synthesis Technology	Twist Bioscience	CustomArray	Twist Bioscience	Integrated DNA Technology (IDT)
Sequencing Technology	Ilumina miSeq	Ilumina miSeq	Ilumina NextSeq	MinION

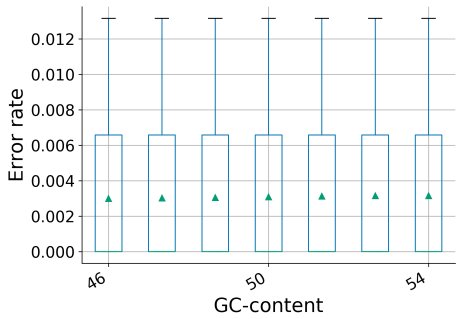


Fig. 14. Error rates by the GC-content (see 2.2.2.6).

3 Results

In this section, we present several results from the analysis of four synthetic DNA libraries. These results are based on previous experiments for storage applications conducted by Erlich and Zielinski [9], Grass et al. [12], Organnick et al. [20], and Yazdi et al. [27]. While OLs are used for a variety of applications, we focused on data storage OLs as the library data for these is typically more accessible. We matched each read with its relevant variant using edit distance estimation as will be described below. The analyzed data sets and their details are summarized in Table 2. We next present how we process the data of each experiment.

3.1 Pre-processing and Filtering of the Libraries

- Erlich and Zielinski [9], will be referred in this paper as EZ-17: As explained in Section 2.2, we analyzed this library using a sample of 1,689,315 reads out of the reported 15,787,115 reads. In this library the design length of each variant was 152. We present example results from three different filtering schemes:

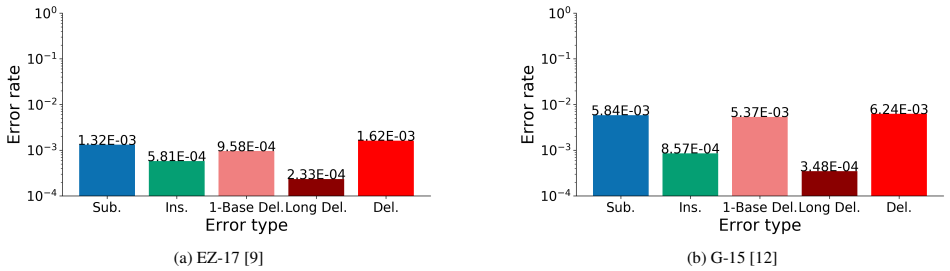
  1. Filtering only reads with length between 148 and 156 - 1,427,781 reads.
  2. Filtering only reads with length between 142 and 162 - 1,466,069 reads.
  3. Analyzing all the reads in the sample - 1,689,315 reads.

  The estimated matching between each of the reads and its design variant was calculated in two steps. First, the edit distance of the first 80 bases between the read and each of the variants was calculated. Then, the read is matched with the closest variant according to this calculation.

- Grass et al. [12], will be referred in this paper as G-15: The analysis of this library is based on all of the 3,312,235 reads. The length of each variant in this library was 158, with two primers of length 20 at the 5' end and 21 at the 3' end. The results presented were calculated according to the 117 bases of the data in each of the reads. The reads were filtered by their length: 1,945,744 reads with length between 112 to 122 bases were analyzed by the tool. The estimated matching of the reads to their corresponding design variants in G-15 was performed as in EZ-17.
- Organick et al. [20], will be referred in this paper as O-17: The analysis of this library is based on a sample of 101,243 out of the 62,879,612 reads of one file of the library. The design length of each variant in this library was 150. Similarly to G-15 [12], there were two primers of length 20 at each end. Hence, the reads were filtered by their length: we omitted the primers from each read, and analyzed 91,898 reads with length between 105 and 115 bases. The results are presented for the information bases (the primers were trimmed). The estimated matching of the reads in O-17 was performed as in EZ-17.
- Yazdi et al. [27], will be referred in this paper as Y-16: The results presented are based on all the 6,660 reads in the library. This library consists of 17 variants - 15 of length 1,000, one of length 1060 and one of length 880. The estimated matching of the reads to their corresponding design variants, was done in a similar way that used for EZ-17. However, since the number of variants was significantly smaller, we were able to calculate the edit distance for the entire strand between each read and all of the variants. Then, the read was matched with the closest variant. In this experiment, the design variants were longer and the reads were sequenced by the MinION sequencing technology. Hence, this data is likely have different error characteristics than those observed for the other three.

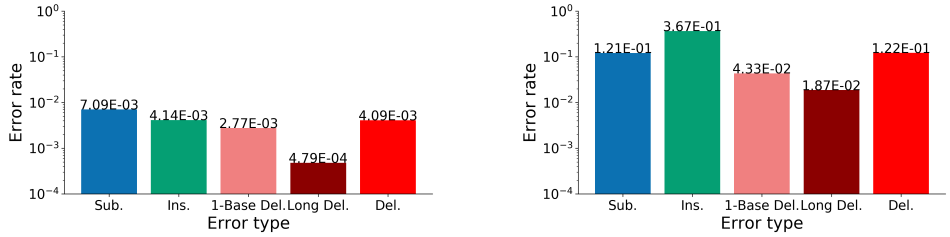
3.2 Analysis of Synthetic DNA Datasets

1. **Total error rates**. The results show significant differences between the four experiments. The three experiments of EZ-17 [9], G-15 [12], and O-17 [20] show higher rates for deletions and substitutions than insertions. EZ-17 and O-17 have the lowest error rates overall. In Y-16 [27], we observe higher rates for insertions rather than deletions and substitutions. Moreover, the error rates in Y-16 [27] are higher by two orders of magnitude than the other three. These results are presented in Fig. 15.
2. **Cummulative distributions based upon the number of errors**. As mentioned above the data of Y-16 [27] is much more erroneous. Indeed, we can see that none of its reads had less than 100 errors. In EZ-17 [9] and O-17 [20], 70% and 60% of the reads were synthesized and sequenced without any error respectively, while only 30% of the reads in G-15 [12] show no errors at all. These results are presented in Fig. 16.



(a) EZ-17 [9]

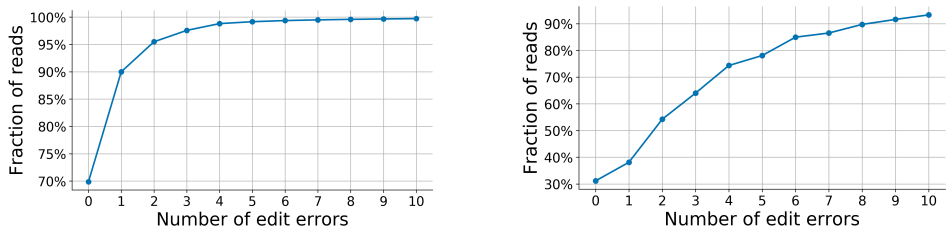
(b) G-15 [12]



(c) O-17 [20]

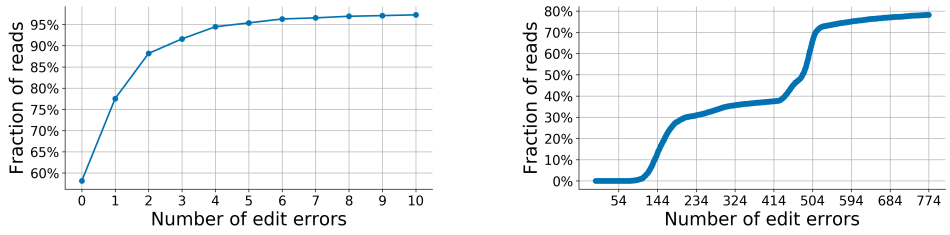
(d) Y-16 [27]

Fig. 15. Total error rates in the four datasets. Note that total error rates are also due to sequencing errors. While EZ-17 [9], G-15 [12], and O-17 [20] were sequenced on Ilumina sequencing machines, the fourth dataset, Y-16 [27], used a much noisier sequencing platform, which explains the differences in the error rates.



(a) EZ-17 [9]

(b) G-15 [12]

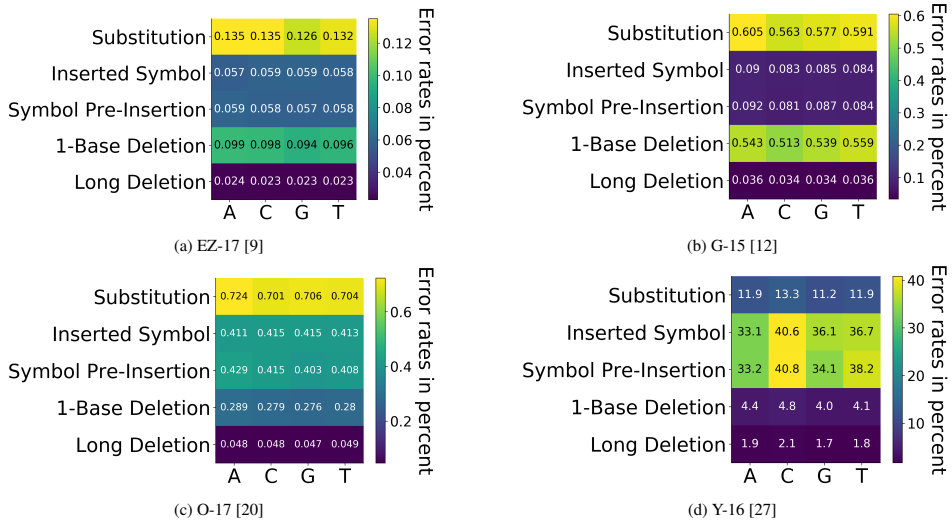


(c) O-17 [20]

(d) Y-16 [27]

Fig. 16. Cummulative distributions based upon the number of errors. Note differences in the range of Y-axes in the four figure. Also note that the X-axes are truncated (see 2.2.2.1).

3. **Error rates, stratified by symbol**. Base G showed slightly less errors compared to the other bases in EZ-17 [9]. Similarly, base A and base G showed slightly lower error rates compared to the other bases in Y-16 [27]. However, in G-15 [12], and in O-17 [20], base C was the least erroneous base. These results are presented in Fig. 17.
4. **Histograms of the length of the reads, using different filtering schemes in EZ-17 [9]**. We can see that in each of the filtering schemes we used, the length of the majority of the reads was 152 (the designed length) or shorter. These results correspond to our findings that deletions were the most dominant errors in the library. These results are presented in Fig. 18.
5. **Histograms of the cluster size per variant**. Fig. 19 shows the distributions of the cluster size per variant for the experiments in EZ-17 [9] and G-15 [12]. While the shape of the distribution of EZ-17 [9] has the form of a normal distribution, there is no similar trend in G-15 [12]. However, it is possible to notice that the number of variants



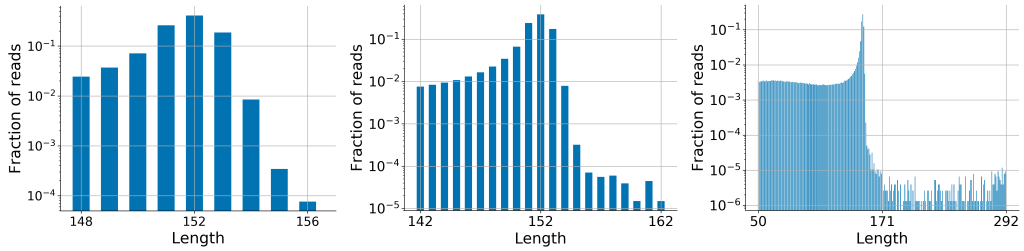
(a) EZ-17 [9]

(b) G-15 [12]

(c) O-17 [20]

(d) Y-16 [27]

Fig. 17. Error rates, stratified by symbol in the four datasets (see 2.2.2.2). Note that the colorbars are different in each plot.



(a) (b) (c)

Fig. 18. Histograms of the length of the reads in EZ-17 [9], using different filtering schemes (1), (2), and (3), respectively (see 2.2.1.4) .

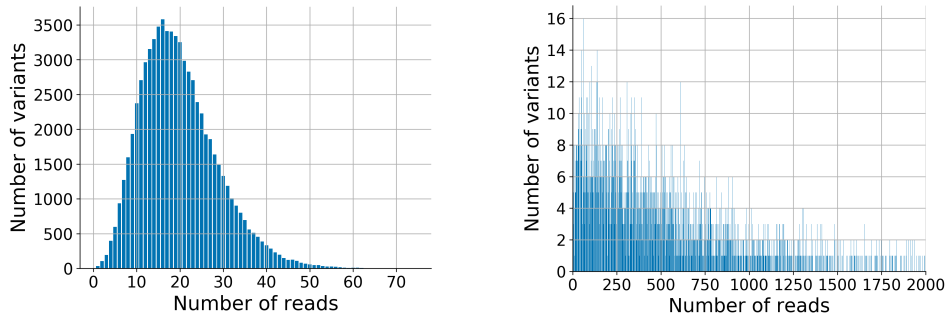
decreases with the size of the cluster size. Furthermore, in G-15 [12] we also observed very large clusters of size ranging between 2,000 and 8,000, which were omitted from the figure for its clarity.

6. **Error rates by GC-content**. The results (presented in Fig. 20) show that the median and the mean values of the read error rates increase with its designed GC-content in G-15 [12]. In Y-16 [27] we can surprisingly see error rates which are greater than 1. Such high error rates are encountered when there is a large number of insertions together with deletions and substitutions in the read such that the number of errors is strictly larger than the design length. These results corresponding to our findings that insertions were the most dominant errors in this library.
7. **Error rate per position**. In all four experiments analyzed, the error rates in the 3' end are greater than the error rates in the 5' end. Note that in Y-16 [27] there were different design lengths: 880, 1,000 and 1,060. For uniformity, we present only results of reads which correspond to variants of length 1,000. These results are presented in Fig. 21.

4 Use-Case Examples

In this section we present several use-case examples for SOLQC.

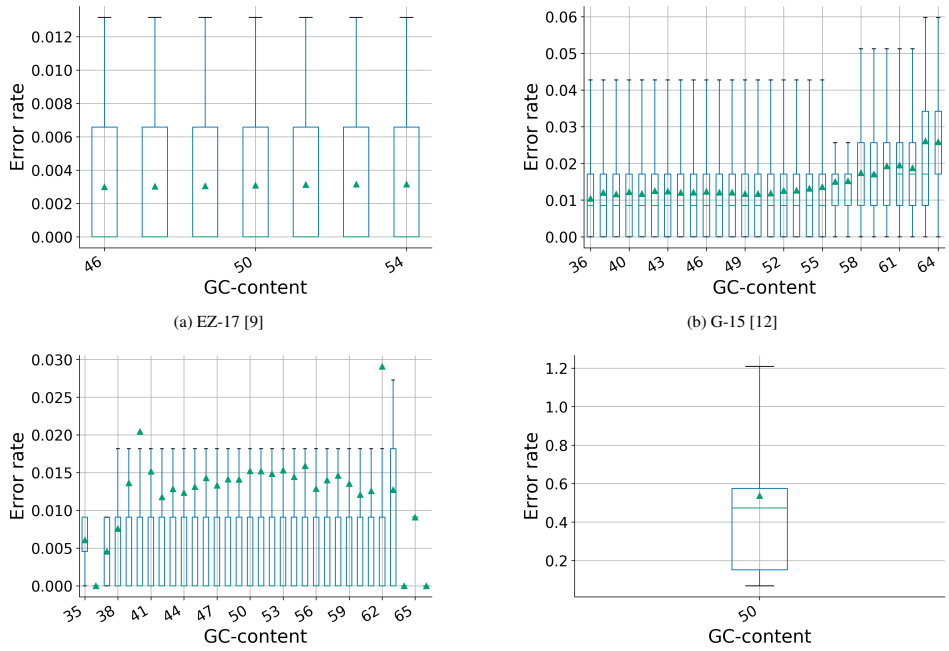
a. **Design-quality evaluation**. Different libraries can have different robustness levels. For example: a design of one library can have many homopolymers, while another can limit the presence of homopolymers. SOLQC outputs statistical reports describing the error behavior of a given library/design. The user can create several small test experiments with different designs and properties. Then, the user can use SOLQC to evaluate the effect of different designs on the error behavior. This analysis can then be considered as part of the final design of the library.
b. **Binning of synthetic DNA-libraries**. The result of a sequencing reaction on a given library does not include the matching of each read to its design variant. SOLQC provides several methods to bin the reads according to their corresponding design variants. The matching/clustering methods can be performed on libraries with or without the barcode. In addition, users can get coverage depth statistics from SOLQC as well as quality related statistics, which can be different for different variants or set of variants. Lastly, in applications like data storage, the set of reads that is binned to any given variant can be used in order to decode the stored variant.



(a) EZ-17 [9]

(b) G-15 [12]

Fig. 19. Histograms of the cluster size per variant in two of the four datasets (see 2.2.1.2).



(a) EZ-17 [9]

(b) G-15 [12]

(c) O-17 [20]

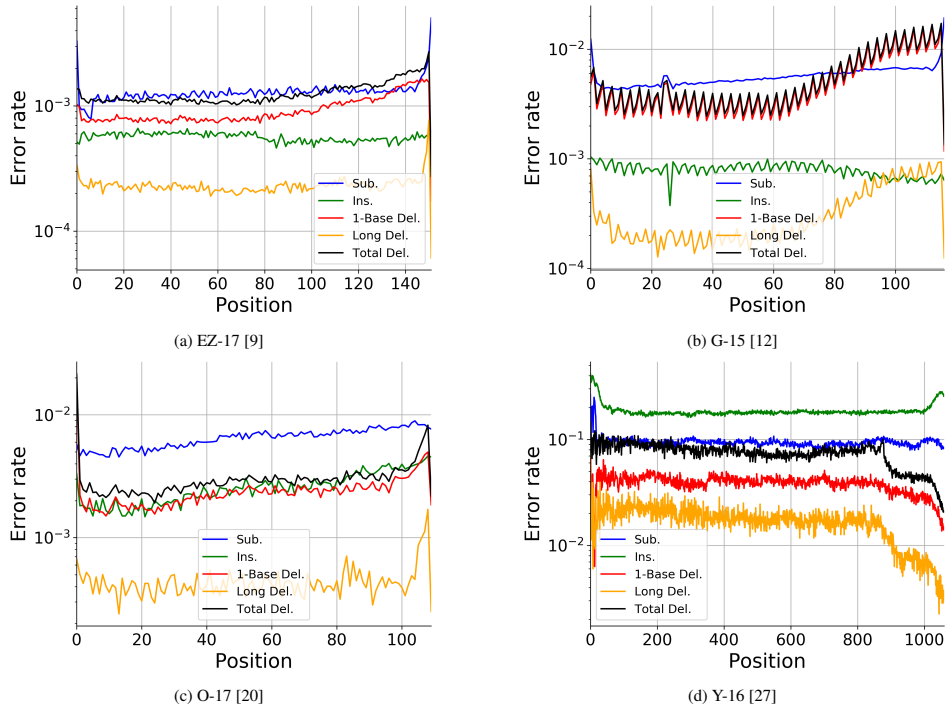
(d) Y-16 [27]

Fig. 20. Error rates by the GC-content in the four datasets. Note the differences in the range of the Y-axis in the four figures (see 2.2.2.6).

c. **Comparison of different synthesis and sequencing technologies**. SOLQC provides its users composition and errors statistics. Accordingly, the user can synthesize libraries using several synthesis technologies and their process parameters. Then, the user can compare the quality of the results of each technology and/or of each parameter configuration. In order to optimize the process parameters, the experiments can be conducted with the same design while using different parameter configuration. Thus, it is possible to determine how to choose the best configuration.

d. **Design of error-correcting codes and coding techniques for DNA-storage**. In data storage applications, SOLQC can be used as a characterization tool of the DNA channel. The user can characterize the DNA channel using data from previous experiments of various technologies and design parameters. Then, using this information, the user can design appropriate error-correcting codes and coding techniques to improve the error rates.

e. **Standardization and reproducibility**. SOLQC enables determining whether a library is behaving as previous libraries from the same vendor with similar preparation characteristics. This



(a) EZ-17 [9]

(b) G-15 [12]

(c) O-17 [20]

(d) Y-16 [27]

Fig. 21. Error rates per position in the four datasets. X-axis represents position counted from the 5' end of the designed variant, note the differences in the range of the X-axis and Y-axis in the four figures (see 2.2.2.3).

enables comparison between the same library preparation protocol performed in different labs, or in the same lab at different times or by different lab members.

5 Discussion

In this work we presented SOLQC, a software tool designed to characterize synthetic DNA libraries. While SOLQC provides many useful tools and features, it will benefit from further development in several aspects:

a. **Matching approximations**. Matching each read with its design variant is a complex calculation, especially when the library is not barcoded and/or when there are many variants in the library. In fact, the matching step is the heaviest step in any initial run of the SOLQC pipeline. Hence, we plan to provide, in the future, several faster approximation approaches to the matching step.

b. **Reconstruction algorithms**. When synthetic DNA libraries are used for data storage applications, the first step of decoding the data is to reconstruct the original variant out of the noisy reads. We plan to add to SOLQC additional features related to this step. Thus, SOLQC will perform reconstruction on a given cluster of reads, in order to decode the sequence of the original variant.

c. **Additional statistics**. SOLQC will report several more statistics in the future. These statistical analysis will examine more deeply whether there is a connection between the characteristics of the design variants and the errors observed for them.

6 Installation Link

https://yoav-orlev.gitbook.io/solqc/.

7 Acknowledgement

We thank the authors of [9, 12, 20, 27] for sharing and providing the data of their DNA-storage experiments to this paper. We also thank Hossein Yazdi, Lee Organick, Karin Strauss, and Yaniv Erlich for helpful discussions. We thank the Yakhini research group for useful discussion and comments. We thank Roee Amit and his lab, especially Sarah Goldberg, for meaningful discussion and useful feedback. Finally, we thank Matilda Lidgi, Danit Goldberg, Amir Biran, Alex Yucovich and Batel Carmona for their great contribution to this work. This project has received funding from the European Union's Horizon 2020 Research And Innovation Programme under grant agreement No. 851018.

References

[1]L. Anavy, I. Vaknin, O. Atar, R. Amit, and Z. Yakhini. Data storage in DNA with fewer synthesis cycles using composite DNA letters. *Nature Biotechnology*, 37(10):1229–1236, 2019.

[2]S. L. Beaucage and R. P. Iyer. Advances in the synthesis of oligonucleotides by the phosphoramidite approach. *Tetrahedron*, 48(12):2223 – 2311, 1992.

[3]M. Blawat, K. Gaedke, I. Hütter, X.-M. Chen, B. Turczyk, S. Inverso, B. W. Pruitt, and G. M. Church. Forward error correction for DNA data storage.

Procedia Computer Science, 80:1011–1022, 2016.

[4]J. Bornholt, R. Lopez, D. M. Carmean, L. Ceze, G. Seelig, and K. Strauss. A DNA-based archival storage system. *ACM SIGARCH Computer Architecture News*, 44(2):637–649, 2016.

[5]Y. Choi, T. Ryu, A. C. Lee, H. Choi, H. Lee, J. Park, S.-H. Song, S. Kim, H. Kim, W. Park, and S. Kwon. High information capacity DNA-based data storage with augmented encoding characters using degenerate bases. *Scientific Reports*, 9(1):6582, 2019.

[6]G. M. Church, Y. Gao, and S. Kosuri. Next-generation digital information storage in DNA. *Science*, 337(6102):1628–1628, 2012.

[7]C. T. Clelland, V. Risca, and C. Bancroft. Hiding messages in DNA microdots. *Nature*, 399(6736):533, 1999.

[8]L. d'Espaux, D. Mendez-Perez, R. Li, and J. D. Keasling. Synthetic biology for microbial production of lipid-based biofuels. *Current opinion in chemical biology*, 29:58–65, 2015.

[9]Y. Erlich and D. Zielinski. DNA fountain enables a robust and efficient storage architecture. *Science*, 355(6328):950–954, 2017.

[10]D. G. Gibson, J. I. Glass, C. Lartigue, V. N. Noskov, R.-Y. Chuang, M. A. Algire, G. A. Benders, M. G. Montague, L. Ma, M. M. Moodie, et al. Creation of a bacterial cell controlled by a chemically synthesized genome. *Science*, 329(5987):52–56, 2010.

[11]N. Goldman, P. Bertone, S. Chen, C. Dessimoz, E. M. LeProust, B. Sipos, and E. Birney. Towards practical, high-capacity, low-maintenance information storage in synthesized DNA. *Nature*, 494(7435):77, 2013.

[12]R. N. Grass, R. Heckel, M. Puddu, D. Paunescu, and W. J. Stark. Robust chemical preservation of digital information on DNA in silica with error-correcting codes. *Angewandte Chemie International Edition*, 54(8):2552–2555, 2015.

[13]R. Heckel, G. Mikutis, and R. N. Grass. A characterization of the DNA data storage channel. *arXiv preprint arXiv:1803.03322*, 2018.

[14]A. D. Johnson. An extended iupac nomenclature code for polymorphic nucleic acids. *Bioinformatics*, 26(10):1386–1389, 2010.

[15]S. Kosuri and G. M. Church. Large-scale de novo DNA synthesis: technologies and applications. *Nature Methods*, 11(5):499, 2014.

[16]E. Kotler, O. Shani, G. Goldfeld, M. Lotan-Pompan, O. Tarcic, A. Gershoni, T. A. Hopf, D. S. Marks, M. Oren, and E. Segal. A systematic p53 mutation library links differential functional impact to cancer mutation pattern and evolutionary conservation. *Molecular Cell*, 71(1):178–190, 2018.

[17]A. Leier, C. Richter, W. Banzhaf, and H. Rauhe. Cryptography with DNA binary strands. *Biosystems*, 57(1):13–22, 2000.

[18]L. Levy, L. Anavy, O. Solomon, R. Cohen, M. Brunwasser-Meirom, S. Ohayon, O. Atar, S. Goldberg, Z. Yakhini, and R. Amit. A synthetic oligo library and sequencing approach reveals an insulation mechanism encoded within bacterial σ 54 promoters. *Cell Reports*, 21(3):845–858, 2017.

[19]L. A. Miles, R. J. Garippa, and J. T. Poirier. Design, execution, and analysis of pooled in vitro crispr/cas9 screens. *The FEBS Journal*, 283(17):3170–3180, 2016.

[20]L. Organick, S. D. Ang, Y.-J. Chen, R. Lopez, S. Yekhanin, K. Makarychev, M. Z. Racz, G. Kamath, P. Gopalan, B. Nguyen, C. N. Takahashi, S. Newman, H.-Y. Parker, C. Rashtchian, K. Stewart, G. Gupta, R. Carlson, J. Mulligan, D. Carmean, G. Seelig, L. Ceze, and K. Strauss. Random access in large-scale DNA data storage. *Nature Biotechnology*, 36:242 EP –, 02 2018.

[21]W. Pan, M. Byrne-Steele, C. Wang, S. Lu, S. Clemmons, R. J. Zahorchak, and J. Han. DNA polymerase preference determines pcr priming efficiency. *BMC Biotechnology*, 14(1):10, 2014.

[22]J. Ruijter, C. Ramakers, W. Hoogaars, Y. Karlen, O. Bakker, M. Van den Hoff, and A. Moorman. Amplification efficiency: linking baseline and bias in the analysis of quantitative pcr data. *Nucleic Acids Research*, 37(6):e45–e45, 2009.

[23]E. Sharon, Y. Kalma, A. Sharp, T. Raveh-Sadka, M. Levo, D. Zeevi, L. Keren, Z. Yakhini, A. Weinberger, and E. Segal. Inferring gene regulatory logic from high-throughput measurements of thousands of systematically designed promoters. *Nature biotechnology*, 30(6):521, 2012.

[24]C. Sheridan. Synthetic biology firms pivot from biofuels to cheap biologics, 2016.

[25]M. Šošić and M. Šikić. Edlib: a c/c++ library for fast, exact sequence alignment using edit distance. *Bioinformatics*, 33(9):1394–1395, 2017.

[26]J. Tian, H. Gong, N. Sheng, X. Zhou, E. Gulari, X. Gao, and G. Church. Accurate multiplex gene synthesis from programmable DNA microchips. *Nature*, 432(7020):1050, 2004.

[27]S. H. T. Yazdi, R. Gabrys, and O. Milenkovic. Portable and error-free DNA-based data storage. *Scientific Reports*, 7(1):5011, 2017.

[28]S. H. T. Yazdi, Y. Yuan, J. Ma, H. Zhao, and O. Milenkovic. A rewritable, random-access DNA-based storage system. *Scientific Reports*, 5:14138, 2015.

2 Single cell RNA seq and investigating the intrinsic dimension of synthetic sparse embedded data

2.1 Background

High throughput measurement of gene expression has led to significant progress in our understanding of cellular and organism level biological functions in health and disease. In an early review [4] the authors argued the importance of gene expression profiling. Many other studies , such as [5], [6],[7], [8] describe examples of how these profiles can be useful in various contexts. Single cell measurement gives an even more accurate picture of the sample and can thus support even greater understanding and possibly greater clinical or process relevance. Single cell sequencing examines the sequence information from individual cells with optimized next-generation sequencing (NGS) technologies, providing a higher resolution of cellular differences and a better understanding of the function of an individual cell in the context of its micro-environment [9]. Single-cell RNA-sequencing (scRNA-seq) has emerged as a revolutionary tool that allows researchers to address scientific questions that eluded examination just a few years ago. In 2013, Sandberg claimed we are entering the era of scRNA-seq in biology and medicine [10], and Svensson et al. reviewed the exponential growth of the technique [11]. Alongside the advantages of scRNA-seq come computational challenges that are just beginning to be addressed. Recently, we have seen a large number of works trying to apply deep learning techniques on this type of data [12], [13],[14], [15]. Deep learning has also been applied to other domains in genomics. In [**?**] the authors report the prediction of methylation status from sequence and expression data. In [16] the authors provide an overview of applications in medicine, taking a molecular medicine angle.

Autoencoders, a method which dates back to 1986 [17], [18], is an unsuperivsed learning method that addresses represntation learning, denoising, data generation and more. This chapter is motivated by the potential use of autoencoders in analyzing scRNA seq data [19], which is

typically of very high dimension. A basic theoretical question related to this potential use of autoencoders is what bottle neck dimension should be used to improve the performance. Gupta et al. [20] use an automated elbow method for this task, in the context of text data. Motivated by the scRNA seq question we continued to investigating whether autoencoders capture the intrinsic dimension of low dimensional data which is embedded in higher dimensional ambient space, using synthetic data. We have not yet found literature on autoencoders bottleneck investigation using synthetic data where the intrinsic dimension is known. We hereby describe the results of the initial analysis of the scRNA seq data and the subsequent theoretical investigation related to capturing the intrinsic dimension of synthetic data.

2.2 Classification of human blood cells by scRNA seq profiles

We worked with Dr. Roy Avraham's lab at the Weizmann Institute of Science on investigating scRNA seq acquired for samples of infected and non infected blood cells. The goal of his research is to understand the infection mechanism, in terms of the host molecular level response.
The data was gathered in two studies. In the first, blood cells were extracted from a single individual. Then, a portion of the cells were infected with salmonella while a healthy population was kept on the side. Both were sequenced. in the second study blood cells were extracted from 2 individuals, and here, the cells were infected with both salmonella and listeria. Again, both infected and healthy cell populations were sequenced. A break down of the cells is shown in Figure 1.

Given this data we've constructed a feed forward neural net which achieves 85.7% classification (infected vs non-infected) accuracy, an AUROC of 0.84, average precision of 0.88, average recall of 0.86 and a f1-score of 0.85. This is compared to a logistic regression base-line of 0.75% accuracy, 0.73 AUROC, 0.82 precision, 0.76 recall and a f1-score of 0.74. (the non-standard train-test split is discussed below). In seeking approaches to improve that score we turned to in-

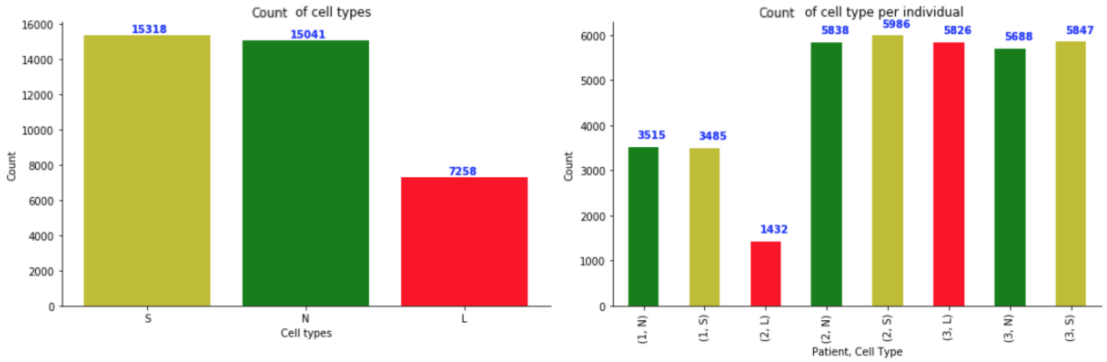


Figure 1: Number of cells sequenced broken according to the different populations investigated in the study (S- Salmonella, N - Naive, L - Listeria). In the right panel the three individuals are separated for the counting.

vestigating dimensionality reduction and autoencoders. The classifier confusion matrix is reported below in Figure 2:

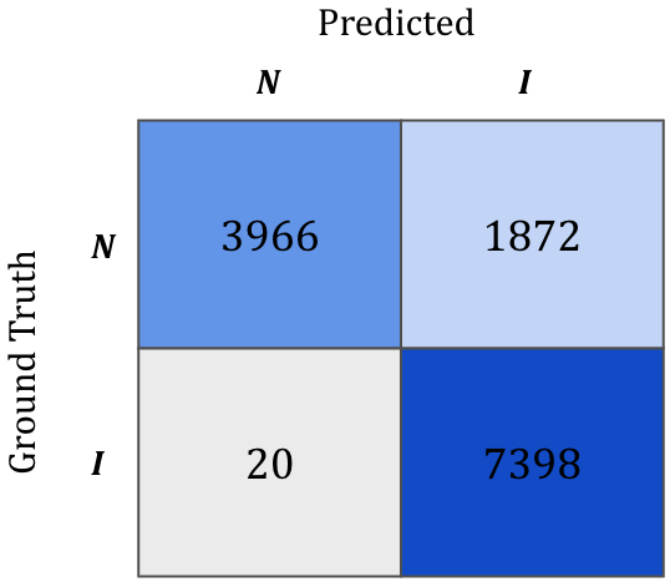


Figure 2: Confusion Matrix. Size of test set = 13,256, consisting of cells of all 3 types from Individual number 2.

We note that this is not a classical dataset for machine learning models since most of the

(single cell) samples are highly correlated with one another. Therefore the performance reported is based on splitting the data so that Individuals 1 and 3 served as training set, while Individual 2 served as test set.

2.3 The intrinsic dimension of sparse data: an investigation using synthetic data

An important issue, for scRNA seq is the sparsity of the data. As indicated above, one may hope to overcome the sparsity issue by working in a dimension that better fits the data. We describe an approach here, to help us understand the relationship between the bottle neck dimension of autoencoders and the intrinsic dimensions of data they are able to faithfully represent. Next steps may allow us to discover a meaningful dimension of scRNA sequence data for clustering purposes and for improved classification performance. The current dimension of the Avraham data, for example, is $\sim$33k. Since most of the genes introduce noise to the network, our hypothesis is that once such an intrinsic dimension is found, we would be able to improve the classification performance. Another motivation comes from gene clustering. We believe that working in a latent space with better representation can facilitate in finding improved clustering insights and features. We now explore, therefore, how autoencoders can learn a known intrinsic dimension of synthetic data. This synthetic data resides in dimension a, but was generated from a dimension of size t where $a > t$. We would like to demonstrate that an autoencoder can learn the intrinsic dimension of the data, t. We can demonstrate this by showing that a very low reconstruction loss can be obtained with a latent dimension of t, and that using a higher latent space dimension has little effect on the loss since the added dimensions are redundant.

2.3.1 Methods

Data We performed 4 experiments to study this question, working with different synthetic datasets.

1. The first experiment, which serve as kind of a base for the others followed this procedure:

   (a) Randomly generate t vectors, each of dimension a. Denote the resulting $t \times a$ matrix as B.

   (b) Generate n vectors, each of dimension t. Denote the resulting $n \times t$ matrix as S.

   (c) Let $D = S \times B$

   This results in n vectors of dimension a (we refer to this size as the Ambient Dimension). These vectors, however, all reside in a t-dim (linear) manifold. Given a row vector $d \in D$ and the matrix B, we only need the i scalars that operated on each vector in B to fully recover d.

2. The second experiment followed the same procedure, only at the end we squared each value in D to introduce a nonlinear transformation.

3. The third experiment followed a similar base procedure, and this time, for the matrix S instead of sampling n vectors in dimension t we sampled n vectors in dimension $t-1$ and set $x_t = \sqrt{\sum_{x_0}^{x_{t-1}} {x_t}^2}$. The challenge here was to see if we can train an autoencoder with a bottle neck of size $t-1$.

4. The fourth experiment followed a similar base procedure and at the end we add random noise ,coordinate wise, by sampling values from multivariate log-gaussian with mean 0 and standard deviation of 0.02.

   We tested out 2 parameter settings:

(a) $t = 10$ $a = 100$ and $n = 10000$

(b) $t = 25$ $a = 200$ and $n = 10000$

Architecture We wanted an architecture that will be able to learn all of our synthetic datasets without the need to adjust for each one individually. This made the comparison between experiments and conclusions easier as there are less variables to take into consideration.

The model, as in most auto-encoders, consists of an encoder and a decoder.

The encoder consists of 3 Linear Residual blocks (LRB), which are explained in the next part, all of size 32. The bottle neck layer is different from experiment to experiment, as this is the area we focus our investigation on. The decoder is a mirror of the encoder and consists of 3 layers of sizes 32. Both the input and the output layer are dependent on the ambient space. A schematic of the network is described in Figure 4.

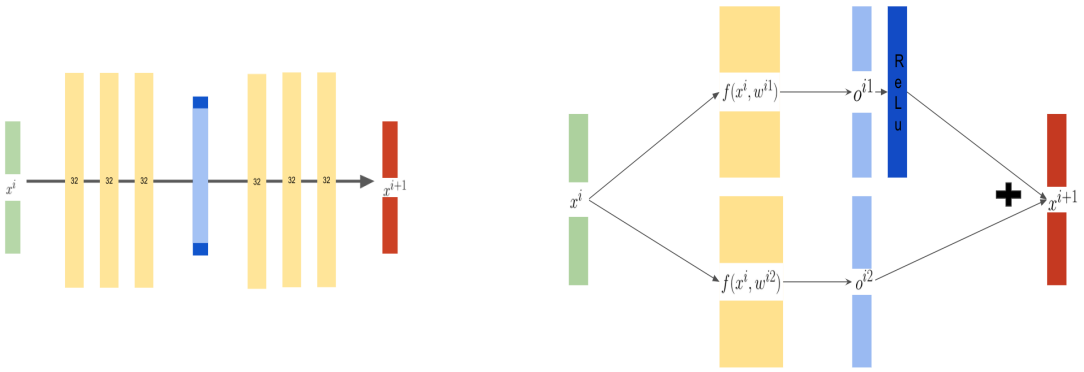


Figure 3: Auto-encoder architecture on the left and the lrb block on the right..

Deep learning methodologies At first we worked on finding the right architecture that will result in a low reconstruction loss for all data sets.

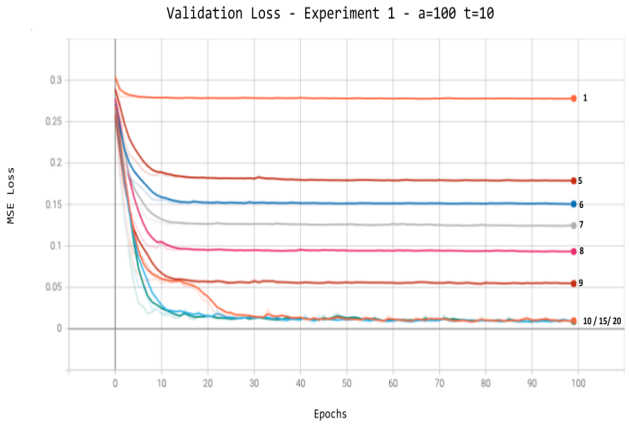
Our early experiments showed that in order to resolve our vanilla experiment (Experiment 1) all we had to do is construct a network without any activation's. Basically, acting as an iterative optimization solver for the PCA problem. But obviously this kind of architecture will fail once we

try to employ it on a data that has seen a non-linear transformation. This lead us to understanding that we want to enable the network to have a linear "path" in the forward pass, but still have the ability to learn non-linear transformations.

Drawing inspiration from [21] we came up with an architecture component we call a Linear Residual Block or LRB for short. The LRB works as follows: Given an input x^i we have 2 sets of weights w^{i1}, w^{i2}, which will output two results of the same size $o^{i1} = Relu(f(x^i, w^{i1})), o^{i2} = f(x^i, w^{i2})$. The final output of the layer will be $o^{i1} + o^{i2}$. Under this construction if the network is required to learn a linear transformation at a certain layer all that is required is to set the weights of w^{i1} to zero. Figure 3 illustrates the schematics of our architecture, which uses the LRB.

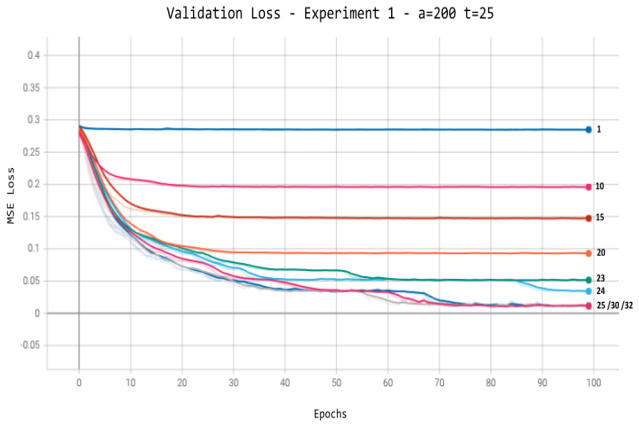
2.3.2 Results

1. Experiment 1 - Data was generated by randomly generating ambient base vectors and then coefficients for the data points. Data is then is then produced by a matrix multiplication (see 2.3.1).



Bottle Neck	Test Loss
1	0.278
5	0.1781
6	0.1506
7	0.1245
8	0.09
9	0.05
10	0.0095
15	0.0095
20	0.0086

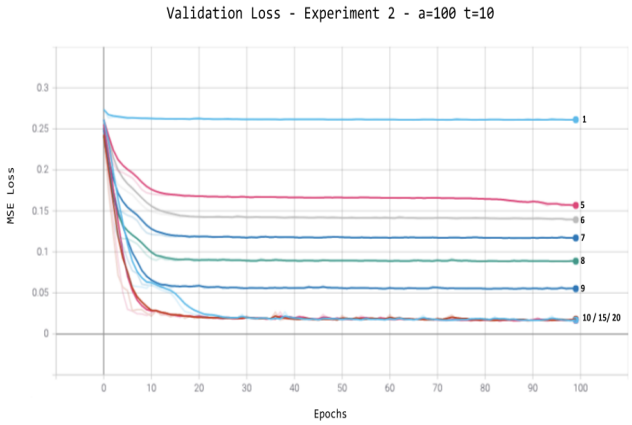
Figure 4 & Table 1: Results on the first Data Set (linear). The figure on the left shows the validation loss throughout the experiment.



Bottle Neck	Test Loss
1	0.2848
10	0.1958
15	0.1472
20	0.0928
23	0.0514
24	0.0342
25	0.0111
30	0.0115
32	0.0118

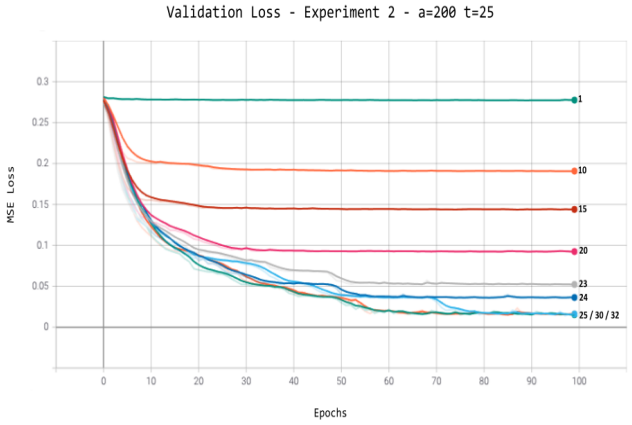
Figure 5 & Table 2: Results on the first Data Set (linear). The figure on the left shows the validation loss throughout the experiment.

2. Experiment 2 - Data was generated by sampling ambient vectors and coefficients. After matrix multiplication each entry is squared.



Bottle Neck	Test Loss
1	0.2612
5	0.1567
6	0.1392
7	0.1170
8	0.0800
9	0.05516
10	0.01701
15	0.01795
20	0.01668

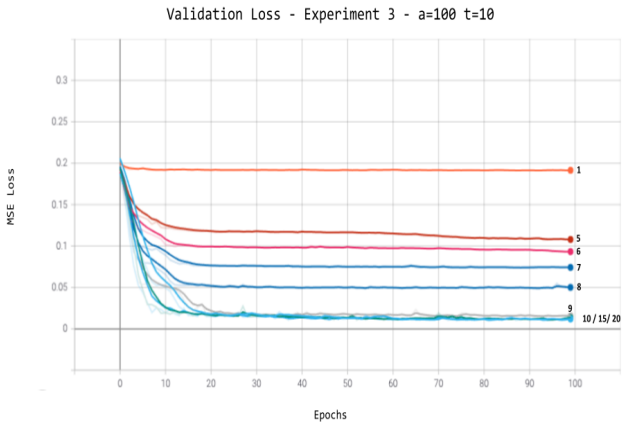
Figure 6: Results on the second Data Set (data values are squared). The figure on the left shows the validation loss throughout the experiment.



Bottle Neck	Test Loss
1	0.2774
10	0.1918
15	0.1449
20	0.0930
23	0.0690
24	0.03
25	0.01603
30	0.01525
32	0.01614

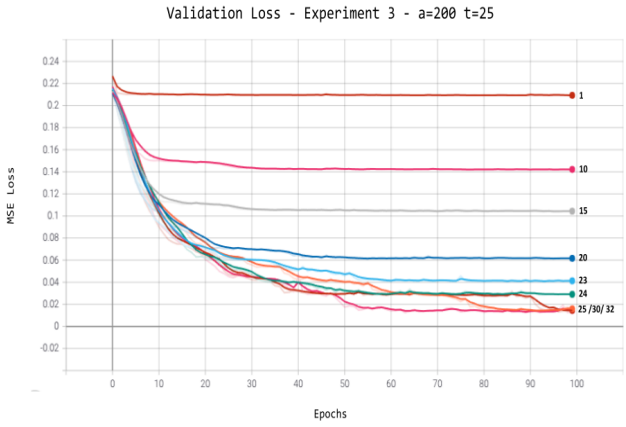
Figure 7 & Table 4: Results on the second Data Set (Data values are squared). The figure on the left shows the validation loss throughout the experiment.

3. Experiment 3 - Data was generated by sampling ambient vectors and t-1 coefficients, the last coefficient is functionally dependent on the the other coefficients. In this case that last entry is the norm of the $t-1$ dimensional vector represented by the others.



Bottle Neck	Test Loss
1	0.1915
5	0.1079
6	0.0932
7	0.0741
8	0.0501
9	0.0157
10	0.0119
15	0.01335
20	0.01202

Figure 8 & Table 5: Results on the third Data Set (functional dependence - the Cone). The figure on the left shows the validation loss throughout the experiment.

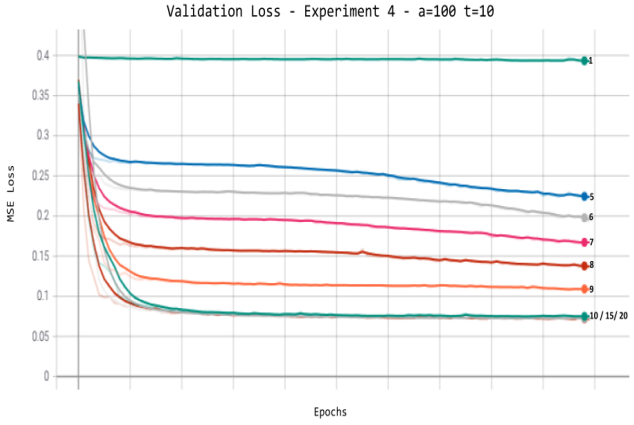


Bottle Neck	Test Loss
1	0.2093
10	0.1422
15	0.1043
20	0.0617
23	0.0288
24	0.01526
25	0.01453
30	0.01450
32	0.0.1437

Figure 9 & Table 6: Results on the third Data Set (functional dependence). The figure on the left shows the validation loss throughout the experiment.

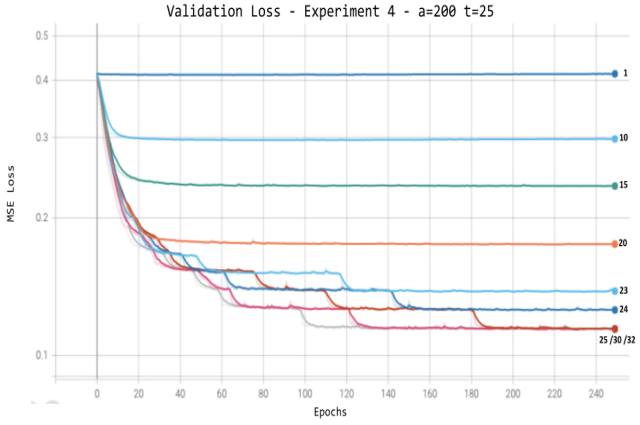
4. Experiment 4 - Data was generated by sampling ambient vectors and coefficients, then using matrix multiplication and adding noise.

The effect of noise Perturbing data which has a low intrinsic dimension, even with a small magnitude full dimension noise, would give rise to data that also has the full dimension. In effect, however, such data is very close to actually living on a low dimension manifold. The models described above used the clean data. In this section we do add noise and investigate the effect of noise on the autoencoder performance and on the inferred dimension.



Bottle Neck	Test Loss
1	0.3933
5	0.2244
6	0.1978
7	0.1672
8	0.1370
9	0.1091
10	0.0747
15	0.07253
20	0.07214

Figure 10 & Table 7: Results on the fourth data set (adding noise). The figure on the left shows the validation loss throughout the experiment. The table on the right displays the test loss.



Bottle Neck	Test Loss
1	0.4133
10	0.0.2977
15	0.2348
20	0.1752
23	0.1383
24	0.1258
25	0.1143
30	0.1142
32	0.1142

Figure 11 & Table 8: Results on the fourth data set (Added noise). The figure on the left shows the validation loss throughout the experiment.

2.3.3 Discussion

We have demonstrated the ability of autoencoders to learn a latent dimension matching the dimension of the embedded data. That is - the intrinsic dimension is the one that attains h minimal loss. We also see a negligible loss improvement, if any, when the latent dimension is higher. Future research directions include exploiting various ways in which autoencoders can be used to discover more meaningful latent spaces for gene expression data analysis.

An exploration software tool We encapsulated our software into a tool that enables users to understand the relationship between autoencoder bottleneck dimensions and data that they can faithfully represent. The tool can be found under this repo - https://github.com/yoavo1984/intrinsic$_{a}utoencoders$

Future Directions The topic we had touched upon in the study described above is, in fact, much broader and deeper than covered in this work. Specific future directions for further investigation include:

1. Extrapolate a technique for finding an approximation of the intrinsic dimension (these

should be defined) for data that represent a simple manifold.

2. Use the above technique to approximate the intrinsic dimension on actual scRNA seq data and use different techniques to see if meaningful insights can be extracted from it. This includes possibly improved classification and/or clustering performance in the latent space of the autoencoder.

3. Different data types and structure, like vision and audio.

4. Better characterize dependence on noise.

3 References

References

[1] Omer Sabary, Yoav Orlev, Roy Shafir, Leon Anavy, Eitan Yaakobi, and Zohar Yakhini. Solqc: Synthetic oligo library quality control tool. *Bioinformatics*, 08 2020. btaa740.

[2] Leon Anavy, Inbal Vaknin, Orna Atar, Roee Amit, and Zohar Yakhini. Data storage in dna with fewer synthesis cycles using composite dna letters. *Nature biotechnology*, 37(10):1229–1236, 2019.

[3] Luis Ceze, Jeff Nivala, and Karin Strauss. Molecular digital data storage using dna. *Nature Reviews Genetics*, 20(8):456–466, 2019.

[4] Stéphane Audic and Jean-Michel Claverie. The significance of digital gene expression profiles. *Genome research*, 7(10):986–995, 1997.

[5] Todd R Golub, Donna K Slonim, Pablo Tamayo, Christine Huard, Michelle Gaasenbeek, Jill P Mesirov, Hilary Coller, Mignon L Loh, James R Downing, Mark A Caligiuri, et al.

Molecular classification of cancer: class discovery and class prediction by gene expression monitoring. *science*, 286(5439):531–537, 1999.

[6] Meltzer Bittner, Paul Meltzer, Yidong Chen, Youfei Jiang, Elisabeth Seftor, M Hendrix, M Radmacher, Richard Simon, Zohar Yakhini, Amir Ben-Dor, et al. Molecular classification of cutaneous malignant melanoma by gene expression profiling. *Nature*, 406(6795):536–540, 2000.

[7] Ingrid Hedenfalk, David Duggan, Yidong Chen, Michael Radmacher, Michael Bittner, Richard Simon, Paul Meltzer, Barry Gusterson, Manel Esteller, Mark Raffeld, et al. Gene-expression profiles in hereditary breast cancer. *New England Journal of Medicine*, 344(8):539–548, 2001.

[8] Keiichi Kodama, Momoko Horikoshi, Kyoko Toda, Satoru Yamada, Kazuo Hara, Junichiro Irie, Marina Sirota, Alexander A Morgan, Rong Chen, Hiroshi Ohtsu, et al. Expression-based genome-wide association study links the receptor cd44 in adipose tissue with type 2 diabetes. *Proceedings of the National Academy of Sciences*, 109(18):7049–7054, 2012.

[9] James Eberwine, Jai-Yoon Sul, Tamas Bartfai, and Junhyong Kim. The promise of single-cell sequencing. *Nature methods*, 11(1):25–27, 2014.

[10] Rickard Sandberg. Entering the era of single-cell transcriptomics in biology and medicine. *Nature methods*, 11(1):22–24, 2014.

[11] Valentine Svensson, Roser Vento-Tormo, and Sarah A Teichmann. Exponential scaling of single-cell rna-seq in the past decade. *Nature protocols*, 13(4):599–604, 2018.

[12] Xiangjie Li, Kui Wang, Yafei Lyu, Huize Pan, Jingxiao Zhang, Dwight Stambolian, Katalin Susztak, Muredach P Reilly, Gang Hu, and Mingyao Li. Deep learning enables accurate

clustering with batch effect removal in single-cell rna-seq analysis. *Nature communications*, 11(1):1–14, 2020.

[13] Carlos Torroja and Fatima Sanchez-Cabo. Digitaldlsorter: Deep-learning on scrna-seq to deconvolute gene expression data. *Frontiers in genetics*, 10:978, 2019.

[14] Cédric Arisdakessian, Olivier Poirion, Breck Yunits, Xun Zhu, and Lana X Garmire. Deepimpute: an accurate, fast, and scalable deep neural network method to impute single-cell rna-seq data. *Genome biology*, 20(1):1–14, 2019.

[15] Babak Alipanahi, Andrew Delong, Matthew T Weirauch, and Brendan J Frey. Predicting the sequence specificities of dna-and rna-binding proteins by deep learning. *Nature biotechnology*, 33(8):831–838, 2015.

[16] Michael Wainberg, Daniele Merico, Andrew Delong, and Brendan J Frey. Deep learning in biomedicine. *Nature biotechnology*, 36(9):829–838, 2018.

[17] Dana H Ballard. Modular learning in neural networks. In *AAAI*, pages 279–284, 1987.

[18] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning internal representations by error propagation. Technical report, California Univ San Diego La Jolla Inst for Cognitive Science, 1985.

[19] Gökcen Eraslan, Lukas M Simon, Maria Mircea, Nikola S Mueller, and Fabian J Theis. Single-cell rna-seq denoising using a deep count autoencoder. *Nature communications*, 10(1):1–14, 2019.

[20] Parth Gupta, Rafael E. Banchs, and Paolo Rosso. Squeezing bottlenecks: Exploring the limits of autoencoder semantic representation capabilities. *Neurocomputing*, 175:1001–1008, 2016.

[21] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

המרכז הבינתחומי בהרצליה
בית-ספר אפי ארזי למדעי המחשב
התכנית לתואר שני (M.Sc.) - מסלול מחקרי

הפעלת שיטות סטטיסטיות ולמידה בביולוגיה מולקולרית כולל בקרת איכות לד.נ.א סינתטי

מאת
יואב אורלב

עבודת תזה המוגשת כחלק מהדרישות לשם קבלת תואר מוסמך M.Sc.
במסלול המחקרי בבית ספר אפי ארזי למדעי המחשב, המרכז הבינתחומי הרצליה

מאי 2021

תקציר

SOLQC - בשנים האחרונות אנו עדים לתנופה מחקרית ומספר הולך וגדל של שימושים מעשיים בספריות דנ"א סינטטיות בעולם הבילוגיה הסינטטית.
ככל שניסויים אלו גדלים במספרים ובמורכבות, כלי ניתוח ואנליזה יכולים להקל על בקרת האיכות ולעזור בהערכה והסקת נתונים סטטיסטיים.
אנו מציגים כלי ניתוח חדשני, הנקרא SOLQC, המאפשר ניתוח מהיר ומקיף של ספריות דנ"א סיננטיות.
SOLQC לוקח כקלט ניתוח של ספריה שעברה ריצוף (NGS) על ידי המשתמש.
לאחר מכן הכלי מספק כפלט מידע סטטיסטי כגון, התפלגות וריאנטים, שיעורי שגיאה שונים ותלותם בסדר הריצוף ומאפייני הספריה.
פלט נוסף הוא תיאור גרפי של תוצאות הניתוח.
אנו מדגימים את יכולותיו של הכלי על ידי ניתוח מספר ספריות מן הספרות העכשווית בביולוגיה סיננטית.
במסגרת הניתוחים הללו אנו דנים ביתרונות וברלוונטיות של הכלי ויכולותיו.
הכלי האינטראקטיבי שפיתחנו יוכל לשמש את הקהילה בהמשך מחקר בכיוון זה.

Intrinsic Autoencoders - כלי חשוב המשרת את מדעי החיים
בכדי לקבל תובנות לגבי תפקודם של תאים חיים הוא ניתוח ביטוי גנים בתאים ובאוכלוסיות. התוצאות של פרופיל ביטוי הגנים שוכנות בדרך כלל במרחב ממימד גבוה מאוד, עובדה המקשה על הסקה יעילה ומדוייקת של ניתוחים כגון סיווג וחלוקה לאשכולות.
אנו מאמינים שעבודה במרחב ייצוג הולם יותר יכולה להקל ולעזור על ניתוחים מסוג זה.
כצעד לכיוון זה אנו חוקרים את המימד האינטריסי (פנימי) של נתונים סינטטים ופשוטים. בפרט, אנו מראים כיצד ניתן להשתמש באוטו-אנקודרס לשחזור מלא של נתונים השוכנים ביריעה אשר מימדה קטן בהרבה מזה של המרחב המשכן.
אנו חוקרים שיכונים שונים והתנהגות תחת רעש.

עבודה זו בוצעה בהדרכתו של פרופ' **זהר יכיני** מבי"ס אפי ארזי למדעי המחשב, המרכז הבינתחומי, הרצליה.