UNIVERSITÉ CÔTE D'AZUR

ÉCOLE DOCTORALE
SCIENCES ET TECHNOLOGIES DE L'INFORMATION ET DE LA COMMUNICATION

THÈSE DE DOCTORAT

Stockage intelligent sur ADN synthetique pour l'archivage des images numeriques

Eva GIL SAN ANTONIO

Laboratoire d'Informatique, de Signaux et Systèmes de Sophia Antipolis (I3S)
UMR7271 UCA CNRS

Présentée en vue de l'obtention du grade de docteur en Automatique Traitement du Signal et des Images **d'**Université Côte d'Azur

Dirigée par: Marc ANTONINI, Directeur de Recherche, CNRS

Soutenue le: 31/03/2023

Devant le jury composé de:
Pascal BARBRY, Directeur de Recherche, CNRS
Thomas HEINIS, Professeur, Imperial College London
Dominique LAVENIER, Directeur de Recherche, CNRS
Athanassios SKODRAS, Professeur, University of Patras
Marc ANTONINI, Directeur de Recherche, CNRS
Anthony GENOT, Chercheur, University of Tokyo
Reinhard HECKEL, Professeur Assistant, Technical University of Munich
Olgica MILENKOVICH, Professeure, University of Illinois Urbana-Champaign
Laura CONDE-CANENCIA, Director of Strategic Collaborations and IP, CATALOG

STOCKAGE INTELLIGENT SUR ADN SYNTHETIQUE POUR L'ARCHIVAGE DES IMAGES NUMERIQUES

Intelligent storage on synthetic DNA for archiving digital images

Eva GIL SAN ANTONIO

Jury:

Président du jury

Pascal BARBRY, Directeur de Recherche, CNRS

Rapporteurs

Thomas HEINIS, Professeur, Imperial College London
Dominique LAVENIER, Directeur de Recherche, CNRS
Athanassios SKODRAS, Professeur, University of Patras

Examinateurs

Marc ANTONINI, Directeur de Recherche, CNRS
Anthony GENOT, Chercheur, University of Tokyo
Reinhard HECKEL, Professeur Assistant, Technical University of Munich
Olgica MILENKOVICH, Professeure, University of Illinois Urbana-Champaign

Directeur de thèse

Marc ANTONINI, Directeur de Recherche, CNRS

Membres invités

Laura CONDE-CANENCIA, Director of Strategic Collaborations and IP, CATALOG

Université Côte d'Azur

Eva GIL SAN ANTONIO

Stockage intelligent sur ADN synthetique pour l'archivage des images numeriques

xiii+117 p.

Ce document a été préparé avec LaTeX2e et la classe `these-ISSS` version v. 2.10.

Impression : output.tex – 1/6/2023 – 13:08

Révision pour la classe: these-ISSS.cls,v 2.10 2020/06/24 14:16:37 mpelleau

DNA neither cares nor knows. DNA just is. And we dance to its music.

Richard Dawkins

Stockage intelligent sur ADN synthetique pour l'archivage des images numeriques

Resume

La croissance rapide de la consommation de données numériques, communément appelée "l'explosion des données", présente un défi important pour le stockage des données. L'univers numérique devrait atteindre 175 zettaoctets d'ici 2025, une grande partie de ces données étant rarement consultées, mais nécessitant toujours un archivage sécurisé pour des raisons de sécurité et de conformité réglementaire. Les dispositifs de stockage conventionnels, tels que les disques durs, ont une durée de vie limitée de 10 à 20 ans, ce qui rend nécessaire de trouver des solutions alternatives pour la préservation des données à long terme qui soient non seulement rentables, mais également économes en énergie. Des études récentes ont montré que l'ADN est un candidat très prometteur pour l'archivage à long terme des données numériques. L'ADN a une capacité allant jusqu'à 215 pétaoctets par gramme et une durée de vie théorique allant jusqu'à 1 000 ans, ce qui en fait une option appropriée pour stocker de grandes quantités de données pendant des siècles, voire plus. Cependant, le processus d'encodage des données numériques dans un flux quaternaire composé des symboles A, T, C et G, qui représentent les quatre composants de la molécule d'ADN, tout en respectant d'importantes contraintes d'encodage, fait l'objet de recherches en cours. Des travaux pionniers ont proposé différents algorithmes pour le codage de l'ADN, mais des améliorations sont encore possibles. Dans ce contexte, une nouvelle génération de séquenceurs utilisant des nanopores offre la possibilité de lire des brins d'ADN beaucoup plus rapidement et à moindre coût, avec l'inconvénient d'un taux d'erreur plus élevé. Cette thèse porte sur l'étude de ces erreurs afin d'adapter et de rendre encore plus robuste le codage quaternaire des données. De plus, des techniques de post-traitement adaptées au contexte de stockage des données ADN sont proposées pour corriger les erreurs restantes après décodage. Les résultats d'expériences en laboratoire sont présentés dans lesquels diverses images ont été stockées dans l'ADN à l'aide de différentes méthodes de codage et séquencées à l'aide de différentes technologies telles que Illumina et nanopore. Nous présentons une étude des erreurs introduites avec chaque plate-forme de séquençage et la robustesse des différentes solutions de codage testées expérimentalement. L'objectif de cette recherche est de contribuer au développement de systèmes efficaces et fiables de stockage d'archives sur ADN.

Mots-clés: Stockage de données ADN, Correction des erreurs, Décodage robuste.

Intelligent storage on synthetic DNA for archiving digital images

Abstract

The rapid growth of digital data, commonly referred to as the "data explosion," presents a significant challenge for data storage. The digital universe is projected to reach 175 zettabytes by 2025, with a large portion of this data being infrequently accessed, yet still requiring safe archival for security and regulatory compliance reasons. Conventional storage devices, such as hard drives, have a limited lifespan of 10-20 years, making it necessary to find new solutions for long-term data preservation that are not only cost-effective, but also energy-efficient. Recent studies have shown that DNA is a very promising candidate for the long-term archival storage of digital data. DNA has a capacity of up to 215 petabytes per gram and a theoretical lifespan of up to 1,000 years, making it a suitable option for storing large amounts of data for centuries or even longer. However, the process of encoding digital data into a quaternary stream made up of the symbols A, T, C and G, which represent the four components of the DNA molecule, while also respecting important encoding constraints, has been a subject of ongoing research. Pioneering works have proposed different algorithms for DNA coding, but there is still room for further improvement. In this context, a new generation of nanopore-based sequencers offers the possibility of reading DNA strands much faster and cheaper, with the disadvantage of a higher error rate. This thesis focuses on the study of the nature of such errors in order to further adapt and robustify the encoding of the data into a quaternary code and ensure its decodability. Additionally, post-processing techniques adapted to the context of DNA data storage are proposed to correct the remaining errors after decoding. We also present the results of a wet-lab experiment in which various images were stored in DNA using different encoding methods and sequenced using different technologies such as Illumina and nanopore. We provide a study of the errors introduced with each sequencing platform and the robustness of the different encoding solutions tested in the wet-lab experiment. The goal of this research is to contribute to the development of efficient and reliable DNA-based archival storage systems.

Keywords: DNA data storage, Error correction, Robust decoding.

Acknowledgements

A PhD is like a rollercoaster with the difference that the ride lasts for a few years rather than a couple of minutes (and that there isn't cotton candy). The "journey" takes you up, down, forward, backwards... and sometimes you just find yourself upside down. More often that we'd like, it makes you suffer, but it also brings extremely rewarding moments which make it all worth.

I want to start by thanking my supervisor Marc Antonini for offering me the opportunity work on such an amazing topic and guide me along the process. A very special thanks also to Pascal Barbry, Raja Appuswamy and their colleagues in IPMC and EURECOM for all the interesting discussions and collaborations, as well as all the partners from the OligoArchive project.

Secondly, thanks to all the amazing people I met during my 4 years in I3S laboratory, starting by Melpomeni, who has been right next to me since the very first moment and has taught me what it means to be committed, passionate and hardworking. I would have never believed that we would go through so much together if someone had told me the day when I first met you. Also thanks for bringing Dimitris into my life, I love you guys and I'm extremely happy and proud to see the beautiful family you are creating. To Xavier, for all his help during these last months and for becoming one of the reasons why I love coming to the office. To Nadia, for being kind, caring and encouraging. Since my first day in the lab, she made sure that I was ok, always willing to help even when it wasn't her job. If it wasn't because we look like sisters, I'd even say that you have been the mum I never had in France. To Andrew, André and Jean, without whose help I'd have finished the PhD 1 year earlier. To Fernando, who was not only my colleague, but also my roommate (I didn't know a human being could eat so much until I met you). Finally, to Vasilina and Diana, my angel and my demon, who always made bad days more bearable, and good days even better.

I also want to give a very big thanks to the people I've shared with most of my time during this PhD : Melissa, for teaching me the meaning of being "strong like a woman" ; Miguel, for always finding a reason to laugh, even when everything seems to be falling apart ; Thomas, for babysitting me every time I needed it ; and Lucy, whom I consider a sister today. You guys are my little family from everywhere, and I will never be able to pay you back for everything you did for me these years.

Thanks to all my Spanish friends. Leaving your home and moving somewhere else isn't always easy, but they proved day after day that no matter how far we are, we can always count on each other. María, Lucía, Óscar, Bea, Juan, Gonzalo, Nuria, Andrei, Ortiz, Huisman, Mais, Marc, Badal, Edith, Patri... It was never a "goodbye", it just was a "see you later".

To my family, Jose, Elena, Irene and Joey. They were part of this adventure since the very beginning, when they drove all the way to Nice from Valencia so I wouldn't have to do it alone. They have celebrated my victories as their own as well as they have suffered my pain as their own. Having a strong and caring family like you means everything to me. To Blanqui and César, who have always been by my side and didn't think twice during my first month here to come and visit my new home.

Last but not least, I want to thank Ale. You unexpectedly appeared one day and never left. You are the breath of fresh air that I had been waiting for. Thanks for having my back, and thanks for being the extraordinary person that you are.

Contents

CHAPTER 1

Introduction

1.1 Digital storage crisis

Digital media explosion has led to an exponential increase of the amount of data generated worldwide. Every day, around 2.5 exabytes (1 EB = 10^{18} bytes) are generated, which would be the equivalent to 10,000 billion photos taken with a smartphone. The total amount of data created, copied and consumed globally is forecast to increase rapidly, and it is expected to grow up to more than 180 zettabytes (1 ZB = 10^{21} bytes) by 2025 (Taylor, 2021). With the current supply of data storage mediums only 20% of the generated data could be stored by then.

The continuous expansion of the global datasphere challenges the usage of traditional storage media. One of the main problems is the density limit. If mainstream media keeps being used, soon there will be no physical space to store the data. To illustrate the challenge, if we were to store the world's digital data on DVDs, we would need a stack of DVDs that could reach the moon 23 times or circle Earth 222 times. In order to keep up with the storage needs, more than 8 million data centers have been built so far, up from approximately half a million in 2012. The construction of data centers entails extremely high investments on storage systems (mostly HDD, SSD or tape) as well as a complex infrastructure for security, power supply and environmental controls to guarantee optimal conditions in the building. Additionally to the high costs associated to the construction, maintenance, climate control (i.e. cooling systems) and hardware replacements, data centers have a rather negative environmental impact. Currently, data centers generate as much carbon emissions as the global airline industry and its energy consumption doubles every 4 years, with some models estimating that it could reach 10% of the global electricity supply by 2030. A single medium-sized data center uses as much water as 3 average-sized hospitals and 20 to 50 million metric tons of server, storage and networking equipment (called E-waste) are thrown away worldwide every year (McNerney, 2019).

Interestingly enough, according to the International Data Corporation (IDC) * between 60% and 80% of this data is either inactive or infrequently accessed (generally referred to as "cold") but it still has to be preserved, traditionally to comply with security and regulatory compliance reasons but also because of its long-term strategic value.

Today this data is mostly stored into magnetic media (tape, hard disks), solid state media (flash drives) or optical media (CD-ROMs) which despite having proven to be an optimal solution during the past decades, induce high-energy costs, and have a limited life-span which varies from 3-5 years for HDD drives to 20-30 years for back-up tape drives, as they are prone to mechanical failure, damage due to temperature, or damage due to magnetic fields. As a result, data has to be migrated to new storage units every few years to ensure its reliability.

*. https://www.idc.com/about

It was in 1959 when Richard Feynman suggested for the first time the use of DNA as replacement for mainstream media but it was not until the past decade that this research field started attracting great interest due to the biochemical properties of DNA, which make it a very promising candidate for the task. First, it is an extremely dense three-dimensional storage medium that has the theoretical ability to store 1 Exabyte/mm^3 while in the case of tape (currently the densest commercial storage medium) is around 4.6 Gigabytes/mm^3 (Nguyen et al., 2020). In other words, DNA offers a density which is 200 million times higher than current media. Second, DNA can last several centuries even in harsh storage environments (the oldest DNA record found ages back 1.6 million years and it is still readable) (Callaway, 2021). And third, it is very easy, quick, and cheap to perform in-vitro replication of DNA while tape and HDD have bandwidth limitations that result in hours or days for copying large EB-sized archives. Currently, the main obstacle faced by DNA data storage lies on the biochemical processes of DNA writing and reading, which remain expensive and time-consuming. However, the rapid technological advances in the field portray a promising future for DNA archival systems.

1.2 DNA as storage medium

DNA (Deoxyribonucleic acid) is the molecule that carries the genetic information in living beings. It constitutes the hereditary material that contains the necessary information for the development, functioning and reproduction of organisms. DNA is a polymer composed of two strands called polynucleotides that attach together in a spiral forming a double helix. Each polynucleotide is formed by simpler monomeric units, the nucleotides (nts) or bases, which are composed of a sugar (deoxyribose), a phosphate group and a one of the four nucleobases (Cytosine [C], Guanine [G], Adenine [A] or Thymine [T]). The nucleotides are joined to one another in a chain by covalent bonds between the sugar of one nucleotide and the phosphate of the next, resulting in an alternating sugar-phosphate backbone. If one thinks about the sugar as a puzzle piece, it would have a knob (the 5' phosphate) in one side, and a hole (the 3' hydroxyl) on the other. Then, the polynucleotide would be formed by coupling knobs and holes and the resulting chain will have all its nucleotides aligned in the same orientation, giving the DNA strand a chemical polarity indicated by referring to one end as the 3' end and the other as the 5' end. The order in which the bases are arranged to form the polynucleotide determines the information available for building and mantaining an organism. The two strands are linked by chemical bonds between the nucleobases, which follow a complementary base pairing rule: adenine always pairs with thymine and guanine with cytosine. Figure 1.1 depicts the structure of DNA and its building blocks.

DNA possesses some biological properties that make it a very promising candidate to replace current storage media:

— **Information Density.** Theoretically, DNA can store 1 $Exabyte/mm^3$ which is eight orders of magnitude higher than tape.
— **Longevity.** DNA can be preserved without major alterations for thousands, even millions, of years if stored under the right conditions.
— **Easy replication.** In-vitro replication of DNA is easy and quick without the bandwidth limitations of HDD and tape.
— **Reliability.** Current storage media has a life-span of five to twenty years, meaning that data has to be migrated to new devices to ensure its reliability or to deal with technology

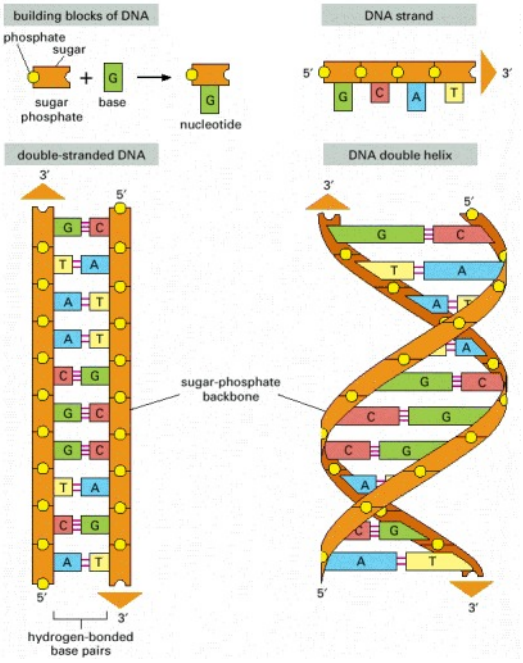


Figure 1.1 – The structure of DNA. Source: (Alberts et al., 2002).

upgrades. In the case of DNA, once the data has been stored it can be left untouched as we can be sure that it will never suffer from obsolescence.

— **Relevance.** DNA is the support of heredity in living organisms. As long as there is life on earth, there will always be the need to synthesize and sequence DNA.

— **Sustainability.** Contrary to mainstream media, the production of DNA does not involve heavy metals or other rare elements harmful for the planet and can be safely stored with no need for climate control or rewriting which significantly decreases the total energy demand.

— **Biodegradability.** No harm is caused to the environment when the DNA is discarded.

Although the aforementioned assets of DNA make it suitable for data storage, the biochemical processes involved in the storage channel introduce some challenges that cannot be ignored when designing a DNA-based archival storage system. These constraints will be discussed in section 2.2.

1.3 General workflow

Roughly, the general workflow for DNA data storage can be described as depicted in figure 1.2. Any kind of input data can be stored into DNA as long as it is encoded first into a quaternary representation using the 4 symbols of the DNA (A, C, T and G). The encoded sequence is formatted into several shorter fragments or oligos which will contain an address indicating their position in the initial sequence to be able to reconstruct it. The short oligos are then biologically synthesized in a lab and stored (e.g. in capsules under a protective atmosphere or embedded in a protective material to further improve its storage potential). Whenever the stored data needs to be retrieved, the DNA molecules are read using some special devices called sequencers. However, the

biochemical processes of synthesis and sequencing are prone to errors, which makes it necessary to first increase the redundancy of the data through a process called Polymerase Chain Reaction (PCR) to allow error correction. The amplification step generates many copies of the original oligos which will then be sequenced. Once sequenced, the noisy reads must be post-processed to obtain the most representative sequence of each oligo. This post-processing involves a clustering step in which all the copies from the same reference oligo are grouped together and a consensus step in which the representative sequence of each cluster is inferred. Finally, the initial data can be decoded from the consensus sequences by following the inverse process of the encoding. Even so, the biochemical processes of DNA synthesis and sequencing are complex and introduce some major constraints, adding extra complexity to the encoding and decoding steps which will be further discussed in the coming sections.

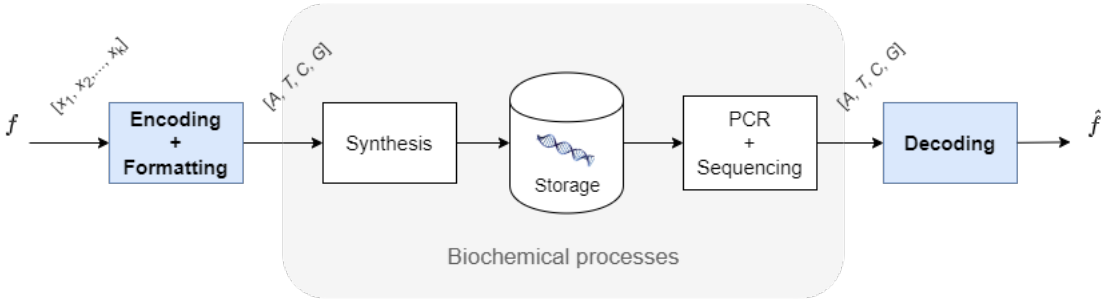


Figure 1.2 – General overview of DNA storage process.

The rapid evolution of sequencing technologies has brought DNA data storage closer to reality, with an almost constant improvement of the accuracy and cost-effectiveness of the process during the past years. Readers should keep in mind that the error rates used for the simulations in the different experiments presented in this thesis, were selected according to the best noise levels achieved up to the date when the experiments took place.

1.4 OligoArchive

This thesis is part of the OligoArchive European Project †, a three year project with the objective of developing both, theoretical foundations and systems technology that can overcome all the challenges derived from the current limitations of the biochemical processes involved in the DNA data storage channel, aiming to make DNA an intelligent storage medium that can serve as a drop-in replacement for today's storage media. The consortium comprises Imperial College London (UK), EURECOM (France), CNRS (I3S and IPMC, France), and Helixworks Technologies (Ireland) leading in the area of DNA synthesis and DNA data storage.

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 863320.

1.5 Contributions

The different contributions presented in this thesis are organised as follows:

†. https://oligoarchive.eu/

In chapter 3, we propose an alternative encoding schema for the JPEG-inspired DNA coder developed by the Mediacoding group [‡] to robustify the code against sequencing errors. This work proposes to encode the DC coefficients using Vector Quantization (VQ) and use a previously proposed noise resistant mapping (NoRM) to reduce the impact of substitution errors on the decoded data while also ensuring its decodability. Also in chapter 3, we propose a method for the automatic detection of errors and correction with inpainting, adapted to the wavelet domain, to correct persistent errors after the decoding.

In chapter 4, we propose a decoding workflow and a consensus algorithm for non-complete dictionaries to be able to recover information stored in DNA and sequenced with MinION. We also use the error rates from the just mentioned experimental data to validate a new DNA data storage channel simulator.

In chapter 5, we show and analyse the results of a wet-lab experiment carried out with the goal of testing the robustness of various encoding solutions for the storage of digital images into DNA. We compared fixed-length and variable-length solutions, as well as simple transcoding of the binary output of JPEG and sequenced the data using various sequencing platforms. We also extract the noise statistics and combine it with the decoding results to draft some conclusions regarding the robustness for the tested encodings as well as the reliability of the tested sequencers.

Finally, in chapter 6, we establish the theoretical foundations for a novel encoding schema addressing oligonucleotide synthesis by the enzymatic ligation of motifs, which holds promise for being cheaper and faster in the future.

1.6 Publications and other achievements

1.6.1 Journals

"A JPEG-inspired coder for DNA coding robust to sequencing noise." To be submitted to IEEE transactions on Image Processing.

1.6.2 Conference papers

Dimopoulou, M., Gil San Antonio, E., & Antonini, M. (2020). A quaternary code mapping resistant to the sequencing noise for DNA image coding. In 2020 IEEE 22nd International Workshop on Multimedia Signal Processing (MMSP) (pp. 1–6).

Dimopoulou, M., Gil San Antonio, E., & Antonini, M. (2021a). A JPEG-based image coding solution for data storage on DNA. In 2021 29th European Signal Processing Conference (EUSIPCO) (pp. 786–790).

Dimopoulou, M., Gil San Antonio, E., & Antonini, M. (2021b). A sequencing noise resistant code mapping algorithm for image storage in DNA. In CORESA 2020.

Dimopoulou, M., Gil San Antonio, E., Antonini, M., Barbry, P., & Appuswamy, R. (2019). On the reduction of the cost for encoding/decoding digital images stored on synthetic DNA. In GRETSI 2019.

‡. https://www.i3s.unice.fr/~fpayan/mediacoding/

Gil San Antonio, E., Dimopoulou, M., Antonini, M., Barbry, P., & Appuswamy, R. (2021). Decoding of nanopore-sequenced synthetic DNA storing digital images. In 2021 IEEE International Conference on Image Processing (ICIP).

Gil San Antonio, E., Heinis, T., Carteron, L., Dimopoulou, M., & Antonini, M. (2021). Nanopore sequencing simulator for dna data storage. In 2021 International Conference on Visual Communications and Image Processing (VCIP) (pp. 1–5).

Gil San Antonio, E., Piretti, M., Dimopoulou, M., & Antonini, M. (2021). Robust image coding on synthetic DNA: Reducing sequencing noise with inpainting. In 2020 25th International Conference on Pattern Recognition (ICPR) (pp. 9859–9865).

1.6.3 Other achievements

— Best student paper award at CORESA (2020) for "A sequencing noise resistant code mapping algorithm for image storage in DNA".

— Winner in "3 Minute Thesis" contest EURASIP/EUSIPCO (2021)

— Prize of excellence from the Université Côte d'Azur for the year 2021 (Prix d' excellence UCA 2021)

— Finalist in the 5th edition of "Prix Pierre Laffitte" (2021)

— Member of DNA Data Storage Alliance.

— JPEG DNA expert in the JPEG-DNA Ad Hoc Group for the development of a standard for the storage of digital images and videos into DNA. Contributions:

1. DNA-based media storage: State-of-the-art, challenges, use cases and requirements version 7.0.
2. Nanopore Sequencing Simulators - State of the art, wg1m92077-REQ

Notations

List of Abbreviations

AC	Alternating Current
ACD	Advanced Correction Decoding
DC	Direct Current
DCT	Discrete Cosine Transform
dNTP	deoxyribonucleotide triphosphate
DMT	Dimethoxytrityl
DNA	Deoxyribonucleic Acid
DO	Data Oligo
DWT	Discrete Wavelet Transform
FO	Frequency Oligo
Gb	Gigabase
GO	Global Oligo
HMM	Hidden Markov Model
IDC	International Data Corporation
indels	insertions and deletions
JPEG	Joint Photographic Experts Group
LSH	Locality Sensitive Hashing
MLP	Multilayer Perceptron
MO	Mapping Oligo
MSE	Mean Square Error
NGS	Next Generation Sequencing
NoRM	Noise Resistant Mapping
nt(s)	nucleotide(s)
PCR	Polymerase Chain Reaction
PDF	Probability Density Function
PNG	Portable Network Graphics
PSNR	Peak Signal-to-Noise Ratio
R2C2	Rolling Circle Amplification to Concatemeric Consensus
RCA	Rolling Circle Amplification
RD	Rate-Distortion
SBS	Sequencing By Synthesis
SO	Subband Oligo
SSIM	Structural Similarity
VQ	Vector Quantization

List of Symbols

a_{sb}	fraction of total pixels of subband sb
$\mathcal{B}$	barcode set
B	set of blocks of quantization elements
c_e	erroneous codeword
$\mathcal{C}^*$	constrained codebook provided by PAIRCODE
$\mathcal{D}_b$	block dictionary
d_E	Euclidean distance
d_L	Levenshtein distance
D	distortion
$\mathcal{D}$	set of all possible viable quaternary words composed by the symbols $\{A, T, C, G\}$
D_{sb}	distortion of subband sb
D_T	global distortion
$F(v)$	empirical function used to compute the density of the neighborhood of a vector v
$H(c_k)$	Hamming sphere of c_k containing words with Hamming distance of 1 to c_k.
$\mathcal{I}$	set of images
$\mathcal{I}_q$	set of quantized images
I	image
I_q	quantized image
K	number of indices in the codebook
L	number of codewords in the code
l	length of codewords in the code
$\mathcal{M}$	motif dictionary
m	motif
N_p	1-ring neighborhood of pixel p
O	binary mask
$p(v)$	probability of a vector v
P_{sb}	number of pixels in subband sb
Q_{sb}	set of quantization parameters for subband sb
R_{sb}	rate of subband sb
R_T	global rate
R_{target}	target rate
$S(v_k)$	neighborhood containing the closest vectors around vector v_k
$\mathcal{S}_{dec}$	set of quaternary words of length l
S_E	set of eligible combinations of subsets X_S to encode tile X
SB	number of subbands
sb	subband
$\mathcal{V}$	Set of input source symbols
w	undecodable word
X	2x2-sized tile from a quantized image I_q
X_S	subset from tile X
Γ	mapping function for VQ-based encoding
Γ_b	mapping function for motif-based encoding
λ	Lagrange multiplier (slope of the R-D curves)
$\mu_{\mathcal{B}}$	maximum tolerance of barcode set $\mathcal{B}$
σ	standard deviation
Σ	set of indices
ω_{sb}	weights of non-orthogonality used in source allocation

CHAPTER 2

State of the art

2.1 Biochemical processes involved in the DNA data storage channel

2.1.1 Synthesis methods

Oligonucleotide synthesis is the chemical fabrication of DNA sequences. There are different types of DNA synthesis which use different chemistry and/or methodology. Most of current biological research and bio-engineering involves synthetic DNA. Oligos are synthesized using monomers that replicate the natural bases.

DNA synthesis methods can be divided into two main categories: base-by-base synthesis and synthesis by ligation. The main difference between these methods is that the first one constructs the DNA strands by adding one base at a time while the second method is based on concatenating short pre-defined sequences of few bases each. While base-by-base synthesis, more concretely Phosphoramidite chemistry, is the most used technique for oligonucleotide synthesis, the length of the constructed strands is limited, an issue that could be overcome by ligation methods. The following paragraphs describe briefly the different chemistries for DNA synthesis.

2.1.1.1 Phosphoramidite method for oligonucleotide synthesis

Phosphoramidite chemistry, also referred to as chemical synthesis, was first described in 1981 by Serge Beaucage and Marvin Caruthers (Beaucage & Caruthers, 1981) and it is one of the most widely used methods for DNA synthesis. Phosporamidites are derivatives of natural or synthetic nucleosides and they are used to build DNA strands by adding bases one by one.

The commonly used phosphoramidite synthesis chemistry is the **column-based oligonucleotide synthesis** (figure 2.2A). The process occurs on a solid support made of Polystyrene or Controlled Pore Glass. The support is inside a reaction vessel (or column) that has filters on either side to allow the solutions to pass through but keep the solid material trapped in between. The synthesized molecules grow attached to this support. Each cycle of the process can be described in four steps:

1. **Detritylation (or de-blocking).** The phosphoramidite nucleosides attached to the support contain a Dimethoxytrityl (DMT) blocking group which prevents the 5' end from reacting. In this step, this blocking group is removed by adding an acid solution (typically Trichloroacetic acid) to allow the binding of new bases.
2. **Coupling.** After the DMT blocking group is removed, the 5' is available for adding the next nucleotide (A, T, C or G). The selected nucleoside is then mixed with an activator agent (Tetrazole or Ethylthiotetrazole) which protonates the 3' end of the nucleoside to facilitate the reaction with the 5' end of the oligo attached to the support.

3. **Capping.** Coupling reaction is not 100% effective and there will be a few unreacted 5'-hydroxyl groups on the oligos growing on the support. After washing the solution to remove the acid, these 5' ends that did not get the nucleoside are blocked with a mixture of capping solutions to prevent the extension of the DNA sequences.
4. **Oxidation.** The sugar phosphate backbone of the attached base is stabilized and strengthened through iodine oxidation in order to stabilise the bond.

Oligonucleotide synthesis is done via a cycle of these four chemical reactions that are repeated until all desired bases have been added. Once the synthesis is finished, the new oligos are separated from the solid support. Figure 2.1 depicts the different stages of phosphoramidite synthesis.

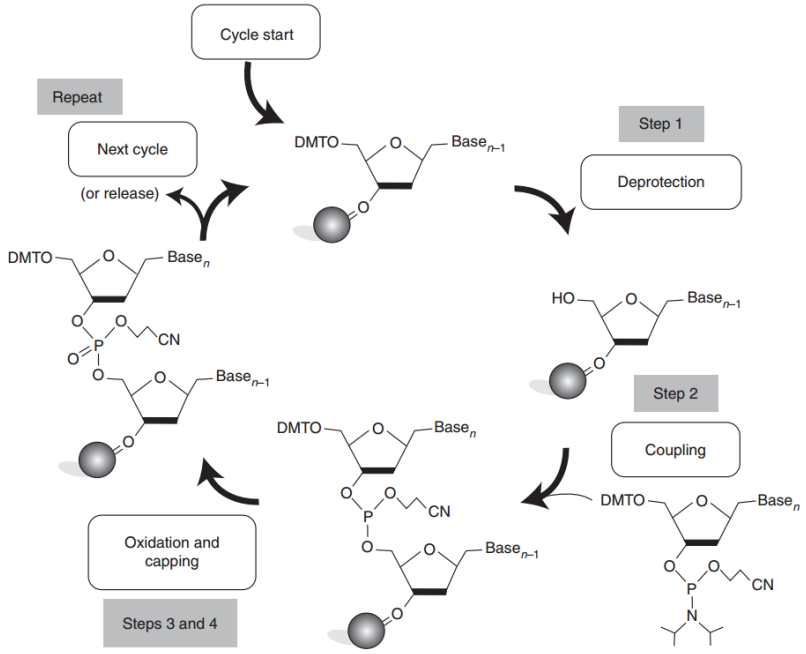


Figure 2.1 – Phases of phosphoramidite method for oligonucleotide synthesis. Source: (Hughes & Ellington, 2017).

This process can be easily automated and forms the basis for oligonucleotide synthesizers.

An alternative to traditional column-based oligonucleotide synthesis has emerged from the use of **micro-array oligonucleotide synthesis** (figure 2.2B). This method allows for a higher synthesis throughput, increasing the amount of different oligos synthesized in a single run. Additionally, micro-array oligonucleotide synthesis requires the use of less reagents, which account for a large percentage of the synthesis cost, dramatically reducing the costs.

2.1.1.2 Enzymatic synthesis

In the past decade, an increasing amount of focus has been placed on enzymatic DNA synthesis, which follows a cyclic process similar to phosphoramidite synthesis in which nucleotides are sequentially added to construct the oligos. The main difference lies on the use of an enzyme to catalyze the synthesis reaction as it happens in living organisms. This new way of DNA synthesis avoids the use of harsh chemicals replacing them by aqueous reagents that do not produce

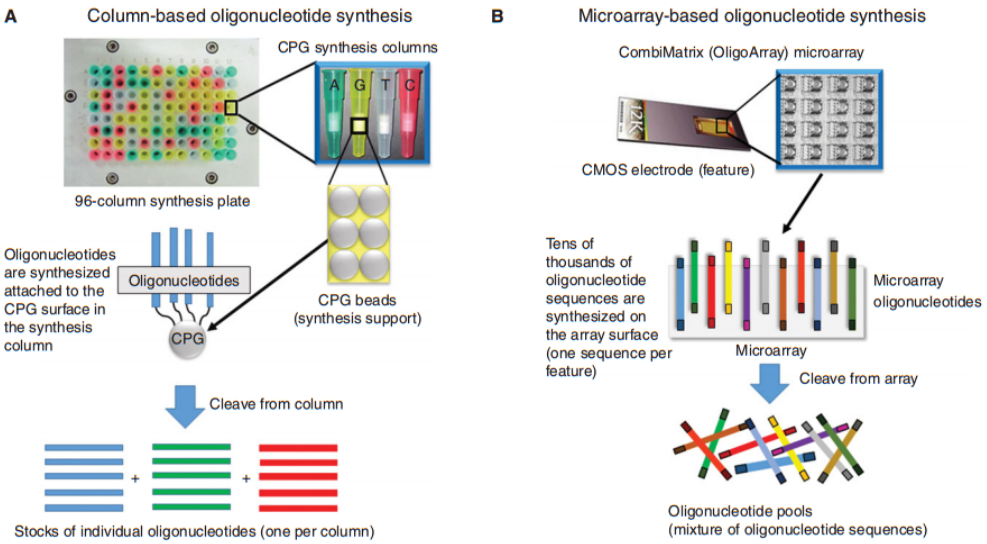


Figure 2.2 – Methods for solid-phase synthesis of oligonucleotides. (A) Column-based oligonucleotide synthesis. (B) Microarray-based oligonucleotide synthesis. Source: (Hughes & Ellington, 2017).

toxic waste, providing a more sustainable way for synthesizing DNA. Other important assets of enzymatic synthesis include speeding up the process to increase the throughput and its ability to generate longer oligonucleotides. In 2022, the french company DNA Script launched the first DNA printer powered by enzymatic DNA synthesis *.

Figure 2.3 depicts the process of enzymatic DNA synthesis. It occurs in a 2-step process:

1. **Elongation.** The enzyme adds a nucleotide to the DNA molecule
2. **Deprotection.** The reversible terminator of the nucleotide is removed, leaving the strand ready to be elongated again.

Steps 1 and 2 are repeated until the longest oligo on the plate is completed.

2.1.1.3 Synthesis by ligation

The synthesis methods previously described construct the oligos by adding a single base to the growing strand in each cycle. The main challenge that comes with this approach is that each step in the synthetic cycle must have very high yield in order to obtain a final product in the required amount with a low accumulated error rate. This constitutes a major limitation in the length of the sequences that can be built using these methods (200-300 nts) as the longer the synthesized strand, the lower the yield that can be obtained.

A solution that could overcome the aforementioned challenge is synthesis by ligation (Borodina, Lehrach, & Soldatov, 2003). In few words, this method is based on the construction of DNA strands by binding predefined short oligonucleotides (DNA strands of few nucleotides length). This method involves two main steps: creation of the dictionary and ligation. The first step refers to the creation of the dictionary containing all the predefined short sequences, which could be seen as the construction of the building blocks. These short words are synthesized by

*. https://www.dnascript.com/products/syntax/

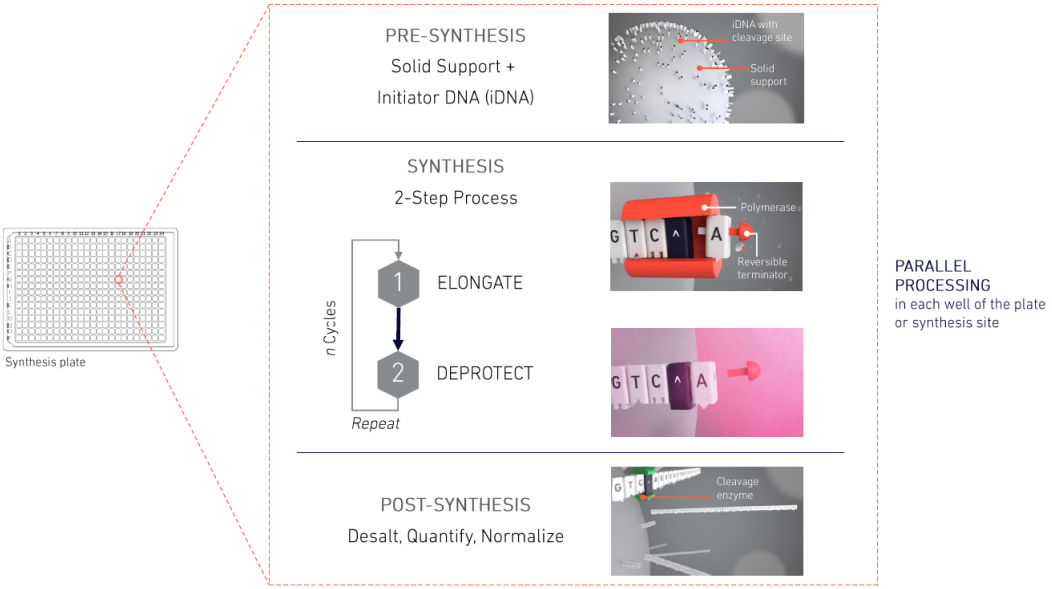


Figure 2.3 – Process of enzymatic DNA synthesis. Source: (DNAScript, 2022).

some base-by-base method such as phosphoramidite or enzymatic synthesis. However, the construction of the dictionary is a one-time task: once built, the short DNA words can be replicated using PCR (see section 2.1.2) to create more copies, process that is easy, quick and cheap. Once the dictionary is ready, the oligos can be created by binding the different words into longer sequences. The ligation of the short sequences is performed with DNA ligase - an enzyme that facilitates the joining of the DNA strands together. Figure 2.4 depicts an example of synthesis by ligation. This method offers numerous advantages such as the reduction of the cost in the long term, higher speed and the possibility of constructing longer DNA sequences compared to the aforementioned methods.

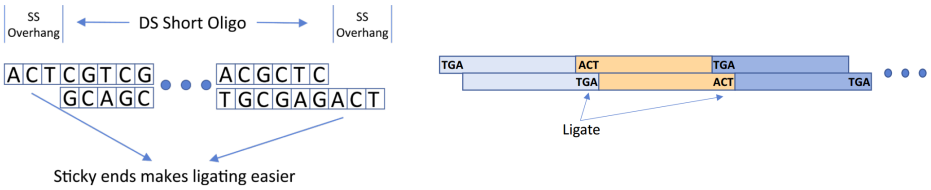


Figure 2.4 – Example of synthesis by ligation. Left: Dictionary of double-stranded short oligonucleotides with single-stranded overhangs. Right: Short oligonucleotides are combined into longer one by the natural pairing of the complementary bases in the overhang segments. DNA ligase makes these bonds permanent. Source: (Alliance, 2021).

2.1.2 Amplification with Polymerase Chain Reaction (PCR)

Thanks to the complementary pairing of the bases, both of the strands forming a helix contain the same information. When the double helix "unzips", one of the strands can be used to replicate its complementary one in a fast and easy way through a cloning process called Polymerase Chain Reaction. Briefly, each cycle of the process can be described in three phases:

1. **Denaturation**. The sample is heated up to split the double helix into two single-stranded DNA.
2. **Annealing**. Primers are bound to the single-stranded DNA complementing the 3' end. A primer is a short sequence of nucleotides that provides the starting point for DNA synthesis.
3. **Elongation**. An enzyme called DNA Polymerase synthesizes the strand complementary to the template.

These steps are repeated until the desired amount of copies is achieved. Figure 2.5 depicts the PCR process.

PCR provides exponential amplification, doubling the amount of product at every cycle (after 30 cycles, one copy can be increased up to 1 billion). For this reason, PCR plays an important role in the context of DNA data storage as it can be used to increase the redundancy of the data in an easy and cheap way, allowing for error correction. In (Dimopoulou, Gil San Antonio, Antonini, Barbry, & Appuswamy, 2019) we show how the quality of the decoded data is correlated to the amplification rate of the DNA molecules before sequencing.

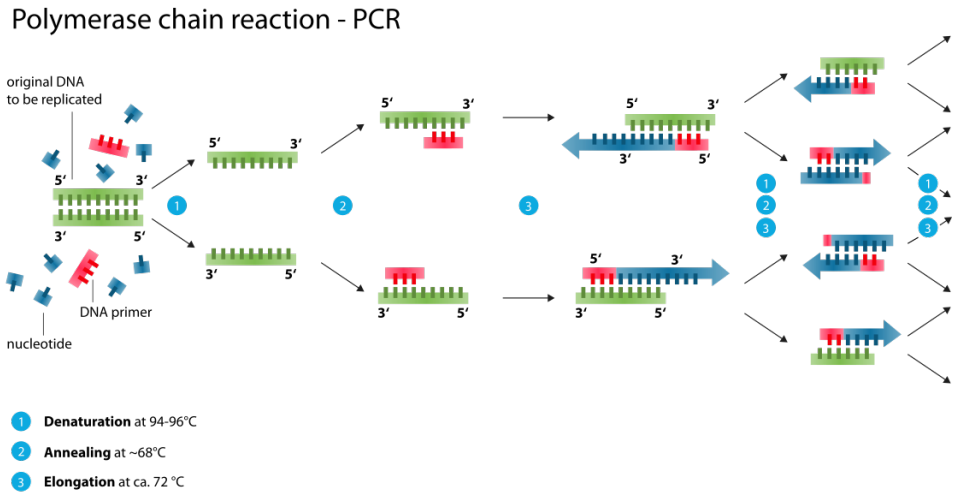


Figure 2.5 – Phases of Polymerase Chain Reaction. Source: Wikimedia. Figure under creative commons license.

2.1.3 Storage

Despite DNA being one of the most stable molecules known, the storage conditions are key if one wants to make sure to preserve its integrity during the storage period. There are several options to store DNA: suspended in an aqueous solution, refrigerated or frozen, dehydrated and embedded into silk matrices or salts, etc.

A particularly interesting solution when talking about long-term storage is the encapsulation of DNA into very small sealed containers (see figure 2.6) that protect DNA from alteration factors such as water, oxygen or light by preserving the purified and desiccated DNA molecules under an inert atmosphere.

All this methods present advantages and drawbacks. Thus, the selection of one of them depends on various factors: the type of DNA to be stored, the duration of storage, the storage tem-

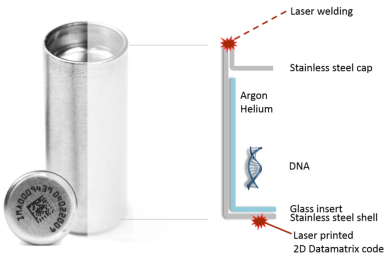


Figure 2.6 – The DNAshell® minicapsule, consisting of a stainless steel case and a stainless steel cap, hermetically sealed by laser welding. Source: (Imagene, s. d.).

peratures and conditions and the downstream applications for which the DNA is to be used. This is further described in section 2.2.

2.1.4 Sequencing methods

DNA sequencing consists on determining the sequence of nucleotides in a DNA molecule. The first sequencing methods were introduced in the decade of the 70s with the pioneer works of Sanger (Sanger, Nicklen, & Coulson, 1977) among others, which allowed the development of the first automated DNA sequencer in 1986. Since then, the progress continued and mostly over the last two decades the advances in nanotechnology and informatics have contributed to the new generation of sequencing methods. Figure 2.7 shows some of the most important advances in DNA sequencing.

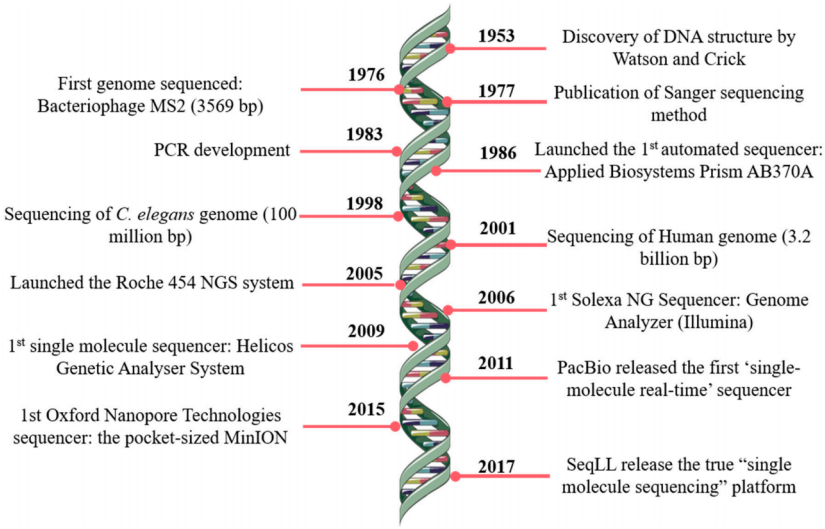


Figure 2.7 – DNA sequencing timeline including some of the most revolutionary and remarkable events in DNA sequencing (Pereira et al., 2020).

The description of all the sequencing technologies developed during the past decades is out of the scope of this work. Instead, we will define the two most relevant sequencing strategies for the work presented in this thesis: sequencing by synthesis and nanopore sequencing.

2.1.4.1 Sequencing by synthesis

Illumina [†] (see figure 2.10 top left) is currently one of the most accurate sequencing devices, with an overall error rate of about 1%. It is based on Sanger sequencing (Sanger et al., 1977) with some modifications which allow to massively parallelize the process to achieve high throughput. Its main limitations are the read length, which ranges from 50 to 300 nts, and the high instrument costs.

Illumina sequencing is based on the idea of sequencing by synthesis (SBS), a cyclic process which comprises the following steps:

1. **Library preparation.** DNA sample is fragmented into chunks of 200-500 nts in the case of biological samples (not needed for short synthetic oligos) and adapters ligated to the 5' and 3' ends. Adapter-ligated fragments are then PCR amplified and gel purified to finish the construction of the the sequencing library.
2. **Cluster generation.** The fragments from the sequencing library are then attached to the surface of the flow cell, a thick glass slide with lanes coated with oligos complementary to the library adapters. The attached strands are cloned using bridge PCR, creating clonal clusters (thousands of copies of the same strand positioned closely together).
3. **Sequencing.** DNA polymerase, connector primers and 4 dNTP (deoxyribonucleotide triphosphate) with base-specific fluorescent markers are added to the reaction system. The DNA polymerase catalyzes the incorporation of the fluorescently labeled nucleotides into the DNA templates. When the binding occurs, they emit light which is imaged and used to identify the base. This step determines the sequence of nucleotides in the DNA template, one base at a time.

The most common errors in Illumina-sequenced data are substitutions. This is due to the strong intensity correlation of the pairs A and C as well as T and G as a result of a similar emission spectra of the fluorophores.

Figure 2.8 depicts the Illumina sequencing process.

2.1.4.2 Nanopore sequencing

Nanopore sequencers from Oxford Nanopore Technologies (ONT) [‡] offer numerous advantages compared to Illumina, allowing the sequencing of long reads (up to hundreds of thousands of bases) in a much cheaper and faster way. The main drawback of this sequencing approach is that it provides a lower accuracy. However, the further development and improvement of this technology during the past years have yielded to a $> 99\%$ accuracy.

Nanopore-based sequencers identify DNA bases by measuring the changes in electrical conductivity generated as DNA strands pass through a biological pore. Biological nanopores are transmembrane protein channels embedded in a matrix (i.e. lipid bilayers, liposomes or other polymer films) that are naturally produced by bacteria and can be genetically engineered by using molecular biology techniques, allowing for DNA sequencing and protein detection (Magi, Semeraro, Mingrino, Giusti, & D'aurizio, 2018). The workflow of this device can be divided into 3 main steps:

†. https://www.illumina.com/systems/sequencing-platforms/nextseq.html
‡. https://nanoporetech.com/products

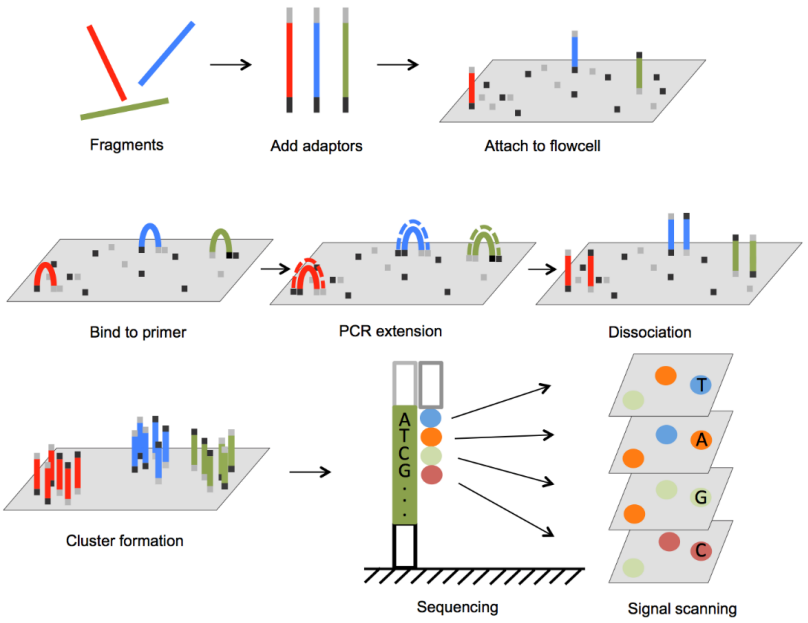


Figure 2.8 – Principle of sequencing by synthesis (Illumina). Source: (PraxiLabs, s. d.).

1. **Sample preparation.** The DNA sequences are prepared by adding to its ends two adapters that guide the fragments to the pores.
2. **Signal measurement.** In an electrolytic solution, a constant voltage is applied to produce an ionic current through the nanopore such that the single-stranded DNA molecules are driven through. Then, a sensor measures the ionic current changes with a constant sampling frequency, generating electrical signals.
3. **Basecalling.** The raw data (electrical signals) is processed using machine-learning techniques into basecalled data (the sequence of DNA bases). In other words, the electrical signals are translated into sequences of nucleotides. This procedure is carried out using k-mer tables that translate sequentially fragments of the electrical signals into sets of k nucleotides.

The recent advances in the chemistry and software of nanopore devices have extended the scope Oxford Nanopore Technology to not only long reads but also short fragments down to 20 bases in length. Figure 2.9 depicts the basic functioning of nanopore sequencing.

ONT offers various sequencing devices such as PromethION and MinION (see figure 2.10). All of them follow the sequencing approach just described. The main difference among them lies in the throughput. PromethION offers high-coverage, it contains 2675 nanopore channels for sequencing DNA or RNA and has a yield of up to 14 Tb. On the other hand, MinION is an affordable, lightweigth portable version with 512 channels and has a yield of up to 50 Gb. It can plug directly into a standard USB3 port on a computer with low hardware requirement and simple configuration. Its high speed makes it suitable for real-time applications.

Recently, ONT presented SmidgION, the smallest sequencing device up to the date which can run on a smartphone or other mobile, low power devices (see figure 2.10).

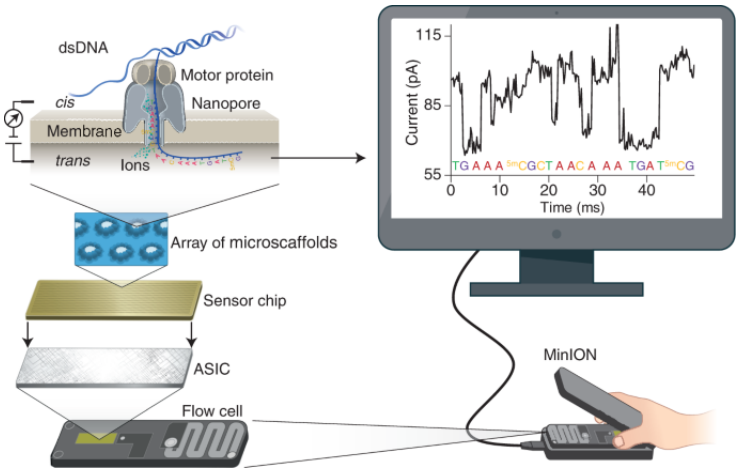


Figure 2.9 – Principle of Nanopore Sequencing. (Wang, Zhao, Bollas, Wang, & Au, 2021)

2.2 Characterization of the DNA Data Storage Channel: source of errors

Applying robust and efficient encoding techniques is highly important for DNA data storage to reduce its cost and maximize its reliability due to the biochemical procedures involved in the process which can corrupt the DNA molecules. Synthesis, sequencing, storage and the manipulation of DNA (mainly PCR amplification) may introduce imperfections in the DNA strands and cannot be ignored when designing DNA storage systems as they jeopardize the integrity of the stored content. The error can be either due to a physical change in the DNA strands, or due to an erroneous reading of a correct DNA strand and appears in the form of insertions, deletions and substitution of nucleotides (see figure 2.11).

Synthesis: The error rate of DNA synthesis is almost negligible when the synthesized oligos do not exceed 300 nts length and increases exponentially for longer strands. The noise can be introduced on single nucleotides or in the form of early termination, which is an extreme case of deletions. Early termination depends on the synthesis method and the position within the DNA sequence and occurs when the nucleotides can no longer be added to the growing strand due to the loss of chemical reactivity at its end. Additionally, not all the growing strands corresponding to the same target sequence face the same errors, meaning that there might be different variations for every reference. Finally, depending on the selected synthesis technology, the number of copies generated per reference may vary, leading to an uneven distribution of the synthesized strands.

Polymerase Chain Reaction: PCR amplification is commonly used to increase the redundancy in the DNA strands. However, the addition of redundancy is not its sole purpose. As the process of PCR requires the presence of primers in both ends of the amplified sequences, when performed in non-complete sequences (e.g. broken strands or sequences that suffered early termination during synthesis) those strands will not be amplified and thus, cleaned from the oligo pool. Despite being a reliable process, PCR introduces certain bias in the distribution of the number of copies as this process has a predisposition towards certain sequences over others. As an example,

Figure 2.10 – Top left: Illumina NextSeq 550 (Illumina, s. d.). Top right: PromethION, bottom left: MinION and bottom right: SmidgION (ONT, s. d.).

the number of generated copies is smaller for sequences with a high GC content.

Storage: DNA is prone to chemical decay if not stored under optimal conditions. Its degradation can lead to the loss of entire DNA molecules, which ultimately alters the distribution of the molecules in the pool. DNA decay has a minimal effect on the probabilities of insertions and deletions but significantly increases the substitution rate. The estimated half-life of fossilized DNA is around 500 years but it can be increased if stored in a controlled environment. There are several methods for the storage of DNA and the selection of the optimal one mainly depends on the frequency of access.

— **Long-term storage: cold data** (accessed every +10 years)
One of the main applications of long-term storage is the preservation of 'cold' data as it is the case of historical records, legal evidence or cultural patrimony. In this context, DNA can be stored in its dehydrated form and embedded into silk matrices, salts or even encapsulated. Some studies have shown that the encapsulation of DNA preventing its contact with water and oxygen provides the stored DNA a half-life at room temperature of approximately 170,000 years (Washetine et al., 2019). However, determining the stability of DNA in the long term remains a challenge as the methods to measure the degradation are not sensitive enough in most of the cases. Additionally, aging models are usually dependent on temperature but it remains unclear the existence of other degradation mechanisms which do not have dependency on temperature.

— **Medium-term storage: warm data** (accessed multiple times per year)
Although DNA encapsulation is a promising solution for the storage of DNA which allows to maintain its stability for hundreds or even thousands of years, it is not efficient when it comes to data that has to be accessed every few months as its physical manipulation is delicate and time-consuming. Instead, semi-accessible forms of storage are used for such cases as for instance refrigerated or frozen in aqueous solution and dry solid. Studies show that the stability of DNA stored at 4°C or below (either in aqueous form or dried) can last for around 2 years. However, the main challenge lies in the amount of degradation that

occurs every time the information has to be accessed. Every time the solution containing the DNA is frozen or thawed, ice crystals are formed increasing the probability of strand breakage, which increases with the length of the strands.

— **Short-term storage: hot data** (dynamic handling)
In-storage computation merges DNA-based computation and DNA storage systems. It has the potential of allowing direct search and edit of DNA and promises to lower the latency of conventional systems. However, such emerging systems will require the physical manipulation of the DNA molecules stored in a soluble aqueous form which risks the integrity of the DNA strands (e.g. strand breakage). Furthermore, although the biochemical environment of the molecules can be controlled to keep the optimal conditions to increase DNA stability, some biological manipulations involved in in-storage computation, editing or PCR amplification require temporary exposure to high temperatures increasing the probability of DNA degradation.

Sequencing: Since the release of nanopore sequencers, this technology has become more and more popular thanks to its affordability, small size and speed, which make it suitable for real-time applications. More precisely, nanopore-based sequencers measure the changes in the electrical conductivity as DNA strands pass through the pore. This electrical signal is then translated into a sequence of nucleotides in a process known as basecalling. Despite all the advantages that sequencers such as the MinION offer, it has a major drawback as it remains an error-prone process. Owing to the random and highly variable speed at which DNA molecules flow through the nanopores, the electrical signal suffers from random time-warping distortion, which causes insertion and deletion errors in the base-called sequences. Furthermore, measurement noise causes substitution errors. This constitutes the main challenge when using this device in the context of DNA data storage, compromising the decodability of the data. (Antonini et al., 2022)

Additionally, the error rate increases if the sequence contains homopolymers (>5 nts), pattern repetition or/and high GC content. According to some studies, the noise introduced by nanopore sequencers dramatically affects both ends of the DNA strands (Jain et al., 2015) but it remains unclear how the length of the input sequences affects its performance.

2.2.1 Main constraints in DNA coding

As described in the previous section, the biochemical processes involved on the DNA storage channel such as synthesis, PCR and sequencing are prone to errors and introduce some constraints that should be respected when designing coding mehtods for DNA data storage purposes. The main constraints include:

— Homopolymers: consecutive occurrences of the same nucleotides should be avoided.
— Patterns: the codewords used to encode the oligos should not be repeated forming patterns throughout the oligo length.
— G, C content: the percentage of G and C in the oligos should be between 40% and 60%.
— Length: the length of the oligos is limited to around 300 nucleotides in order for the synthesis process to be reliable, introducing the need for formatting.

Although respecting the constraints mentioned above does not guarantee that errors will not appear, it does increase the robustness of the code, reducing the chances of data loss.

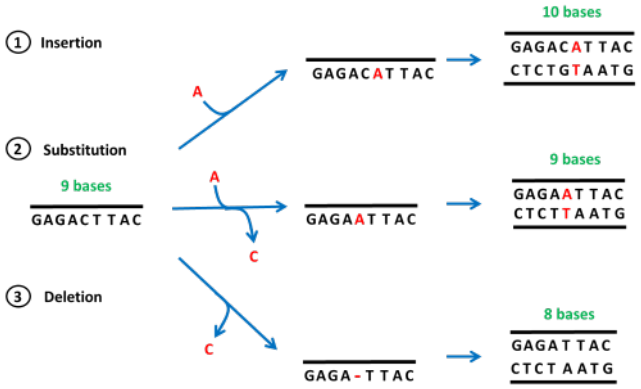


Figure 2.11 – Types of errors in DNA. Source: (Vivadifferences, s. d.).

2.3 State of the art in DNA coding

2.3.1 First attempt to store data into DNA

The first application of DNA data storage can be found in the work presented in (Church, Gao, & Kosuri, 2012) in 2012. Church et al. proposed a simple encoding to translate binary sequences into a quaternary representation by randomly assigning A or C to zeros and T or G to ones and tested the algorithm in a wet-lab experiment in which a 659-Kbyte book was stored into DNA. The synthetic DNA was sequenced using an Illumina sequencer and the analysis of the results constituted the first study of the sequencing errors as well as established the main constraints to respect when encoding data into DNA. The later works introduced these constraints together with error correction mechanisms to minimise the sequencing error.

2.3.2 Robust encoding and error correction

In 2013, Goldman et al. presented the first constrained encoding algorithm for DNA data storage. They proposed the use of ternary Huffman to avoid homopolymers (Goldman et al., 2013). They also included error correction based on redundancy on the formatted data by overlapping bases between adjacent fragments so each input bit was represented by multiple oligos.

Grass et al. were the first to introduce error correction in their encoding algorithm in 2015 using Reed Solomon codes (Grass, Heckel, Puddu, Paunescu, & Stark, 2015). Later on in 2016, Blawat et al. proposed a new method based on the creation of more than one dictionary to provide a multiple representation for each input symbol, allowing forward error correction (Blawat et al., 2016). Once created, the dictionaries were filtered to ensure that no homopolymers are created. Additionally, this work proposed the use of Reed Solomon codes to robustify the headers and allow a more reliable decoding. The same year, Elrich et al. published a method based on Fountain codes (Erlich & Zielinski, 2016), erasure codes initially designed for multicast and streaming applications. In few words, many possible oligos were created and the ones which did not respect the biochemical constraints were discarded. Authors also included Reed Solomon codes in the resulting oligos to confer extra robustness against sequencing errors.

Also in 2016, Borhholt et al. presented a DNA-based archiving system which used the encoding proposed by Goldman et al. but avoiding the fourfold redundancy as part of a Microsoft

research (Bornholt et al., 2016). The archiving system also integrated random access. In 2019, a Microsoft team demonstrated the first fully automated end-to-end system (Takahashi, Nguyen, Strauss, & Ceze, 2019) successfully writing, storing and reading back the word “hello".

It is important to mention that most of the methods proposed during the early stages of DNA data storage presented solutions for the encoding of any kind of digital data, translating from binary to quaternary representation. For the specific case of image coding, most of the studies proposed compressing images with JPEG before encoding and translating into quaternary using some constrained code and often adding redundancy for error correction. However, as the field of DNA coding grew, more works started to adapt the proposed encoding to the source. More concretely, since 2019 the Mediacoding team, has proposed various encoding solutions for the storage of digital images and that we will introduce in the following paragraphs.

In previous works our team proposed Paircode (Dimopoulou, Antonini, Barbry, & Appuswamy, 2019), an algorithm that generates a constrained quaternary code which avoids homopolymers and ensures that the GC content is kept within the optimal values. This work introduced for the first time image compression techniques in the process of DNA data storage, more concretely Discrete Wavelet Transform (DWT) and scalar quantization, unlike previous works that have been transcoding directly compressed binary sequences onto DNA. The proposed fixed-length solution was optimized thanks to a nucleotide allocation process across the different wavelet subbands by taking into account the input data characteristics, allowing the selection of the desired compression rate. Another asset of Paircode is that it can be used to detect errors by making use of the redundancy of the code. Later works enhanced the compression efficiency of the proposed algorithm by using vector quantization (Dimopoulou & Antonini, 2021).

Other works like (Pan et al., 2019) have also proposed a method for storing quantized images in DNA integrating Huffman coding in the encoding and applying image processing techniques to correct discolorations in the reconstructed image and further improve the quality of the decoding.

In 2021, our team proposed a variable-length solution based on the JPEG standard for image compression (Dimopoulou, Gil San Antonio, & Antonini, 2021a). The algorithm follows the same workflow of classical JPEG but replacing the two binary encodings originally used, Huffman coding and simple binary coding, by quaternary ones which respect the encoding constraints of DNA coding: Godlman encoding and Paircode.

2.3.3 Noise simulators

In the past years, several works have introduced sequencing simulators aiming to ease the implementation of new algorithms targeting nanopore-sequenced data. Such simulators allow testing while developing new tools thanks to their speed, low cost and high throughput. Commonly, simulators generate noisy reads using a model error profile. There are different types of noise models used by sequencing simulators, each of which aims to model the errors that can occur during DNA sequencing and storage. Some simulators use i.i.d. channel models, which assume that errors are independent and randomly generated. Other simulators model the errors in a more sophisticated way, including the transition probabilities for either single nucleotides or k-mers, which are short DNA sequences of k nucleotides. Some simulators even use deep learning methods to mimic the electrical current obtained with nanopore sequencing and introduce errors in a way that resembles the real-world scenario. This allows for a more realistic evaluation of the performance and robustness of DNA data storage systems. The main challenge when using these simulators for DNA data storage applications lies in the fact that their error models are generated from the sequencing

of complete genomes and thus, longer reads than in the case of synthetic DNA, whose length is currently limited to around 300 nts.

Some of the latest open source simulators are described in the following paragraphs:

NanoSim (Yang, Chu, Warren, & Birol, 2017)

This nanopore sequencing simulator models errors from 6 data sets using different generations of MinION sequencing kit as statistical mixture models. It samples the input reference and generates reads with specific length distribution. Each of those reads is then aligned to the training genomes. In case an alignment is found, the simulator will add errors from the mixture model corresponding to the genome, otherwise, if no alignment is found for the reads, it will add an arbitrary high error rate compared to the aligned reads.

Source code: `https://github.com/karel-brinda/NanoSim-H.git`

DeepSimulator (Y. Li et al., 2018)

DeepSimulator is a deep learning based simulator able to generate reads with almost the same properties as the real data. The workflow could be divided into three main modules:

1. Sequence generation module: Given the user-specified reference genome, as well as the number of reads to be generated, the sequence generation module randomly chooses a starting position in the reference sequence to produce the relatively short sequences, which satisfy the length distribution of the experimental nanopore-sequenced reads.
2. Signal simulation module: the pore model takes as an input a nucleotide sequence and outputs the expected current signal for each 6-mer (subsequence of 6 nucleotides) in the sequence. Then, random Gaussian noise is added according to the user-defined variance parameter to produce the simulated signals.
3. Basecaller: same as in the nanopore sequencer (described in section 2.1.4.2).

What makes DeepSimulator closer to the real sequencer is the fact that it does not explicitly introduce substitutions, insertions or deletions directly at the read level as is usually done in the rest of available simulators. Instead, it mimics the electrical signal produced by Nanopore sequencing as similar as possible and then, it is the basecaller that introduces the errors by itself as it happens in the real sequencing procedure.

Source code: `https://github.com/liyu95/DeepSimulator.git`

Badread (Wick, 2019)

Badread is a noise simulator for long reads targeting sequencers such as nanopore (ONT) and PacBio. Although the results might be less realistic than with other simulation tools, it has been built to give the users total control over the parameters of the simulations. The addition of the noise is performed by replacing k-mers (short fragments of k nucleotides) by erroneous ones following predefined transition probabilities. In this simulator, read lengths follow a gamma distribution with the mean and std specified by the user rather than experimental read length distribution. Additionally, it includes chimeras (reads that result from the concatenation of different

DNA sequences), addition of sequencing adapters, glitches and junk reads.

Source code: https://github.com/rrwick/Badread

MESA (Schwarz et al., 2020)

To our knowledge, MESA was the first simulator able to model the full DNA storage channel, including synthesis, PCR amplification, storage and sequencing. It uses either published error profiles or user-defined rates. The sequencing module allows to simulate reads from different sequencing platforms such as Illumina, PacBio and Nanopore. In addition, this tool offers the possibility of assessing the quality of the DNA fragments regarding their content (i.e. whether the biological constraints are respected or not).

Source code: https://github.com/umr-ds/mesa_dna_sim
Web application: https://mesa.mosla.de/

PBSIM2 (Ono, Asai, & Hamada, 2021)

PBSIM2 is another sequencing simulation tool for long-read sequencers. This simulator considers the non-uniformity of errors (or quality scores) thanks to a generative model for quality scores based on hidden Markov model (HMM), achieving a tendency closer to experimental reads. It considers the read length, accuracy distribution and quality score distribution from experimental reads and introduces single nucleotide errors according to this last one. Results show that the generative model simulates quality scores that are more consistent with real reads of PacBio and Nanopore (ONT) than other existing simulators.

Source code: https://github.com/yukiteruono/pbsim2

DNA Storage Error Simulator (Alnasir, Heinis, & Carteron, 2022)

This DNA Storage Error Simulator simulates errors at all stages of the DNA storage workflow. The simulation is based on data obtained from the literature, and for the synthesis and sequencing phases, pre-determined probabilities for insertions, deletions and probabilities can be used from pre-existing datasets. The tool can also simulate errors in PCR, given the number of cycles, as well as degradation of DNA in the storage phase given the number of years and temperature the DNA was stored at.

Web application: https://master.dbahb2jho41s4.amplifyapp.com/

2.3.4 Clustering

The first step when retrieving data stored into DNA is to increase the redundancy. Oligos are physically replicated through PCR amplification, resulting in a pool which contains many noisy copies of the original sequences. Thus, a read consensus procedure is required to infer the original oligos based on the noisy reads so that the inferred sequences can be passed to a decoder to recover

the original data. Consequently, it is imperative to cluster the reads into groups which contain all the noisy sequences coming from the same reference. Consensus is applied then to each cluster to get the most representative sequence. These two steps prior to decoding (clustering + consensus) constitute a computational bottleneck in the DNA data archival pipeline due to two main reasons. First, DNA amplification leads to millions of strings which need to be clustered without a priori information which often results in memory issues. Second, the string similarity metric used by the clustering algorithm should consider not only substitutions but also insertions and deletions, such as Levenshtein, or edit, distance (defined in section 2.3.6), whose computation is much heavier than simpler metrics like Hamming distance (also defined in section 2.3.6) which only considers substitutions.

Some of the state of the art clustering algorithms for DNA sequences are defined in the following paragraphs.

Starcode (Zorita, Cusco, & Filion, 2015)

Starcode is a DNA sequence clustering software based on all-pairs search. It matches pairs of reads within a pre-defined Levenshtein distance. Clustering is done after the all-pair matching step and it allows to select among three different clustering algorithms: message passing, sphere clustering or connected components depending on the needs of the user. Starcode returns the canonical sequence of the cluster, the cluster size, the set of different sequences that compose the cluster and the input line numbers of the cluster components.

Source code: `https://github.com/gui11aume/starcode`

OneJoin (Marinelli & Appuswamy, 2021)

OneJoin is a cross-architecture edit similarity join. It is based on EmbedJoin (Zhang & Zhang, 2017), a string similarity join that uses edit-to-Hamming embedding together with Locality Sensitive Hashing. The main idea behind is to transform a set of strings into an embedded representation such that the Levenshtein distance between two strings in the original set can be approximated by the Hamming distance between strings in the embedded set. This embedding significantly reduces the complexity of the problem (while Hamming distance has a complexity that is linear with the string length, Levenshtein distance has a complexity that is quadratic). As a result, the OneJoin retrieves all pairs of oligos which are similar.

OneConsensus (Marinelli et al., 2022)

OneConsensus is a new consensus procedure developed to solve the OneJoin's high memory and computational usage. The algorithm relies on CKG-Embedding and Locality Sensitive Hashing (LSH). The first one is used for embedding problems from an edit space into a Hamming space and Hamming LSH is used over the embedded reads to group together these sequences that, with a very high probability, are similar to each other. As LSH can produce false positives, the reads in each pool are sorted by moving to the front the ones with length matching the length of the reference oligos. Then the reads in the back of a pool are aligned to the ones in the front, progressively building the clusters. Finally, base-by-base consensus is applied on the aligned reads of each cluster.

2.3.5 Other tools for data handling and analysis

Many other tools have been released in the past decade to facilitate the extraction, analysis and data handling of nanopore sequencing data. However, many of these tools have been developed targeting biological data and not all of them can be used in the context of DNA data storage.

Porechop (Porechop, 2018)

Adapters and primers are a key component for the library preparation prior to sequencing. They are short, chemically synthesized sequences that are ligated to both ends of the DNA strands and which allow the sequencing machines to recognize the target data. Once sequencing is finished, these short fragments should be removed from the output reads as they were artificially added to the initial oligos.

Porechop is a tool for finding and removing adapters. Adapters on the ends of reads are trimmed off, and when a read has an adapter in its middle, it is treated as chimeric and chopped into separate reads. Porechop performs thorough alignments to effectively find adapters, even at low sequence identity (i.e. noisy data). Although it was specifically designed for trimming adapters from Oxford Nanopore reads, the source code can be easily modified to add customised ones, such as the Illumina primers.

Source code: `https://github.com/rrwick/Porechop`

Minimap2 (H. Li, 2018)

Minimap2 is a general-purpose pairwise aligner to map DNA or sequences against a reference. It can be used as a read mapper, long-read overlapper or a full-genome aligner.

Source code: `https://github.com/lh3/minimap2`

NanoOk (Leggett, Heavens, Caccamo, Clark, & Davey, 2016)

NanoOk constituted the first integrated tool that allowed the extraction of the nanopore output and provide an alignment-based quality control and error profile analysis. It generates a report including error profile, quality and yield data. NanoOk uses pre-existent alignment tools to map nanopore reads to the reference (LAST, BWA-MEM, BLASR or marginAlign). Even though it was developed for biological purposes rather than synthetic DNA, it is multi-reference, meaning that it allows the analysis of multiple references as in the case of DNA data storage, thus, can be exploited with synthetic data.

Source code: `https://github.com/TGAC/NanoOK`

Squigglekit (Ferguson & Smith, 2019)

SquiggleKit is a toolkit for manipulating and analysing nanopore data that simplifies file handling, data extraction, visualization and signal processing. It allows the management of the large number of data files generated during nanopore sequencing and constitutes a starting point for the development of bioinformatic tools as well as the creation of probabilistic models for nanopore-sequenced reads and fine-tuned data sets for machine learning.

Source code: https://github.com/Psy-Fer/SquiggleKit

2.3.6 Evaluation metrics

Hamming distance

In information theory, Hamming distance is defined as the number of positions in which two strings of equal length differ. The computation of the Hamming distance is easy and its complexity increases linearly with the sequence length. However, this metric only considers mismatches (substitution errors) which constitutes a major drawback for its use in the context of DNA data storage as the errors can appear in the form of not only substitutions but also insertions and deletions of nucleotides. To better illustrate the limitations of this string metric, one can consider the two strings ATCTG and TCTGC, the Hamming distance would be 5. However, the first string can be transformed into the second just by removing the A and adding a C, i.e. the strings are only two edits apart (one deletion and one insertion).

Levenshtein (or edit) distance

In information theory, Levenshtein or edit distance is a metric for measuring the differences between two strings given by:

$$d_L(i,j) = \begin{cases} \max(i,j) & \text{if } \min(i,j) = 0 \\ \min \begin{cases} d_L(i-1,j)+1 \\ d_L(i,j-1)+1 \\ d_L(i-1,j-1)+1_{(a_i \neq b_j)} \end{cases} & \text{otherwise.} \end{cases}$$

In few words, the edit distance could be described as the number of edits in terms of substitutions, insertions and deletions of nucleotides necessary to transform one string into another. The main limitation of this string metric is that the complexity increases exponentially with the string length, making its computation very expensive when dealing with long sequences.

Peak signal-to-noise ratio (PSNR)

PSNR is a metric widely used to quantify the quality of reconstructed images and videos after lossy compression. It is defined as the ratio between the maximum possible power of a signal and the power of the noise affecting the signal. It is usually expressed as a logarithmic quantity in the decibel scale as many signals present a very wide dynamic range. Given an image I and its noisy approximation $\hat{I}$, both of size mxn, the PSNR can be defined via the mean squared error (MSE) given as follows:

$$PSNR = 10 \log_{10} \frac{MAX_I^2}{MSE}$$

where MAX_I^2 is the maximum possible pixel value of the image and

$$MSE = \frac{1}{mn} \sum_{i=1}^{m} \sum_{j=1}^{n} (I(i,j) - \hat{I}(i,j))^2.$$

CHAPTER 3

Enhancing DNA image retrieval through robust encoding and post-processing

3.1 Introduction

The storage of digital data into DNA has the potential to be a highly efficient and durable method for long-term data storage. However, in order to realize this potential, it is crucial to develop mechanisms that ensure the decodability of the data. There are several processes in the DNA data storage channel which contribute to the degradation of the DNA molecules such as synthesis, PCR amplification, sequencing and even storage if the DNA is not preserved under the right conditions. Robust encoding and error correction mechanisms can help to ensure that the stored data remains accurate and accessible for long periods of time, even in the presence of degradation or errors.

To mitigate the impact of such errors, researchers in the field of DNA data storage have developed various error correction mechanisms, such as redundancy encoding and error correction codes, to help detect and correct errors during the decoding process. However, most of those methods introduce important redundancy without promising full error correction.

Up to the date, Mediacoding group has proposed two main encoding solutions for the storage of digital images into DNA: a fixed-length solution based on Discrete Wavelet Transform (DWT) and vector quantization (VQ), and a variable-length solution inspired by the classical JPEG. In this chapter, we propose solutions adapted to these encoding schemes to ensure the decodability of the stored data and improve the quality of the reconstruction.

Studies have shown that, in the case of variable-length codes, even small amounts of errors can greatly impact the decodability of the data and, in some cases, the structure of the image is completely lost. Hence, we propose a method to robustify a JPEG-inspired codec for DNA data storage with the goal of ensuring the decodability of the stored data even in the presence of errors and degradation. On the other hand, we also propose a post-processing technique adapted to the wavelet domain. The algorithm is applied to the decoded wavelet subbands and aims to correct the errors that remain after decoding. The algorithm also includes an automatic detection of errors and correction with inpainting, more concretely texture synthesis. Overall, these methods provide

a robust and efficient solution for DNA data storage, ensuring accessibility of the stored data for long periods of time.

3.2 Robustifying the JPEG-inspired coder for DNA coding

One of the main limitations in data storage into DNA is the high cost of DNA synthesis, which yields the need to efficiently compress the data in order to reduce the cost. Despite the higher complexity compared to fix-length solutions, variable-length encodings provide a more compressed representation of the input data. However, one of the main drawbacks is the lower robustness to errors, which can jeopardise the correct decoding of the stored data and consequently cause the loss of information. In (Dimopoulou et al., 2021a), we proposed a variable length encoding solution inspired by the main workflow of the classical JPEG * standard. Results showed that a single error in the encoded quaternary strand can affect the structure of the stored image. Hence, robustifying the code is imperative if it is to be used in the context of DNA data storage.

3.2.1 JPEG-inspired DNA codec

JPEG is a commonly used method of lossy compression for digital images. Roughly, the JPEG algorithm can be divided into:

— Colour transform and sub-sampling (for RGB images)
— DCT on 8x8 pixel blocks
— Quantization
— Variable-length encoding

The output of the DCT is a set of 64 coefficients which can be divided into DC and AC coefficients. After quantization, the DCT coefficients and the categories associated to them (see Table 3.1) are encoded using two main coding techniques: Huffman coding and simple binary coding.

Table 3.1 – Example of Category-Coefficient relation table

Category	Negative coefficients	Positive coefficients
0	Null	0
1	[-5, -1]	[1, 5]
2	[-25, -6]	[6, 25]
3	[-75, -26]	[26, 75]
4	[-275, -76]	[76, 275]
5	[-775, -276]	[276, 775]
6	[-2775, -776]	[776, 2775]
7	[-7775, -2776]	[2776, 7775]
8	[-27775, -7776]	[7776, 27775]

Categories are encoded using Huffman coding, a variable-length coding technique that works by assigning binary words to the different values while respecting two rules:

1. Each binary representation is not a prefix of another.

*. https://jpeg.org/jpeg/index.html

2. The shortest words are assigned to the most frequent elements.

The first rule prevents any ambiguity in the decoding process while the second one enhances the performance of the encoding in terms of compression. On the other hand, DCT coefficients are encoded using binary coding, which simply transforms each value into its binary representation using a number of bits which is predefined.

The JPEG-inspired coder for DNA coding proposed in (Dimopoulou et al., 2021a) follows the same workflow of classical JPEG but those two binary encodings have been replaced by quaternary ones which respect the encoding constraints of DNA coding: Goldman encoding and Paircode. Goldman encodes the categories into a stream of the trits † 0, 1 and 2 using a ternary Huffman. Then, each trit is replaced with one of the nucleotides, excluding the one which was previously used, this way ensuring that no homopolymers are generated. Paircode, described in the next section, is used as a fixed length coder for the indexes instead of binary coding. Figure 3.1 shows the modified JPEG workflow adapted to DNA coding.

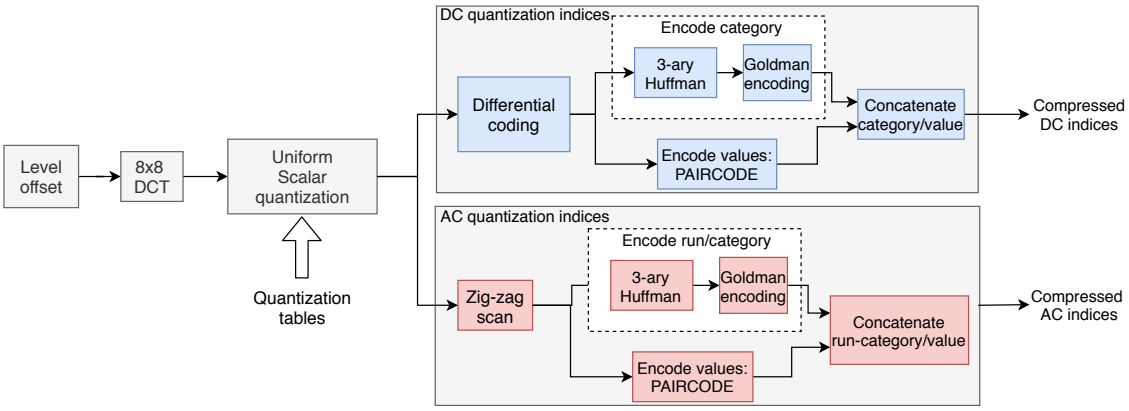


Figure 3.1 – Workflow of the modified JPEG workflow to suit the needs of DNA coding.

3.2.2 Codeword construction with Paircode

Paircode (Dimopoulou, Antonini, et al., 2019) is an algorithm for the construction of a DNA-like quaternary code inspired by the restrictions imposed by the biochemical procedures included in the process of DNA data coding. Those constraints involve avoiding homopolymers and keeping the GC content between 40% and 60%. Paircode encoding algorithm works as follows:

Let's assume the source set $\mathcal{V}$, and let $\Sigma = \{1, 2, \ldots, K\}$ with $|\Sigma| = K$, be a set of indices of the elements $v_k \in \mathcal{V}$ to be encoded into a set $\mathcal{C}^* = \{c_1, c_2, \ldots, c_L\}$ of L quaternary codewords (with $L \geq K$) of length l. The goal of this encoding algorithm is to generate the code Γ where $\Gamma : \Sigma \to \mathcal{C}^\star$. We denote $\Gamma(k) = c_k$ the codeword associated with the index $k \in \Sigma$.

The main idea is the creation of quaternary codewords by selecting elements from the following dictionaries:

— $\mathcal{D}_1 = \{AT, AC, AG, TA, TC, TG, CA, CT, GA, GT\}$

— $\mathcal{D}_2 = \{A, T, C, G\}$

†. Trit: ternary equivalent of a bit.

Codewords of an even length l, are constructed by selecting $\frac{l}{2}$ pairs from dictionary $\mathcal{D}_1$. Codewords of an odd length l, are constructed by selecting $\frac{l-1}{2}$ pairs from $\mathcal{D}_1$ and adding a symbol from $\mathcal{D}_2$ at the end of the codeword. To ensure that the code does not create homopolymers, dictionary $\mathcal{D}_1$ omits the pairs AA, TT, CC, and GG and to keep the GC content within an acceptable range, the pairs CG and GC are also excluded. As a result, the size L of the code $\mathcal{C}^*$ is restricted to specific values as the words should be created according to the constraints imposed by the biochemical process of DNA coding. Consequently $L \in \{l_1, l_2, \dots\}$ where:

$$l_1 = 10 \text{ and } l_{i+1} = \begin{cases} 4l_i, & i\text{: odd} \\ 10l_{i-1}, & i\text{: even} \end{cases}$$

Despite the fact that this encoding algorithm does not contain all the possible permutations of the four DNA symbols (A, C, T and G) could be considered a drawback in terms of coding potential, the a priori knowledge about the words which are not considered in the code can be used to achieve a better decoding by adding some sort of error detection and correction during the decoding phase, as it will be further described in section 4.2. Figure 3.2 depicts an example of a 3-nt codebook where the red words are omitted according to the above code construction algorithm, ensuring that the concatenation of the codewords will respect the biochemical constraints of DNA sequencing.

AAA	AAT	AAC	AAG	ATA	ATT	ATC	ATG	ACA	ACT
ACC	ACG	AGA	AGT	AGC	AGG	TAA	TAT	TAC	TAG
TTA	TTT	TTC	TTG	TCA	TCT	TCC	TCG	TGA	TGT
TGC	TGG	CAA	CAT	CAC	CAG	CTA	CTT	CTC	CTG
CCA	CCT	CCC	CCG	CGA	CGT	CGC	CGG	GAA	GAT
GAC	GAG	GTA	GTT	GTC	GTG	GCA	GCT	GCC	GCG
GGA	GGT	GGC	GGG						

Figure 3.2 – All the possible permutations of the four DNA symbols for creating codewords of 3 nts. The algorithm for the construction of the codeword excludes all the codewords in red.

3.2.3 NoRM: Noise Resistant Mapping

In (De Marca, Jayant, et al., 1987), the authors proposed an algorithm for assigning binary words to codevectors of a multi-dimensional vector quantizer in such a way so to be resistant to single-word errors which are inserted by a binary symmetric channel. Inspired by this idea, in (Dimopoulou, Gil San Antonio, & Antonini, 2020) and (Dimopoulou, Gil San Antonio, & Antonini, 2021b) we presented an extension of the method which we call NoRM (Noise Resistant Mapping). The main goal is to map the input vectors obtained from a Vector Quantization algorithm and the quaternary codewords from our code in a way that the impact of a substitution error in the quaternary sequence is minimized.

Before describing the proposed mapping algorithm it is important to introduce some basic notions and definitions. As explained in the previous section, the size L of the code $\mathcal{C}^*$ is restricted to specific values as the words should be created according to the constraints imposed by the process of DNA coding. Hence, the size of the vector codebook $\mathcal{V}$ will be restricted to $K = L$.

The objective is to build a mapping function $\Gamma : \Sigma \rightarrow \mathcal{C}^{\star}$ which maps codewords c_k from a code $\mathcal{C}^*$ which differ from each other at exactly one position (Hamming distance of 1) to codevectors v_k from a vector set $\mathcal{V}$ which are close in terms of Euclidean distance. The purpose of this mapping lies in the fact that in case of an error during sequencing and assuming that the sequencing noise is small enough, a correct codeword will be transformed to another one which will have a small Hamming distance with the correct one. Thus, if those two codewords are assigned to input vectors which are close, the error will not significantly affect the image quality. To perform the assignment of DNA codewords to the input vectors, we first construct for each codeword $c_k \in \mathcal{C}^*$ a sphere $H(c_k)$ containing the codewords which have a Hamming distance of 1 compared to c_k. To further define which vectors can be considered as close vectors we introduce for every vector $v_k \in \mathcal{V}$ a set $S(v_k)$ which contains the closest vectors to the vector v_k in terms of Euclidean distance. The central idea of the proposed mapping is to assign codewords of the same sphere to vectors which belong to the same neighborhood as shown in figure 3.3. However, such an assignment is not possible for every neighborhood $S(v_k)$ and thus it is necessary to perform this assignment according to some priority. To this end, we define an empirical function $F(v)$ introduced by (De Marca et al., 1987) for a vector v_k as:

$$F(v_k) = \frac{p(v_k)}{\alpha^{\beta}(v_k)}$$

where $p(v_k)$ is the probability of v_k in the input sequence, and

$$\alpha(v_k) = \sum_{j|v_j \in S(v_k)} d_E(v_j, v_k)$$

where $d_E(v_j, v_k)$ represents the Euclidean distance between the vectors v_j and v_k, and $\beta \geq 0$ a trade-off parameter. Therefore vectors with a higher F are considered to belong in a denser neighborhood and should consequently get higher priority to be assigned to the same sphere of words.

For further information regarding the algorithm used to progressively map the vectors to the DNA codewords, interested readers should refer to (Dimopoulou et al., 2020). Results showed a very promising increase of PSNR, providing a noticeable visual improvement of up to 7 dB when using complete codebooks.

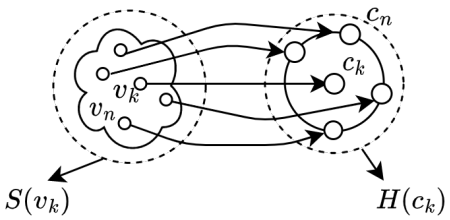


Figure 3.3 – Ideal mapping scenario in which all the vectors in $S(v_k)$ are mapped to the codewords in the Hamming sphere .

3.2.3.1 Applying NoRM to the JPEG-inspired DNA codec

In this work, we propose the use of NoRM to robustify the JPEG-inspired coder for DNA coding described in section 3.2.1. In this coding algorithm, DCT is applied block by block like

in classical JPEG. As previously mentioned, the output coefficients are divided into DC and AC as they do not describe the same information. A DC coefficient is the first coefficient obtained through the DCT and describes the zero frequency characteristic of the signal on the input block. It is comparable to an average of the signal over the whole block and losing this information would lead to value shifts at a block-wide scale when decoding. To provide a more intuitive explanation, we can say that the juxtaposition of all the DC coefficients is equivalent to a smaller representation of the input image with its size reduced by a factor equal to the size of the blocks. This smaller representation is commonly called thumbnail. On the other hand, AC coefficients describe non-zero frequencies, meaning variations over the values in the block. Consequently, losing information on this part has a less significant impact specially at very high frequencies. In other words, DC coefficients contain more critical information and should be further protected to ensure the decodability of the stored data. As an example, figure 3.4 shows the decoding result of an image that has been encoded using the JPEG-inspired DNA codec and which contains substitution errors in the encoded DC coefficients.

Figure 3.4 – Example of visual distortion on images encoded with the JPEG-inspired codec for DNA with 0.05% of substitution errors on the encoded DC coefficients.

Since the proposed algorithm concerns the assignment of DNA words to vectors in a way that reduces the visual impact of errors by using the Hamming distances between words of the same length, by construction this algorithm can't be applied to variable-length coding. Hence, in order to robustify the codec using the proposed mapping resistant to noise, the variable-length solution used to encode the DC coefficients must be replaced by some fixed-length code. In this work we replaced ternary Huffman and Goldman encoding by vector quantization and encoding with Paircode as depicted in figure 3.5. Thanks to this modification on the thumbnails encoding, NoRM can be used for assigning the quantization indices to the quaternary words created with Paircode, robustifying the DC coefficients against errors.

One of the main drawbacks of replacing the variable-length solution by a fixed-length one is lowering the compression efficiency. In (Dimopoulou et al., 2021a) we showed how in DNA

coding the selection of the best encoding solution does not only rely upon the performance, but highly depends on the robustness to noise as one single error can cause the loss of the structure of the image.

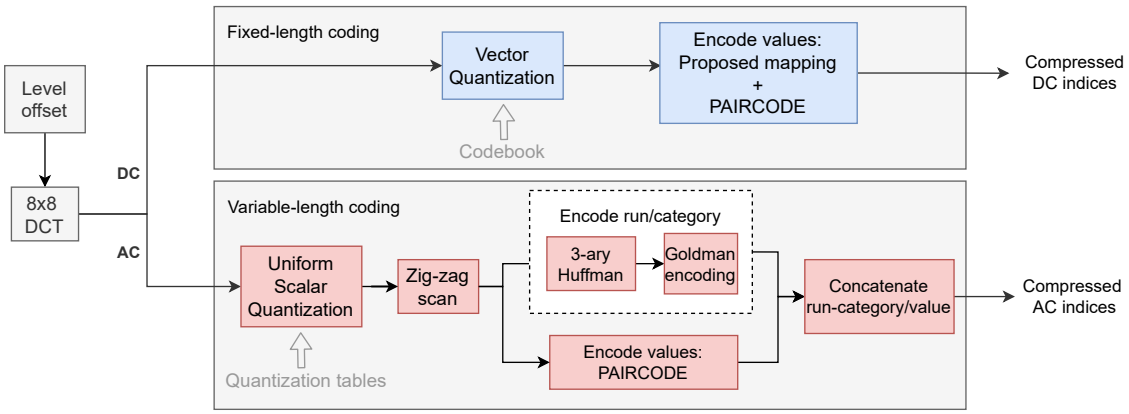


Figure 3.5 – Workflow of the modified JPEG-inspired DNA codec to robustify the encoding of the DC coefficients.

3.2.4 The problem of the undecodable errors

Assuming a substitution error in the encoded quaternary stream, some correct codeword $c_c \in \mathcal{C}^*$ will be transformed to an erroneous codeword c_e with either $c_e \notin \mathcal{C}^*$ or $c_e \in \mathcal{C}^*$. Under the hypothesis that the error rate produced by the sequencer will be reasonably small, the Hamming distance between the correct and the erroneous codeword will be $d_H(c_c, c_e) = 1$. Therefore, for each correct codeword c_c of length l there are different erroneous codewords of distance 1. The set of all possible erroneous codewords can be, according to coding theory, represented in the Hamming space as a sphere $H(c_c)$ of radius 1 the center of which is the correct codeword c_c. An example of such a sphere is depicted in figure 3.6.

Let us define $\mathcal{D}$ the set of codewords of length l constructed by using all 4^l possible arrangements of A, T, C and G such that $\mathcal{C}^* \subset \mathcal{D}$. In the case of $\mathcal{D}$, there would be 4^l different spheres the cardinality of which would be $|H(c_c)| = 3l$. However, as explained in section 3.2.2, the code $\mathcal{C}^*$ used for DNA coding excludes some words which can't be viable due to the biochemical encoding constraints. Therefore, in this work we consider K $(K \leq 4^l)$ different spheres with varying cardinality. In other words, some codewords c_e that would normally belong to some sphere of center c_c might be omitted due to the fact that they do not respect the rules of DNA coding. As a result, a substitution can cause two different possible types of error:

— **Decodable error ($c_e \in \mathcal{C}^*$):** The substitution transforms a correct codeword c_c to an erroneous word c_e which exists in the constrained code $\mathcal{C}^*$ proposed in section 3.2.2. The decoding will then provide an erroneous vector v_e instead of the correctly decoded vector v_c. Thanks to NoRM —the proposed robust mapping— in most of the cases the Euclidean distance $d_E(v_c, v_e)$ will be small and the produced error will have a lower impact on the visual quality of the decoded image.

— **Undecodable error ($c_e \notin \mathcal{C}^*$):** The substitution transforms a correct codeword w_c to an erroneous codeword w_e which does not exist in the constrained code $\mathcal{C}^*$. In this case

decoding is not possible and thus the application of some error correction is necessary to allow decoding. The applied correction techniques are further described in section 3.2.4.1.

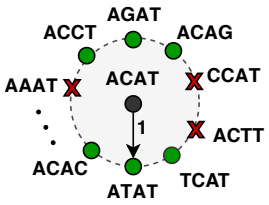


Figure 3.6 – Example of a Hamming sphere. The cross-elements denote non viable words that would belong in the Hamming sphere but are omitted due to the constrained quaternary code.

3.2.4.1 Proposed decoding of undecodable errors

As discussed in the previous section, in the case in which a substitution error creates an undecodable word it is necessary to employ some error correction to allow decoding. In this section we will describe different possible methods for assigning a value to the erroneous undecodable words.

Due to the existence of non-decodable errors, the decoding works in two consecutive cycles. In the first cycle, it performs a first decoding by simply omitting all the undecodable words. More precisely, this step decodes the words which exist in the code $\mathcal{C}^*$ to the corresponding VQ indices of Σ while in parallel reconstructing the image in the VQ block-space to retrieve the spatial information.

After this decoding cycle, the undecodable codewords w are still expressed in quaternary representation but are spatially placed to the corresponding positions in the image resulting in a decoding state which is described in figure 3.7 (left).

The second round of the decoding algorithm aims to decode the remaining non-decodable words.

Before presenting the different decoding methods proposed in this work, it is important to define the following notions:

— Let w be an undecodable erroneous codeword.
— We define as $\mathcal{C} \subset \mathcal{C}^*$ the set of decodable neighboring codewords c_j with $j \in \{1, \ldots, |\mathcal{C}|\}$ where $0 \leq |\mathcal{C}| \leq 8$, which are found around an undecodable word w. The size of the neighborhood depends on the needs of the encoding process and can vary. In our study, for simplicity, we are going to test the proposed algorithms for the 8 neighboring elements around w (1-ring neighborhood).
— We define $\mathcal{V}_c \subset \mathcal{V}$ as the set of vectors v_j with $0 \leq |\mathcal{V}_c| \leq 8$ to which the codewords in $\mathcal{C}$ have been assigned.
— We define as $\mathcal{V}_H$ the set of vectors v_i, $1 \leq i \leq |H(w)|$ to which have been assigned the codewords in the Hamming sphere $H(w)$.
— For each vector v_i in $\mathcal{V}_H$, define the set $S(v_i)$ which contains the closest vectors to v_i.

Baseline

We first propose a simple solution that will serve as baseline to asses the performance of the different solutions described in this section. The baseline simply consists on decoding as zero every pixel of the image to which no value has been assigned after the first round of decoding due to the presence of non decodable erroneous codewords.

Advanced Correction Decoding

In (Dimopoulou et al., 2020) we proposed a method for decoding the non-decodable errors which we called Advanced Correction Decoding (ACD). This method relies on the assumption that there might exist spatial correlations among close vectors in the stored image. In few words, given a non-decodable word, it will be replaced by the intersection between the words belonging to the Hamming sphere whose center is the non-decodable word and the set containing its neighboring words in the reconstructed image. In case that the intersection contains more than one word, the most frequent one in the neighborhood will be selected. If no word is more frequent than the others, it will be selected randomly. Whenever there is no intersection between the two sets, we compute the average Euclidean distance between each vector mapped to the words in the Hamming sphere $H(w)$ and the neighboring vectors $\mathcal{V}_c$. The vector with lowest average euclidean distance will be used for decoding. The method can be expressed as follows:
For each non-decodable codeword w, define the set $P = \mathcal{C} \cap H(w)$

— If $|P| = 1$, $w \leftarrow c_z \in P$

— If $|P| > 1$

- Define $f(c_z)$ as the frequency of a codeword c_z, $c_z \in P$
- $w \leftarrow c_z \in P$ such that $c_z = \arg\max_z f(c_z)$

— if $|P| = \varnothing$

- Compute $D(v_k) = \frac{\sum_{i=1}^{K} \mathbb{1}(v_i \in \mathcal{V}_H) d_E(v_i, v_k)}{|\mathcal{V}_H|}$, $\forall v_k \in \mathcal{V}_H$

with $\mathbb{1}(v_i \in \mathcal{V}_H) = \begin{cases} 1 \text{ if } v_i \in \mathcal{V}_H \\ 0 \text{ otherwise} \end{cases}$

- $w \leftarrow w_d$ where $w_d = \Gamma(v_d)$ with $v_d := \arg\min_v (D(v))$

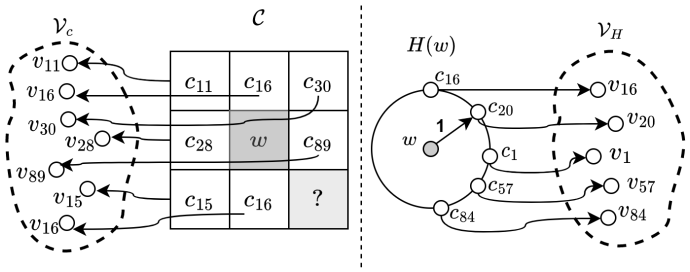


Figure 3.7 – The different sets used for ACD decoding. Each undecodable word w constitutes the center element of a neighborhood $\mathcal{C}$ of neighboring codewords c_j. The set $\mathcal{V}$ contains the vectors to which are mapped the codewords $c_j \in \mathcal{C}$. Each undecodable codeword w is the center of a sphere $H(w)$ containing decodable codewords c_i which have a Hamming distance of 1 to w. The vectors to which are mapped the codewords that belong to the sphere $H(w)$, form the set of vectors $\mathcal{V}_H$.

Advanced Correction Decoding 2

In this thesis we propose a variation of the just described ACD solution, which we call ACD2, following the same assumption which indicates that since the input data is an image there can be correlations between neighboring elements. For this method we also consider the neighbouring codewords $\mathcal{C}$ and the Hamming sphere around the undecodable erroneous word H(w), however, instead of focusing on the intersection of these two sets ($P = \mathcal{C} \cap H(w)$), we go one step further and, for every vector v_i in $\mathcal{V}_H$, we consider the sets $S(v_i)$ which contain the closest vectors to v_i in terms of Euclidean distance. Then, we compute the average Euclidean distance between each set $S(v_i)$ and the vectors in $\mathcal{V}_c$, which contains the vectors mapped to the neighboring elements of w. Finally, the undecodable word w will be decoded as the vector v_i for which $S(v_i)$ provided the smallest average Euclidean distance to $\mathcal{V}_c$.

For each non-decodable codeword w, the algorithm acts according to the following steps:

— Compute

$$D(v_k) = \frac{\sum_{i=1}^{K}\sum_{j=1}^{K} \mathbb{1}(v_i \in \mathcal{S}(v_k))\mathbb{1}(v_j \in \mathcal{V}_c) d_E(v_i, v_j)}{|\mathcal{S}(v_k)||\mathcal{V}_c|}, \forall v_k \in \mathcal{V}_H$$

$$\text{with } \mathbb{1}(v_i \in \mathcal{S}(v_k)) = \begin{cases} 1 \text{ if } v_i \in \mathcal{S}(v_k) \\ 0 \text{ otherwise} \end{cases} \quad \text{and } \mathbb{1}(v_j \in \mathcal{V}_c) = \begin{cases} 1 \text{ if } v_j \in \mathcal{V}_c \\ 0 \text{ otherwise} \end{cases}$$

— $w \leftarrow w_d$ where $w_d = \Gamma(v_d)$ with $w_d \in \mathcal{H}(w)$ and $v_d := \arg\min_v(D(v))$

Figure 3.8 depicts the vectors and neighborhoods involved in the decoding of erroneous, non-decodable words.

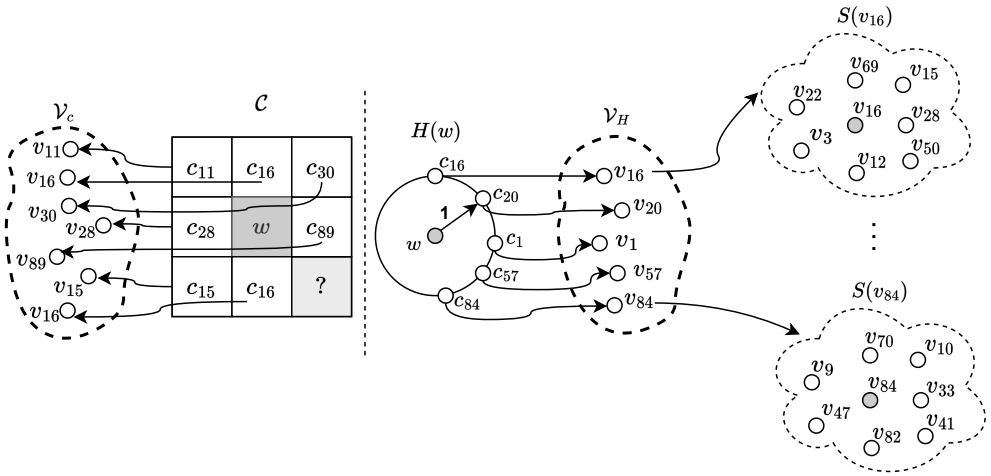


Figure 3.8 – The different sets used for ACD2. In the image spatial domain each undecodable codeword w constitutes the center element of a neighborhood $\mathcal{C}$ of neighboring codewords c_j. The set $\mathcal{V}$ contains the vectors to which are mapped the codewords $c_j \in \mathcal{C}$. Each undecodable codeword w is the center of a sphere $H(w)$ containing decodable codewords c_i which have a Hamming distance of 1 to w. The vectors to which are mapped the codewords that belong to the sphere $H(w)$, form the set of vectors $\mathcal{V}_H$. $\mathcal{S}(v_i)$ represents a set containing the N closest vectors to v_i in terms of euclidean distance.

Inpainting

The missing values in the image caused by non-decodable errors are filled using image inpainting. Inpainting algorithms aim to reconstruct missing regions in an image. There are many families of inpainting techniques, each better suited to handle different types of damage. In this study, the erroneous regions that we aim to correct are known and rather small (vectors of 2 pixels which correspond to errors in the DNA sequence leading to non-decodable words that cannot be translated into the corresponding pixel values). For this reason, this work uses the algorithm described in (Bornemann & März, 2007). In few words, the algorithm is a coherence transport based inpainting method and it estimates the value for each target pixel (defined by a binary mask) from its coherent neighboring pixels with known values.

3.2.5 Workflow of the experiment

One of the most important steps in vector quantization is the creation of an optimal codebook, which stores the vectors used in the quantization process. In this experiment, the codebook was obtained from a set of 20000 images randomly selected from the Flickr data set ‡ (Young, Lai, Hodosh, & Hockenmaier, 2014) which contains photographs of everyday activities, events and scenes. As the goal is to robustify what we call the thumbnail (i.e. the image constructed by concatenating the DC coefficients) after DCT, prior to constructing the codebook we extracted the thumbnails from the training set of images and clustered the vectors obtained from them. Given the extreme importance of having a good quality in the thumbnails, we created a codebook of 1000 vectors. The length of the vectors was limited to 2 pixels due to the reduced size of the target thumbnails in order to avoid blocking effects.

The images used to test the experiment were obtained from the Kodak Lossless True Color Image §. The image test set contains 25 images stored in PNG (Portable Network Graphics ¶), a format for storing raster (bitmapped) images that supports lossless data compression. Many sites use them as a standard test suite for compression testing, etc. All the images have a size of either 768x512 or 512x768 pixels and their thumbnails 96x64 and 64x96 pixels respectively.

All the test images were converted into grayscale and encoded using both solutions, the JPEG-inspired DNA codec and the proposed solution to robustify the thumbnails.

Given that NoRM uses Hamming distance as string similarity metric, it is designed to treat only substitution errors. In this experiment we only consider this type of error when simulating the noise. The substitution rate obtained in the last experiments with nanopore sequencing —later described in chapter 5 (table 5.9)— range from 0.18% and 0.22% for the best cases. Hence, in this experiment we simulate a 0.2% of substitution noise with random distribution. Note that errors were only added into the encoded DC coefficients and not the encoded AC coefficients to better assess the impact of robustifying the thumbnails. For each image, we perform 10 realisations of noise addition and decoding and show the averaged results.

Additionally, each of the noised sequences encoded with the robustified JPEG-inspired DNA codec is decoded multiple times, using the different solutions to decode the non-decodable errors described in the previous section: baseline decoding by zero, ACD, ACD2 and inpainting.

‡. The data set is available in https://www.kaggle.com/datasets/hsankesara/flickr-image-dataset

§. The data set is available in http://r0k.us/graphics/kodak/

¶. http://www.libpng.org/pub/png/

3.2.6 Results of the simulations

Figure 3.9 show the comparison of the averaged PSNR obtained after decoding the noisy data for the different methods. More precisely we compare:

1. The original JPEG-inspired DNA coder (original JPEG-DNA).
2. The proposed one replacing huffman encoding by VQ for the encoding of the DC coefficients (JPEG-DNA with VQ).
3. The proposed one robustifying the encoding of the quantization indices with NoRM (JPEG-DNA with VQ+NoRM)

The images encoded using the original JPEG-inspired DNA coder provided an averaged PSNR of 7.25 dB when introducing a 0.2% of substitution errors on the encoded DC coefficients. In most of the cases, the image lost the structure of the image partially (or even totally). Figure 3.10(a2 and b2) depicts an example of the visual quality of two decoded images.

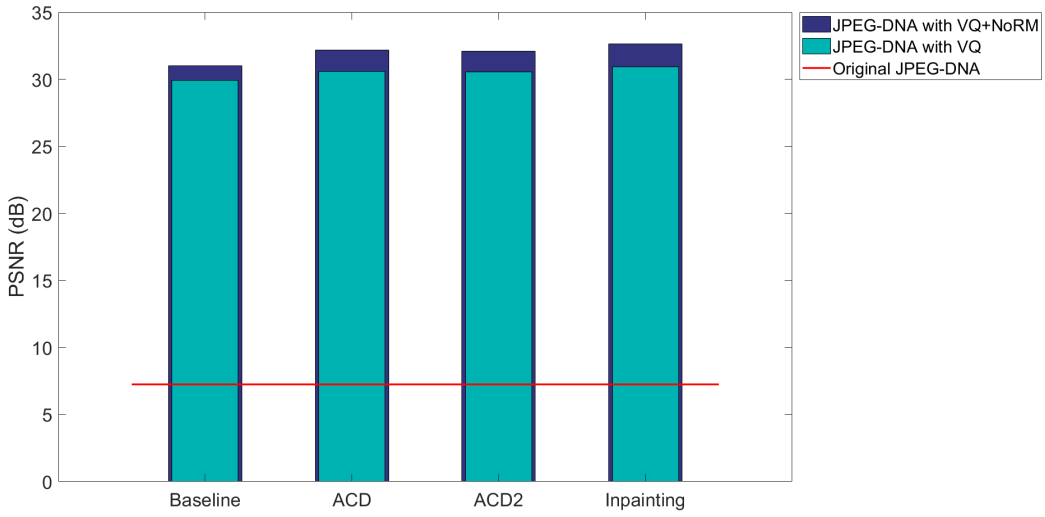


Figure 3.9 – Comparison of the averaged visual quality of the decoded images with each method. Results obtained after adding 0.2% of substitution in the encoded DC coefficients. More concretely, we compare the original JPEG-inspired DNA codec, the modified codec using VQ for encoding the DC values and the modified codec using VQ and NoRM. For the modified codec, we show the result for each of the proposed method for correcting the undecodable errors.

Thanks to modifying the encoding of the DC coefficients using VQ and Paircode instead of 3-nary Huffman and Goldman coding, the overall PSNR of the decoded images increased to 30.5 dB with an additional improvement of around 1.5 dB if the mapping of the quantization vectors to the codewords is robustified with NoRM.

To better compare the different methods for the decoding of the non-decodable errors, we present in table 3.2 the averaged PSNR improvement for each of the different methods proposed in section 3.2.4 compared to the original JPEG-inspired DNA coder. The worst results were obtained with the baseline and the best ones with the impainting solution. There was no significant difference between ACD and ACD2. Figures 3.10(c1 and c2) show the decoding of two images which

had been encoded with the proposed JPEG-inspired DNA coder adapted with VQ and NoRM and correcting with inpainting the non-decodable errors.

Table 3.2 – Averaged improvement on the PSNR of the reconstructed images encoded with the JPEG-inspired DNA coder robustified with NoRM.

	Baseline	ACD	ACD2	Inpainting
$\Delta PSNR$ (dB)	23.74	24.93	24.85	25.4

(a1) (a2) (a3)

(b1) (b2) (b3)

Figure 3.10 – Visual results of the decoded data with a substitution rate of 0.002 on the encoded DC values - (a1) and (b1) correspond to the original images. (a2) and (b2) correspond to images encoded with the original JPEG-inspired DNA coder. (a3) and (b3) correspond to the images encoded using the proposed JPEG-inspired DNA coder robustified with NoRM and inpainting on the non-decodable codewords.

3.3 Improving the visual distortion using inpainting

Applying efficient compression algorithms is key in the context of DNA storage due to the extremely high cost of DNA synthesis. Such is the case of the encoding solution proposed by our team in (Dimopoulou & Antonini, 2021) for storing digital images. Authors proposed the use of

the Discrete Wavelet transform followed by vector quantization on each DWT subband independently. Each quantized DWT subband is then encoded to obtain a DNA-like representation. The challenge arises with the noise introduced during the reading process (sequencing). Mechanisms such as error correction codes or consensus finding algorithms ease the impact of such errors in the sequenced data but some of them might persist and will be perceived as artifacts on the decoded DWT subbands. They will appear in different shapes and sizes depending on the type of error and the subband in which it occurred and will be propagated to the reconstructed image. In such cases, post-processing solutions such as inpainting have proven to be a promising candidate to repair the persistent damage.

Image inpainting is an approach to repair and restore damaged images in a visually plausible way. The main difference from other restoration techniques (for example haze-removal) is that in the case of image inpainting there is no information that can be gained from inside the damaged area. All the information that the algorithms can employ has to come from either the undamaged parts of the image or, at most, from the contour between these and the target area. There are many families of inpainting techniques, each better suited to handle different types of damage. The one selected for our work is a Texture Synthesis algorithm, a subcategory of more general Exemplar Based methods. These approaches aim to repair occlusions in the image by sampling and copying existing pixel values (referred to as patches) from the viable parts of the image onto the damaged ones. While these techniques are effective at repairing real world textures, they can cause artefacts due to the order in which patches are selected. As this could prove very problematic for our needs, we relied on the algorithm in (Criminisi, Perez, & Toyama, 2004). This region filling approach uses an edge-driven method to order the patch selection and filling process, ensuring that the propagation of structure into the damaged area is consistent and avoids both artefacts and texture overshooting.

3.3.1 Proposed inpainting in the Wavelet domain

Our algorithm is specialized to handle the type of damage we incur in when de-formatting and decoding DNA oligos that underwent a noisy sequencing process. As such, it differs from standard Texture Synthesis implementations in two ways. Firstly, it is built to be completely automatic. A series of damage identification steps try to identify and mark the target area of the image. Secondly, the inpainting is conducted on each single subband, obtained from the DWT decomposition, rather than on the whole image. Both of these differences are a product of the way in which we encode our images into DNA. As the subbands are formatted and encoded separately, the noise is applied on each subband rather than on the whole image. This causes problems when trying to rely on traditional inpainting. First and foremost, damaged pixels in the more meaningful subbands can end up affecting the whole image making the definition of a mask, either automatically or manually, impossible. The shape and size of the damage is also uncommon. Most inpainting algorithms are built to handle either large occluded areas or smaller, thinner damages. Our occlusions appear as either large spots and lines (caused by damage to a meaningful subband) or noisy checkerboard and crisscross patterns (caused by damages to the subbands carrying the details of the image). To sidestep all this, we act on the subbands before reconstructing them, which allows to employ a traditional inpainting approach in a more constrained environment and facilitates the damage detection.

3.3.2 Automatic detection of the damage in the wavelet subbands

The detection of the damaged areas is done using a 2-step algorithm, performed on each subband separately. The first step is done to detect errors caused by the erroneous decoding of single values, due to substitutions during the sequencing. This is done by comparing the value of each pixel to that of its neighbors. If the deviation between them is too high, it is likely that the pixel was damaged. This first step is not sufficient to detect extensive damage, for example in the case of one or more oligos being lost due to undecodable headers. In such cases, entire neighborhoods might be affected, and the first step is not able to reliably detect damaged areas. To handle this, a second detection step is performed. It can be observed that neighborhoods damaged in this way tend to have a very high internal variance. As such, during this second step the pixels whose neighborhoods present a standard deviation that is higher than the average are detected as potentially damaged. At the end of the two steps we will have a binary mask that can be overlaid over the original subband (see figure 3.11). This identifies the pixels that the inpainting algorithm recognizes as the target area, and which will in turn will be filled from the source area, effectively the rest of the image. The 2-step automatic detection of the damage is described in algorithm 3.1.

Definitions:

$I : \{p(x,y) \in [\![0;255]\!]^{n\times m}\}$ (Damaged image)
$O : \{o(x,y) \in \{0;1\}^{n\times m}\}$ (Mask)
τ_1: phase 1 threshold
τ_2: phase 2 threshold
N_p: 1-ring neighborhood of pixel p

Phase 1: Detection of single pixel errors (substitutions)

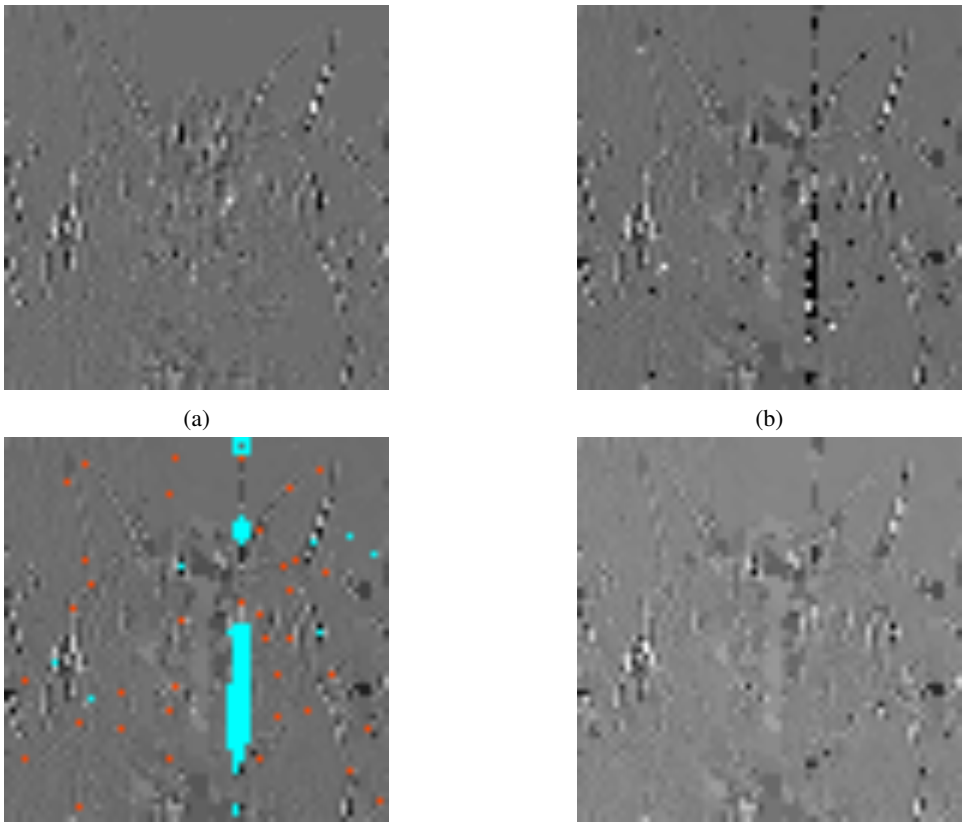
for all $x \in [\![1;m]\!]$ **do**
  for all $y \in [\![1;n]\!]$ **do**
    if $\frac{\sqrt{(p(x,y)^2 - \overline{N}^2_{p(x,y)}}}{\sigma(N_{p(x,y)})} \geq \tau_1$ **then**
      $o(x,y) = True$
    end if
  end for
end for

Phase 2: Detection of damaged neighborhoods (indels)

$$S_{mean} = \frac{\sum_{x=1}^{n}\sum_{y=1}^{m}\sigma(N_{p(x,y)})}{m\times n}$$

for all $x \in [\![1;m]\!]$ **do**
  for all $y \in [\![1;n]\!]$ **do**
    if $\frac{\sigma(N_{p(x,y)})}{S_{mean}} \geq \tau_2$ **then**
      $o(x,y) = True$
    end if
  end for
end for

Algorithm 3.1: Automatic damage detection in the wavelet domain



(a) (b) (c) (d)

Figure 3.11 – Visual results on a DWT subband. (a) Original subband, (b) Noisy subband after nanopore sequencing simulation, (c) Output of the automatic detection of the damaged areas (1st step in red, 2nd step in blue), (d) Inpainted subband.

3.3.3 Simulated results

As DNA synthesis remains a very expensive procedure, the algorithm was not tested on experimental data but on synthetic one. At this point, it is important to note that when the experiments were carried out, the state of the art on DNA sequencing simulators was rather limited and adapted to biological DNA, whose characteristics differ from the ones of synthetic DNA and may lead to differences in the noise and read length distributions. Instead, amplification with PCR and sequencing errors were simulated according to the latest information available about MinION sequencer.

3.3.3.1 Simulation of the sequencing noise

Assuming an almost error-free synthesis when the synthesized sequences are kept below 300 nts length, synthesis error was consider to be none.

On the other hand, the sequencing of the DNA using nanopore technologies remains the main source of errors in the workflow presented in this work. The accuracy of this technology has improved from around 85% when the nanopore sequencing was first introduced to 95% in 2021. We adjusted the rates of substitutions, insertions and deletions provided in (Zeng et al., 2020) to

fit the decreased error rate of nanopore sequencing, resulting in the following values: 2.3% of deletions, 1.01% of insertions and 1.5% of substitutions. The previous percentages were used for the simulation of the sequencing noise, in which 80% of the noise was concentrated on the first and last 20nt of each oligo (Jain et al., 2015). It is important to denote that the simulation of the noise based on the statistics of real nanopore sequencing experiments constitutes a proof of concept of the methods presented in this work.

3.3.3.2 Workflow of the experiment

The image was compressed, quantized, encoded and formatted into short oligos. We then simulated the nanopore sequencing noise and introduced it to the formatted oligos by creating 200 noisy copies of each input oligo. The purpose of this last step is to mimic the process of PCR amplification and production of multiple noisy reads by the nanopore sequencer[‖]. The result of this procedure is a set of multiple copies of the encoded oligos containing different error realisations as would occur in a real wet lab experiment. To decode the noisy data we start by clustering (or grouping) the noisy copies according to their headers. Each cluster is then cleaned by discarding the noisy oligos with high average Levenshtein distance to their cluster (which could be due to either low quality of the oligo or an erroneous header). Afterwards, the remaining oligos in each cluster after filtering are aligned and a consensus sequence is retrieved from each cluster as the most representative version of each oligo. The consensus algorithm is based on majority voting, assigning to each position inside the sequence the most frequent symbol along the cluster as depicted in figure 3.12. Using those consensus sequences we reconstruct a noisy version of the input image which will then be post-processed for smoothing the damaged areas. We used the proposed algorithm for the automatic detection of the damage in each DWT subband and the selected areas are inpainted to provide the final result of the workflow.

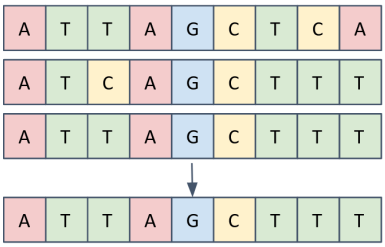


Figure 3.12 – Example of classical consensus.

3.3.3.3 Results of the simulation

For our experiments, we carried out the process described above using a compression quality of 4.9708 bits/nt (1.61 nt/pixel). Figure 3.13(a) shows the quantized image after compression. This image has a PSNR = 48.12 dB and a SSIM = 0.991 compared to the original uncompressed image. Figure 3.13(b) shows the decoded image with sequencing noise and 3.13(c) corresponds to the final image after inpainting. The post-processing of the image led to a PSNR = 38.7 dB and

‖. It has been shown later that PCR-amplified data may follow an uneven distribution, partly affected by the GC content of the DNA sequence (see Section 2.2).

a SSIM = 0.94, which constitute an improvement of 2.5 dB and 0.2 respectively compared to the original quantized image with sequencing noise.

(a) (b) (c)

Figure 3.13 – Visual results of the experiment: (a) Quantized image without sequencing noise, (b) Visual impact of sequencing noise in the image encoded, (c) Post-processed image using inpainting.

3.4 Conclusions

In this chapter we propose a method to robustify a JPEG-inspired coder for DNA data storage with the goal of ensuring the decodability of the stored data even in the presence of errors and degradation. The experiment showed that using VQ and NoRM improved the encoding of noisy data compared to the original JPEG-inspired DNA coder. The encoding using VQ led to a significant increase in PSNR and further improvement was achieved by using NoRM. The best results were obtained using the inpainting method to correct non-decodable errors. These findings demonstrate that modifying the encoding methods can significantly improve the quality of the decoded images.

We also propose using image inpainting to repair the persistent damage. The inpainting technique used is a Texture Synthesis algorithm. We present a method for the automatic detection of the damaged areas on the wavelet level through a two-step algorithm which creates a mask to guide the inpainting process. The proposed solution demonstrates promising results in restoring the decoded images with reduced visual distortion. This work has been published in (Gil San Antonio, Piretti, Dimopoulou, & Antonini, 2021).

CHAPTER 4

From sequenced to decoded: handling nanopore sequencing noise

4.1 Introduction

As the DNA data storage pipeline involves biochemical procedures prone to introduce errors in the DNA molecules, it is of high importance to develop decoding techniques that are able to predict, detect and even correct the sequenced data. As mentioned in section 2.1.2, PCR is a very easy and cheap way of introducing more redundancy in the data by producing many copies of the initial oligos. Some works have shown that when sequencing with Illumina (see section 2.1.4.1) the selection of the most frequent sequences suffices to have an accurate representation of the initial data. However, when applying other sequencing methods such as nanopore (see section 2.1.4.2), errors appear in most of the copies and in different positions. In those cases, the oligo selection step must be adapted to this new scenario, discarding highly corrupted reads and clustering the rest to infer the consensus sequences for decoding.

The use of techniques capable of handling sequencing noise is crucial to ensure the decodability of the data. The use of noise simulators is widely extended in order to speed up the development and testing of new tools addressing DNA data storage. Although several sequencing simulators have been released during the past years, their noise models have been mostly extracted from biological data and not short synthetic DNA. This constitutes a challenge for testing new algorithms without having to continuously carry out expensive wet-lab experiments.

In this chapter, we propose a method for the decoding of nanopore-sequenced reads including oligo selection and a consensus adapted to non-complete dictionaries. Additionally, using the error rates extracted from the experimental data, we parameterize and test the accuracy of a new DNA data storage simulator.

4.2 Decoding schema for nanopore-sequenced data

In (Dimopoulou, Antonini, et al., 2019), our team proposed a method for the specific encoding of digital images into DNA which includes compression to control the DNA synthesis cost (DWT and quantization of each produced subband) and a biologically constrained encoding that respects

the restrictions imposed by the process of DNA sequencing. For a detailed explanation of the encoding algorithm, readers can refer to the aforementioned publication. The performance of the proposed encoding algorithm was tested on a biological experiment.

In this experiment, the team encoded and formatted 2 different images depicted in figures 4.3(a0) and 4.3(b0) of size 128 by 128 pixels and 120 by 120 pixels. The images were lossy compressed with a rate of 2.68 bits/nt (2.985 nt/pixel) and 1.78 bits/nt (4.49 nt/pixel) respectively. The resulting encoded images are depicted in figures 4.3(a1) and 4.3(b1). The formatting of the images led to a total amount of 662 and 875 oligos respectively. All the oligos had a length of 138 nts including primers, which are short special sequences required by the sequencer at both ends of the DNA strands. In both cases, the payload in 11 oligos contained only headers encoding important information about the characteristics of the image and the parameters of the encoding. The rest of the oligos contained the encoded data itself. The formatting of the oligos is depicted in figure 4.1. The payload of each formatted oligo had a length of 84 nts.

The formatted sequences were synthesized by the company Twist Bioscience [*] and stored by Imagene [†] in special capsules (called DNAshell [‡]) that allow long preservation of the DNA. For the decoding, the DNA strands were sequenced using the Illumina Next Seq machine, allowing a perfect reconstruction of the stored images.

After two years of storage, the synthesized oligos were re-sequenced using both, Illumina NextSeq and MinION nanopore machine. More precisely, nanopore sequencing was performed using SQK-LSK109 sequencing kit and MinKNOW 3.6.8 software. Basecalling was performed in 4000 event batches using Guppy 3.2.10. The sequencing step led to a total amount of 3395789 raw reads. Despite being a very promising sequencing platform due to the numerous advantages that it offers, MinION introduces a higher error rate during the reading process (ranging from 3% to 5% at the time when the sequencing was carried out) which jeopardizes the correct decoding of the data.

Due to the low error rate introduced by Illumina NextSeq and the nature of such errors, the selection of the most frequent reads as the most reliable (i.e. less noisy) led to a perfect reconstruction of the data (Dimopoulou, Antonini, et al., 2019 ; Dimopoulou, 2020). However, when using the MinION sequencing machine, due to the higher error rate introduced during the process, the decoding is not that trivial and the aforementioned approach is no longer reliable.

Aiming to deal with the increased error rate, this work proposes a decoding workflow to deal with the MinION sequencing noise which allows to recover the original stored data. The decoding process is described in the following paragraphs.

4.2.1 Decoding of nanopore-sequenced reads

Prior to the sequencing step and aiming to add extra redundancy to deal with the errors that MinION introduces, the initial oligos were replicated into many copies through Polymerase Chain Reaction (see section 2.1.2). Hence, the result of the DNA sequencing is a pool of reads which includes many noisy copies of each reference sequence. This noise comes in the form of substitutions, insertions and deletions of nucleotides, which affects dramatically both ends of the DNA strands. The resulting reads also contained the adapters needed for sequencing, which were removed using the library Porechop (Porechop, 2018). After trimming the adapters, the decoding

*. https://www.twistbioscience.com/
†. http://www.imagene.fr/
‡. http://www.imagene.fr/dnashell-rnashell/dnashell/

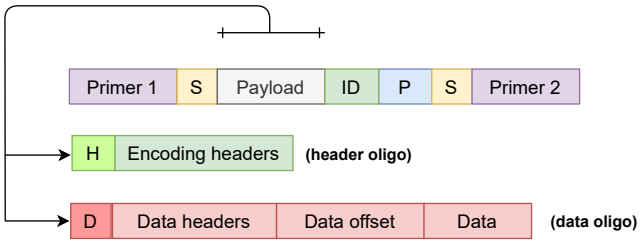


Figure 4.1 – Format of the oligos - All oligos contain primers that are needed for the sequencing: *S* denotes the sense nucleotide which determines whether a strand is reverse complemented when sequenced. *P* is a parity check nucleotide while the *ID* is an identifier of the image so to be distinguished from other data that may be stored. The payload can either contain encoding headers only which hold information about the image characteristics and the encoding parameters used (header oligo), or it can contain some data headers and an offset to denote the position and nature of the data field that follows (data oligo).

procedure can be described in five steps:

Step 1: Read filtering
The introduction of insertions and deletions at different rates creates significant variations in the length of the output reads. Thus, the first step of the decoding phase is to clean the data, discarding those reads that are highly corrupted due to the noise and will not contribute to the improvement of the results. Consequently, reads are filtered by length, keeping only those reads whose size belongs to the interval $L \pm 10$ nts, being L the expected size of the reads.

Step 2: Read selection
The second step corresponds to the retrieval of the reads corresponding to the data we want to decode as the pool of reads does not only contain one image but several as well as other kind of data. To do so, we need prior knowledge about the identifier of the stored data encoded in some header field of the oligos, the ID field (see figure 4.1) and its position. As a consequence of the errors introduced by sequencing, the position of the identifier might be shifted. Hence, in order to retrieve as many reads as possible, we look for the identifier not only in the original position but within a range around it.

Step 3: Clustering
Once the reads which correspond to the data we aim to decode have been retrieved, they are clustered according to their headers. All the reads with non-decodable headers are discarded as they cannot be assigned to any cluster.

Step 4: Consensus finding
The last step before the decoding of the data is the selection of the most representative sequence of each cluster. One of the most widely used algorithms for consensus finding is based on majority voting, assigning to each position inside the sequence the most frequent symbol along the cluster (figure 4.2 left).

Step 5: Decoding
Finally, the quaternary sequence obtained after consensus is transformed back to its initial repre-

sentation to reconstruct the stored information.

The results are depicted in figures 4.3(a2) and 4.3(b2). Even though we were able to retrieve reads corresponding to all the reference oligos, the information decoded from the header oligos that contain important parameters regarding the decoding was corrupted, compromising the decoding of the rest of the data. To allow the decoding, we make the assumption that those parameters are known to the decoder. Although this might not be a realistic scenario, the synthesized oligos had been encoded to be read by a more accurate sequencer (Illumina Next Seq) and thus, those header oligos did not need stronger protection to be correctly retrieved. One solution to this problem when sequencing with MinION —which has a much higher error rate— would be protecting those important fields using of error correcting codes as for example error correcting DNA barcodes (Ashlock & Houghten, 2009 ; Gil San Antonio, Piretti, et al., 2021). The above results prove that despite the assumption of knowledge of the header oligos there is still too much noise corrupting the decoded data. Therefore, it is clear that we are in need of applying a more sophisticated consensus finding algorithm.

4.2.2 Consensus for fixed-length encoding

As shown in the section 4.2.1 it is clear that the nanopore sequencer introduces much noise in the visual quality of the decoded images. In this section, we propose an advanced decoding method which takes advantage of the encoding proposed in (Dimopoulou, Antonini, et al., 2019) in order to improve the quality of the results.

To better understand the proposed consensus adapted to the constrained dictionary described in the following sections, readers should refer to section 3.2.2 where we introduced Paircode, the constrained quaternary code used to encode the images.

As shown in figure 3.2, Paircode uses a dictionary of 4-ary codewords that are constructed using known pairs of symbols. Consequently, according to this algorithm there are some codewords that are excluded from the codebook. In case of a sequencing error (insertion, deletion or substitution), it is probable that a correct codeword is transformed into one of those codewords that are not included in the code, thus denoting an error. This fact can be used for improving the consensus finding algorithm so to provide better estimation of the correct oligos.

In this section, we propose a new implementation of this algorithm which is based on the same principle of majority voting but acts on DNA codewords rather than single nucleotides (see figure 4.2).

The consensus by codeword algorithm is applied to each cluster of reads and is briefly described as follows:

1. Divide each read into the different codewords
2. For each codeword position in the strand, sort codewords by frequency
3. Select as consensus the most frequent decodable codeword (i.e. the most frequent codeword that exists in our dictionary)

With this new consensus we allow an extra step of error correction to find a better consensus by ensuring that the final estimation does not contain undecodable codewords. Given the uniform distribution of the sequencing errors, the most frequent decodable codeword will be the most probable correct one, although there might be exceptions such as in the cases where coverage is too low to allow inferring an accurate consensus. The consensus finding applied on codewords is

further described in algorithm 4.1. Additionally, we proposed another consensus finding approach for the cases in which, due to formatting, a codeword is truncated into two consecutive oligos, i.e. the beginning of the codeword will appear at the end of oligo i and the end of the codeword will appear at the beginning of oligo i+1. The consensus finding for truncated codewords is described in algorithm 4.2.

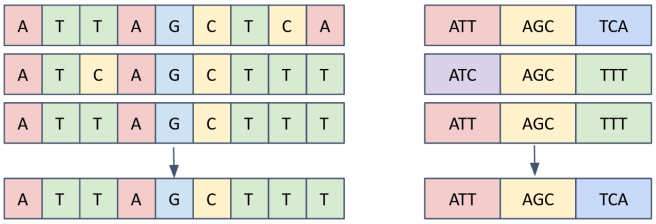


Figure 4.2 – Comparison of two methods for consensus finding. Left: majority voting on single nucleotides. Right: majority voting on DNA codewords, "TTT" is non-decodable as it does not exist in our dictionary (see figure 3.2), therefore, it is discarded when building the consensus even though it appears with a higher frequency.

Definitions:

1: $\mathcal{C}^*$: Set of L codewords of length l (Constrained codebook)
2: $\mathcal{S}_{dec}$: Set of quaternary codewords of length l
3: $d_L(x, y)$: Levenshtein distance between codewords x and y

Consensus finding

1: Initialise: $f_{dec} = \{f_{dec}(i) = 0, i \in [\![1; |\mathcal{C}^*|]\!]\}$
2: **for all** $i \in [\![1; |\mathcal{S}_{dec}|]\!]$ **do**
3: **if** $\mathcal{S}_{dec}(i) \in \mathcal{C}^*$ **then**
4: $j^* = \arg\min_j(d_E(c_j, S_{dec}(i)))\ \forall c_j \in \mathcal{C}^*$
5: $f_{dec}(j^*) = f_{dec}(j^*) + 1$
6: **end if**
7: **end for**
8: **return** $\mathcal{C}^*(\arg\max(f_{dec}))$

Algorithm 4.1: Consensus finding for non-complete codebooks

4.2.3 Results

Figures 4.3(a2) and 4.3(b2) depict the decoded images when applying classical consensus on single nucleotides. The visual results when using the proposed method for consensus finding are depicted in figures 4.3(a3) and 4.3(b3). For both images, the PSNR had a significant improvement of around 15 dB when using the proposed consensus finding algorithm, leading to a notable improvement on the visual quality of the reconstructed images. Note that the PSNR of the decoded images has been computed respect to the lossy-compressed encoded images (4.3(a1) and 4.3(b1)).

Definitions:

1: $\mathcal{C}^*$: Set of L codewords of length l (Constrained codebook)
2: T_{start}: set of symbols of length l_s representing the beginning of the codeword for consensus
3: T_{end}: set of symbols of length l_e representing the end of the codeword for consensus
4: U_{start}: set of unique symbols in T_{start}
5: U_{end}: set of unique symbols in T_{end}
6: f_{start}: frequency of each symbol from U_{start} in T_{start}
7: f_{end}: frequency of each symbol from U_{end} in T_{end}

Consensus finding

Initialise: $max\_score = 0$
for all $i \in [\![1; |U_{start}|]\!]$ **do**
  for all $j \in [\![1; |U_{end}|]\!]$ **do**
    $u = concatenate(U_{start}(i), U_{end}(j))$
    if $u \in \mathcal{C}^*$ **then**
      $score = \frac{f_{start}(i)}{|T_{start}|} + \frac{f_{end}(i)}{|T_{end}|}$
      if $score > max\_score$ **then**
        $max\_score = score$
        $consensus = u$
      end if
    end if
  end for
end for
return $consensus$

Algorithm 4.2: Consensus finding for truncated codewords

(a0) PSNR = Inf
(a1) PSNR = 32.5 dB, Rate = 2.68 bits/nt
(a2) PSNR = 21.7 dB
(a3) PSNR = 40.5 dB
(b0) PSNR = Inf
(b1) PSNR = 29.67 dB, Rate = 1.78 bits/nt
(b2) PSNR = 13.2 dB
(b3) PSNR = 30.9 dB

Figure 4.3 – Visual results of the decoded data - **(a0)** and **(b0)** are the original images. **(a1)** and **(b1)** correspond to the stored images (reference). **(a2)** and **(b2)** correspond to MinION sequencing and classical consensus based on Majority Voting in single nts. **(a3)** and **(b3)** correspond to MinION sequencing and our novel consensus algorithm based on Majority Voting in codewords

4.3 Validation of a nanopore sequencing simulator for DNA data storage

In the past years, several works have introduced sequencing simulators aiming to ease the implementation of new algorithms targeting nanopore-sequenced data. Such simulators allow testing while developing new tools thanks to their speed, low cost and high throughput. Commonly, simulators generate noisy reads using a model error profile extracted from experimental data. The introduction of the errors can be done directly by modifying the bases of the DNA sequences (Baker, Goodwin, McCombie, & Ramos, 2016 ; Faucon, Balachandran, & Crook, 2017 ; Yang et al., 2017) or by simulating the electrical signals and allowing the basecaller to introduce the errors while translating it into a sequence of nucleotides which provides a more realistic scenario (Y. Li et al., 2020 ; Rohrandt et al., 2018). The main challenge when using these simulators for DNA data storage applications lies in the fact that their error models are generated from the sequencing of biological and thus, longer reads than in the case of synthetic DNA, whose length is limited to 300 nts. This work constitutes a first proof of concept of a new DNA storage error simulator proposed in (Alnasir et al., 2022) addressing synthetic DNA in the context of DNA data storage.

In the following sections we introduce the simulator and its capabilities and describe how the error rate of nanopore sequencing has been estimated for the specific case of synthetic DNA and how it has been used to parameterize the simulator.

4.3.1 The simulator

The proposed simulator models errors in all phases of DNA storage, i.e., synthesis, storage, PCR (for amplification) and sequencing. It takes as input the sequences encoding the information and returns the sequences incorporating simulated errors. We discuss the errors and configuration of each phase in the following. Error probabilities to configure the simulator can either be determined experimentally or be taken from related work (Heckel, Mikutis, & Grass, 2019).

4.3.1.1 Synthesis

DNA synthesis is a linear process, i.e., one nucleotide is added after another (in the 5'-3' direction). Errors occur during the physical assembly of the nucleotides. As such, errors are evenly or uniformly distributed across the synthesised sequence, meaning that an error is as likely to occur on the first nucleotide as it is in the middle of the sequence. Our simulation consequently passes over the whole sequence and at each nucleotide decides whether an error should occur based on the error probability.

Different errors such as deletions (absence of a nucleotide or accidental, early termination), insertion (additional nucleotide) or substitution (a different nucleotide than intended is added) can occur. The simulator is configurable in terms of likelihood of an error occurring and, if an error occurs, the likelihood of the type of error (and in case of a substitution, the likelihood of each type, e.g., A substituted by G).

Although insertions and substitutions are uniformly distributed across synthesised sequences, deletions are tail favoured. The simulator hence compensates for that by simulating an error with the same likelihood across nucleotides of the sequence, but if an error occurs, the chances for a deletion are higher at either end of the sequence.

4.3.1.2 Storage

Errors can also be introduced during storage. Increased humidity or temperature can drastically shorten the lifespan of DNA. Experiments with protocols for accelerated ageing by way of increased temperature (to speed up decay and thus simulate a storage time of multiple half-lives) have been carried out to understand the sources of errors.

The decay of DNA is modelled like standard radioactive decay. However, instead of removing nucleotides, the bonds between nucleotides are simply broken resulting in broken sequences. The broken sequences are no longer readable as the forward and reverse primer are no longer located on the same strand.

The simulator uses a derivation of the Arrhenius equation with its values from studies on dated fossils (Allentoft et al., 2012) to model a breakage/decay event on a sequence. The only parameter needed to be configured is the storage duration in years.

Simulating errors in storage starts with the sequences resulting from the synthesis simulation. The process is iterative, meaning that in the event a sequence is fragmented due to decay, both (or all) fragments are added back to the pool of sequences, meaning that they can be broken again.

4.3.1.3 PCR

Polymerase Chain Reaction is a method widely used to amplify sequences, i.e., to rapidly make millions to billions of copies of the sequences before sequencing. PCR is typically done in multiple cycles and at every cycle, the number of sequences is doubled, i.e., two identical sequences are produced for every sequence.

Standard experimental protocols suggest to run 40 PCR cycles. Doing so in a simulation quickly renders the simulation computationally intractable as it produces too many . It is, therefore, necessary to be able to reduce the size of the PCR output to a constant number. By taking a uniform random sub-sample after each PCR cycle, the simulator keeps a constant size in addition to a general representation of the error distribution. The number of PCR cycles in the simulator is configurable.

The downside of taking a random sub-sample is that there will be a bias towards the initial sequences for PCR phases with small cycles. Due to the exponential nature of PCR, the sequences generated during the first cycles will have a disproportionate representation in the sub-sample compared to the later cycles. Increasing the number of cycles will help reduce the bias and help arrive at a more randomly distributed final sub-sample.

4.3.1.4 Sequencing

The approach to modelling sequencing is very similar to modelling the synthesis phase. More specifically, the same errors as in synthesis can occur, insertions, deletions and substitutions due to misreads. Generally, across different sequencing technologies, the distribution of errors across sequences are uniform - as is the case for synthesis.

Thus, the implementation of the simulation of sequencing is the same as synthesis but the probabilities for substitution, deletions and insertions errors and their respective transitions (for substitutions) need to be configured.

4.3.2 Estimation of the nanopore sequencing noise

Although several works have provided studies about the error rates introduced by nanopore sequencing, most of them focus on the sequencing of long DNA strands (thousands of bases) as this technology targets the sequencing of complete genomes. Additionally, state of the art simulators expect a genome as reference, which will be sub-sampled following some nanopore read length distribution model and then corrupted by adding errors in form of substitutions, insertions and deletions. On the contrary, our work uses synthetic DNA, limiting the length of the oligos to 300 nts at most and making the sampling step unnecessary. Therefore, to adapt the characteristics of the noise introduced by the simulator when sequencing short DNA strands, we have computed the noise rates from the nanopore-sequenced reads storing two images presented in section 4.2.

4.3.2.1 Error rates

For the estimation of the error rates, we first mapped each read to its reference using minimap2 (H. Li, 2018) and computed the Levenshtein distance between each read and its reference considering only those reads that could be unequivocally mapped to a reference. Nanopore adapters were not considered when computing the distances. In the same way, we estimated its three different components (substitutions, insertions and deletions):

— Total error rate: 0.0686
— Substitution rate: 0.0253
— Insertion rate: 0.0179
— Deletion rate: 0.0255

In addition, we also computed the error rates for the oligos encoding each of the two images (figure 4.3) independently but no significant variation was found. Figure 4.5 shows the distribution of the different noise components.

4.3.2.2 Parameterization of the simulator

As described in the previous section, the synthesized DNA strands for our wet-lab experiment had a length of 138 nts and they were stored for two years in a sealed capsule. Considering that the error rate of DNA synthesis is almost negligible when the synthesized oligos do not exceed 300 nts length and that the capsule prevents its contact with water and oxygen keeping the DNA intact during the storage period, the only significant source of error is the process of sequencing. Therefore, in our simulations we only considered sequencing noise. More precisely, the estimated rates of substitutions, insertions and deletions from the experimental data were used as target rates. Finally, the coverage was selected so to match the coverage from the experimental data.

4.3.3 Comparison of the results

We tested the performance of the simulator by running 50 realisations for the error rates provided in section 4.3.2.1 and averaged the results. The reference sequences used to feed the simulator were the ones as the ones presented in section 4.2.

In average, the simulations led to a total amount of 3396770 reads with the following error rates:

— Total error rate: 0.0666
— Substitution rate: 0.0242

— Insertion rate: 0.0169
— Deletion rate: 0.0255

For the decoding of the simulated reads we followed the same process described in section 4.2.1. Figures 4.4(a3) and 4.4(b3) depict an example of the reconstructed images from one simulation run.

In average, the decoded images from the simulated sequenced reads provided a Peak Signal-to-Noise Ratio (PSNR) of 40.23 dB and 32.7 dB, respectively. The results obtained from the simulations are comparable to the experimental ones in terms of PSNR as well as the visual quality of the decoded images for all the runs.

We have also compared the distribution of the errors introduced by the MinION and the simulator. Figure 4.5 depicts the Probability Density Function (PDF) for the different error types.

It is important to note that the only parameterization required for the simulations is the average error rates of insertions, deletions and substitutions. Therefore, while the mean error rate of the simulator can be controlled, the standard deviation of the error rate can vary. This fact explains the reason why the variability of the error in the experimental reads is higher than the one computed by the simulations. Nevertheless, this difference in the standard deviation does not have a significant impact on the reliability of the simulator results.

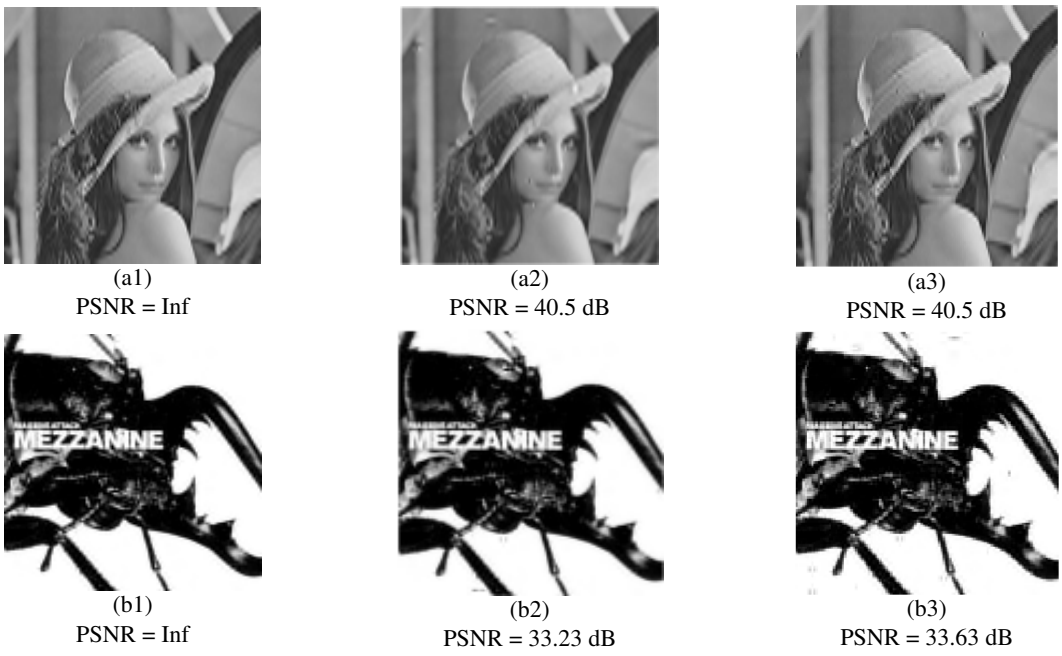


Figure 4.4 – Visual results of the decoded data - (a1) and (b1) correspond to perfectly decoded images. (a2) and (b2) correspond to MinION sequencing and our novel consensus algorithm based on Majority Voting in codewords presented in section 4.2.2. (a3) and (b3) correspond to an example of the decoding of simulated reads.

4.4 Conclusions

In this chapter we propose a decoding method for nanopore-sequenced data which takes advantage of the encoding that has been introduced in our previous works for the storage of images

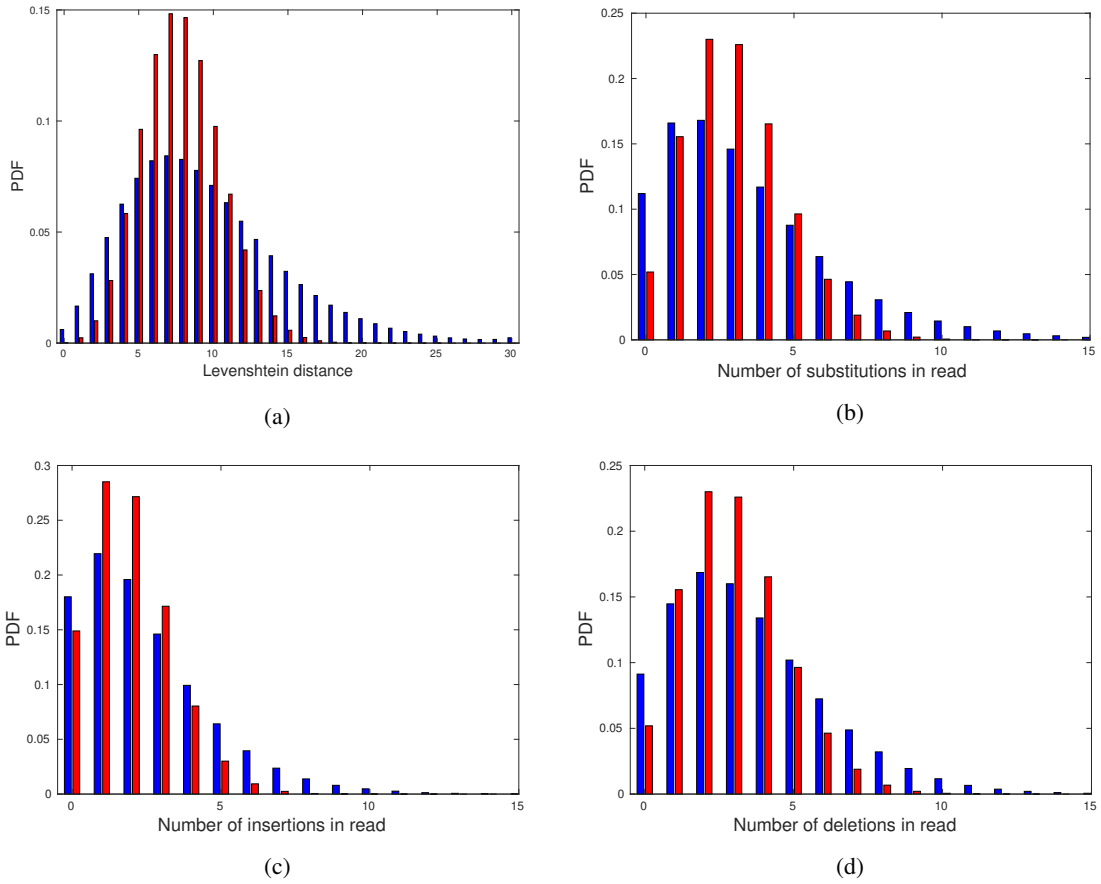


Figure 4.5 – Comparison of the Probability Density Function (PDF) for each kind of error between the reads from the wet-lab experiment (blue) and simulated reads (red) - (a) Levenshtein distance, (b) substitutions, (c) insertions and (d) deletions.

into synthetic DNA. To this end we have sequenced the data encoded and synthesized 2 years ago using two different sequencing technologies: Illumina Next Seq (accurate but slow and expensive) and MinION (real-time, user friendly and affordable but error prone). Our results prove that the proposed decoding can significantly improve the quality of the reconstruction. It is important to note that even though the decoding is not perfect, this proposed method was proven to be very promising for the extremely high error rate of Nanopore sequencing and the results might be even improved by adapting the encoding algorithm to this sequencing technology and by further strengthening the error correction method. This work has been published in (Gil San Antonio, Dimopoulou, Antonini, Barbry, & Appuswamy, 2021).

We also introduce in this chapter a very first demonstration of the potential of a new simulator that models the full DNA data storage channel. Although in this study we have only assessed the performance of the nanopore sequencing module (mainly due to the lack of experimental data), future works will focus on the evaluation of the rest of the existing modules in this simulation tool. Nevertheless, these first results are highly promising as they are comparable to the experimental ones in terms of PSNR and visual quality of the decoded images, proving the capability of the simulator to reproduce the errors introduced during nanopore sequencing in short synthetic DNA

strands. This work has been published in (Gil San Antonio, Heinis, Carteron, Dimopoulou, & Antonini, 2021).

CHAPTER 5

Wet-lab experiment

5.1 Description of the experiment

During the past two years, Mediacoding team has proposed three encoding solution for the storage of digital images into synthetic DNA. The first solution was proposed in (Dimopoulou, Antonini, et al., 2019), a first attempt to optimally compress an image into DNA by using a simple scalar quantizer. The proposed solution was validated in a wet-lab experiment (see chapter 4), generating real synthetic oligonucleotides that were stored in a DNAShell capsule. Although very promising, results could be improved by using some more sophisticated method of compression. Hence, a second fixed-length solution was proposed in (Dimopoulou & Antonini, 2021) using Vector Quantization and a nucleotide allocation algorithm for the optimization of the compression. Although fixed-length encodings are generally more robust to errors, the high cost of current DNA synthesis methods encouraged the development of a variable-length solution, which outperforms the first one in terms of compression efficiency. The variable-length solution, presented in (Dimopoulou et al., 2021a), was inspired by the main workflow of classical JPEG, and adapted for quaternary codes rather than binary.

The goal of this wet-lab experiment is to test the robustness of the fixed-solution using VQ and the JPEG-inspired variable-length encoder. Additionally, we test a third, simpler scenario in which the images are compressed using classical JPEG and the binary output is transcoded into a quaternary DNA-like sequence. The DNA coders are further described in the following sections.

The images used in this experiment were obtained form the Kodak Lossless True Color Image Suit (`http://r0k.us/graphics/kodak/`). More concretely, we selected the two images depicted in figure 5.1, both of them of size 768x512 pixels and stored in PNG format.

kodim15.png

kodim23.png

Figure 5.1 – Images selected for the experiment.

Although generally in lossy compression algorithms users fix the compression rate, in this experiment we fixed the distortion (or image quality), which provides a different compression rate for each encoding. We chose to do for comparison purposes since we aim to compare the visual quality of the decoded images after undergoing the noisy DNA storage channel.

5.2 Encoding and formatting

In this section we describe the three different coding techniques used to encode the digital images into a DNA-like representation: a fixed-length solution based on vector quantization, a variable-length encoding inspired on the workflow of classical JPEG, and a simple transcoding of the binary output of JPEG-compressed images. It also contains the information regarding the formatting of the long encoded DNA strands into smaller fragments that can be physically synthesized with current DNA synthesis methods.

5.2.1 VQ-based encoding

The algorithm

In (Dimopoulou & Antonini, 2021), a fixed-length coding for storing digital images into DNA was proposed. The first step of the algorithm is image compression, where the input image is compressed using a Discrete Wavelet Transform and each resulting subband is independently quantized with vector quantization. In order to optimize the compression we used a source allocation algorithm. The source allocation algorithm —nucleotide allocation— provides the optimal quantization parameters for a given rate. The purpose of the nucleotide allocation is the minimization of the total image distortion such that the total rate is lower than a given target rate. In other words, the goal is to find the optimal parameters for the quantization of each wavelet subband to achieve the best possible image quality for a given compression rate. The codebook of vectors for VQ was constructed using the images from the Kodak Lossless True Color Image Suit data set, excluding the two selected for the experiment.

For the mapping of the input vectors obtained from the VQ algorithm to the codewords from the quaternary code, we used the algorithm described in 3.2.3 which ensures that the impact of an error in the quaternary sequence is minimized. Briefly, the idea is to map quantization vectors with a small Euclidean distance to codewords which have a small Hamming distance. In this way, in case an error occurs during sequencing and assuming that the sequencing noise is reasonably small, a correct codeword will be transformed to another one which will have a small Hamming distance with the correct one. Consequently, the decoded erroneous vector will have a small Euclidean distance compared to the correct one, reducing the visual distortion that an error creates in the decoded image.

It is important to mention that the wavelet decomposition of the image separates the high frequencies —features such as details, edges, noise, etc.— from the low frequencies, allowing for a sparse representation of the signal (i.e the energy of the signal is found in relatively few non-null coefficients, leading to a more compact representation of the signal and paving the way for compression). As a result, high frequency subbands will contain many repeated coefficients. In the case of having a one-to-one mapping, the encoder will translate all these repeated values into the same DNA codeword, hence, creating patterns. As it was mentioned in section 2.2, patterns can decrease the accuracy of sequencing and should be avoided.

If we define $\Sigma = \{1, 2, \ldots, K\}$ with $|\Sigma| = K$, a set of quantization indices to be encoded into a set $\mathcal{C}^* = \{c_1, c_2, \ldots, c_L\}$ of L quaternary codewords (with $L \geq K$) of length l, the mapping step of the encoding algorithm can be expressed as $\Gamma : \Sigma \mapsto \mathcal{C}^*$ associating an index in Σ to one or more possible codewords in $\mathcal{C}^*$. The simplest scenario is a one-to-one mapping in which each index in Σ is mapped to one codeword from $\mathcal{C}^*$ as depicted in figure 5.2(a), with $K = L$ if one wants to maximize the compression efficiency. This is the case for the low frequency subband, whose values are more homogeneously distributed and the creation of patterns less probable. However, for those subbands whose content can lead to pattern repetition, a one-to-many mapping would be more appropriate. Aiming to minimise the impact of the double mapping on the rate-distortion of the encoded data, only the most frequent indices in Σ are assigned to two codewords from $\mathcal{C}^*$. Figure 5.2(b) depicts this second scenario in which the codebook $\mathcal{C}^*$ is divided into two sets of sizes K and K' with $K' = \frac{1}{4}K$ and $K + K' = L$. The first set is used to map every index to a codeword and the second smaller set is used to provide a second representation for the K' most frequent indices in Σ. Additionally, for the extreme cases we also propose a third mapping scenario depicted in figure 5.2(c) in which $\mathcal{C}^*$ is divided into one set of size K and two sets of size K'' with $K'' = \frac{1}{8}K$ and $K + 2K'' = L$. In this case, the K'' most frequent indices in Σ will be mapped to three different codewords from $\mathcal{C}^*$.

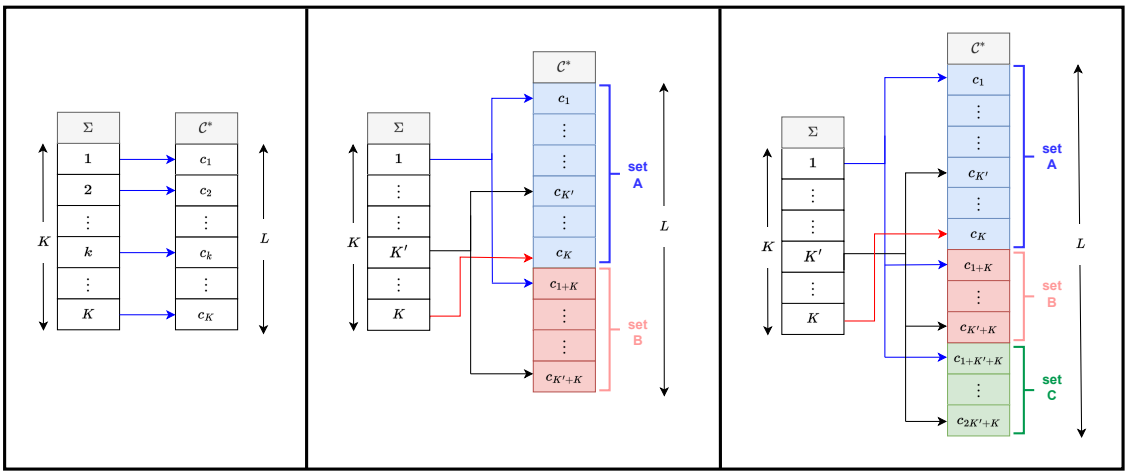


Figure 5.2 – Mapping strategies. (a) Single mapping, (b) double mapping and (c) triple mapping.

Figure 5.3 depicts the coding/decoding workflow. This fixed-length solution was used to encode the two images shown in figure 5.1. Currently, one of the main limitations of DNA synthesis is the high cost of the process. Hence, prior to encoding both images were converted into grayscale to reduce the cost of the experiment. The images were compressed at a rate of 4.6 bits/nt (1.74 nt/pixel), which lead to a PSNR of 39.4 dB for image "kodim15.png" and 41 dB for "kodim23.png" (see figure 5.4).

Formatting

Once the images were encoded into a quaternary DNA-like sequence, they were formatted into shorter chunks due to the constraints of DNA synthesis, which limits the length of the synthesized strands to 300 nts at most. For our experiment, we formatted the encoded data into oligos of 200

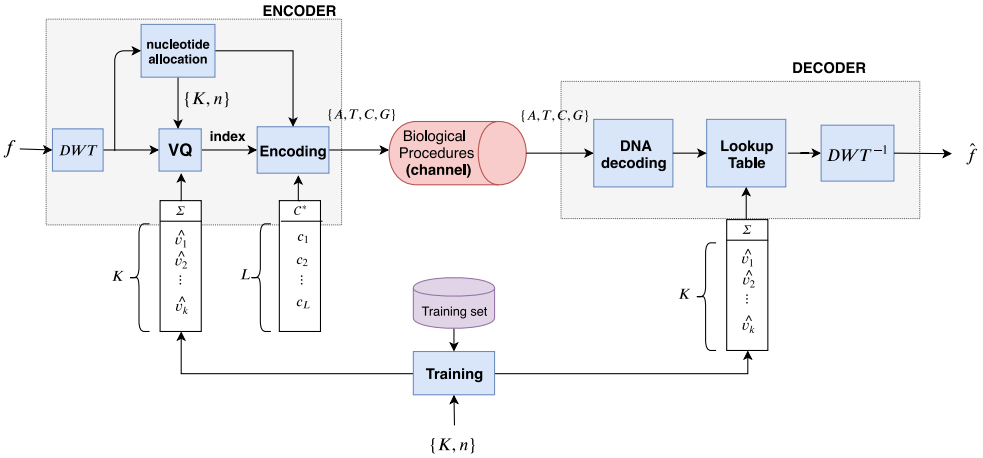


Figure 5.3 – Coding/decoding workflow using Vector Quantization.

kodim15.png
Rate = 4.6 bits/nt (1.74 nt/pixel)
PSNR = 39.4 dB

kodim23.png
Rate = 4.6 bits/nt (1.74 nt/pixel)
PSNR = 41 dB

Figure 5.4 – Encoded/decoded images using the VQ-based solution.

nts length. Figure 5.5(General format) depicts a schema of the general format that all the oligos follow. All the oligos comprise primers (needed for sequencing), sense (S), parity check (P) and an identifier (ID) that specifies the file that the oligo is storing —the image in our case— and that will be the same for all the oligos storing the same image. Payload refers to the stored content and it contains all the information necessary for the decoding as well as the data itself. Depending on the structure of the payload, the oligo can be classified into four different types (see figure 5.5):

1. **Global Oligo (GO)**. It contains general information about the stored image:
   — Header specifying that the oligo corresponds to a GO (G)
   — Number of rows/columns in the stored image (Img rows/Img columns)
   — Levels of wavelet decomposition (lvls of DWT)
   — Padding with random sequence of nucleotides to reach the required oligo length (Rnd).
2. **Subband Oligo (SO)**. They contain information related to the quantization parameters and encoding of the different DWT subbands. There will be as many SIO as wavelet subbands and they store the following information:

— Header specifying oligo type, as well as the wavelet subband level and type (S + DWT lvl & type)
— Type of vectors used for the quantization (row, column or squared blocks) (Vect type)
— Length of the quantization vectors (in case of being blocks, the length refers to both dimensions) (Vect len)
— Size of the quantization codebook (K)
— Size of the quaternary dictionary (L)
— Length of the codewords (l)
— Type of mapping: simple, double or triple (mapping)
— Padding with random sequence of nucleotides to reach the required oligo length (Rnd).

3. **Mapping Oligo (MO)**. They store the information related to the mapping of the quantization indices to the DNA codewords as a result of applying the algorithm presented in section 3.2 to robustify the code against errors:
— Header to specify the oligo type and subband (M + DWT lvl & type)
— Offset that indicates the sorting of the MO to allow the de-formatting of the oligos and reconstruction of the encoded data (offset)
— The mapping information itself
4. **Data Oligo (DO)**. They contain the data itslef (i.e. stored image). They follow the same structure as MO.

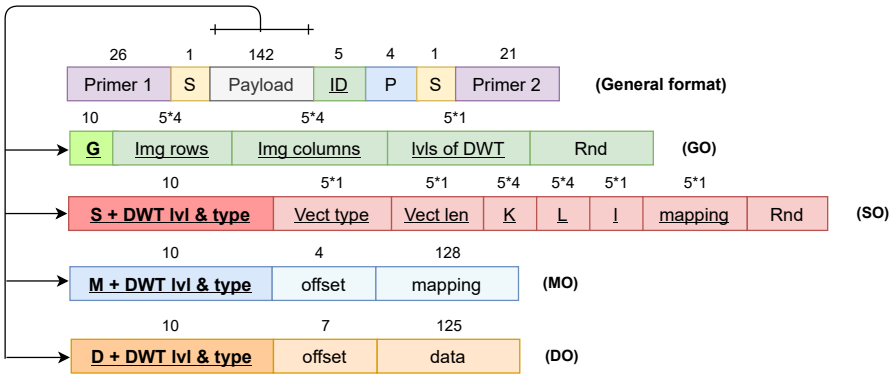


Figure 5.5 – Formatting for VQ-based encoding. Underlined fields are barcoded. The values on top of each field indicate the length.

Barcoding

Since the high throughput DNA sequencing is a procedure which introduces noise in the oligos, the full retrieval of the stored information can be at stake. In fact, if an error occurs in some important headers, the decoding becomes challenging if not impossible. In this experiment all the critical information was robustified by encoding them using error correcting DNA barcodes constructed with Paircode (see section 3.2.2). A set of barcodes $\mathcal{B}$ includes all those codewords among which the Levenshtein distance is high enough to allow correction in case that errors of any type (insertion, deletion or substitution) appear. Interesting studies on DNA barcodes can be found in (Hawkins, Jones, Finkelstein, & Press, 2018), (Buschmann & Bystrykh, 2013) and (Ashlock & Houghten, 2009). To better understand the purpose of barcodes, we will analyse a simple example. Lets suppose the case depicted in figure 5.6, where some information encoded by one of the

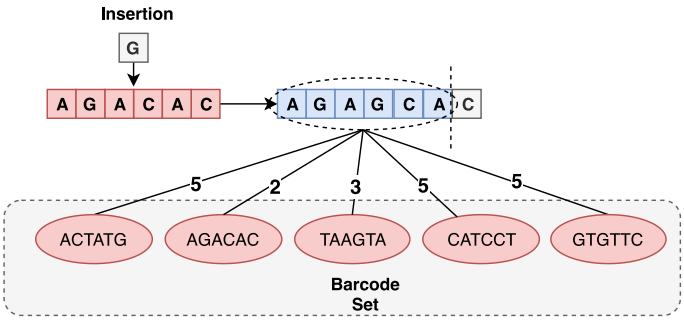


Figure 5.6 – Barcode Example.

5 possible codewords in the barcode set is corrupted by an insertion between the 3^{rd} and the 4^{rth} nucleotide. This will shift the last 3 nucleotides of the codeword pushing the last nucleotide out of the codeword frame. The produced codeword does not exist in the barcode set. This happens thanks to the high distance among the codewords of the barcode set. Thus if this codeword is received in the decoding it is clear that some error has occurred. It is important to mention the fact that the barcode generation algorithm is implicit in the decoding process. To correct the error one can compute the Levenshtein distance between the received codeword and all the codewords in the barcode set, correcting it to the closest codeword existing in the barcode set. For our work, this barcoding method can be used for robustifying headers containing information about the DWT such as the subband type and subband level, information about the VQ such as the number of vectors K and the length of vectors ℓ that was used for each subband, as well as information for the offsets for each chunk of data in each oligo. The construction of all the possible barcode sets using constrained codebooks created by Paircode is achieved by the following procedure:

— Initialization: Add the first codeword c_1 of our codebook $\mathcal{C}^*$ (the Paircode dictionary) in a first codeword set $\mathcal{B}_1$, set $S = 1$.

— For each next codeword c_k for all $k = 1, 2, \ldots, K$: Check the distance between all the codewords in each of the existing barcode sets $\mathcal{B}_i$ with $i = 1, 2, \ldots, S$. If the distances between all codewords in a barcode set $\mathcal{B}_i$ is bigger than $d_{min} = 2 * \mu + 1$, with $\mu_{\mathcal{B}}$ being the maximum tolerance to errors of barcode set $\mathcal{B}$, then this codeword is added to the barcode set $\mathcal{B}_i$. If the previous condition wasn't met for any existing barcode set, create a new barcode set $\mathcal{B}_{S+1}$ containing this codeword.

Once all the possible barcode sets have been created, we select the set $\mathcal{B}_i$ with the biggest cardinality. In this experiment we considered the barcode sets presented in table 5.1.

All the barcoded fields appear underlined in figure 5.5. We used two different barcode sets according to the needs of the field to be encoded. For those fields represented in bold we used a set of barcodes of 10 nts length with resistance to 2 errors (E). The rest were encoded using barcodes of 5 nts length with resistance to 1 error (B). Additionally, trying to further protect the low frequency wavelet subband —which contain the most relevant information for the decoding— we encoded the data values with barcodes of 9 nts length with resistance to 1 error. These correspond to barcode sets E, B and D from table 5.1 respectively. Table 5.2 shows the total amount of oligos of each type generated after formatting.

Table 5.1 – Size and error tolerance of the barcode sets used in the wet-lab experiment.

	Codeword length (nts)	Tolerance (# errors)	Size (# codewords)
A	4	1	4
B	5	1	10
C	7	2	4
D	9	1	426
E	10	2	43

Table 5.2 – Number of oligos of each type obtained after formatting using the VQ-based solution.

Oligo Type	kodim15	kodim23
GO	1	1
SO	10	10
MO	124	241
DO	5436	5608
Total	5571	5860

5.2.2 JPEG-inspired DNA codec

The algorithm

In (Dimopoulou et al., 2021a) we proposed a variable-length encoding solution inspired by the main workflow of the classical JPEG standard. The proposed encoding follows the same workflow of classical JPEG but replacing the two binary encodings (Huffman and binary coding) by quaternary ones which respect the encoding constraints of DNA coding: Goldman encoding (Goldman et al., 2013) and Paircode (Dimopoulou, Antonini, et al., 2019). Goldman encodes the categories into a stream of the trits 0, 1 and 2 using a ternary Huffman. Then, each trit is replaced with one of the nucleotides, excluding the one which was previously used, this way ensures that no homopolymers are generated. Paircode is used as a fixed length coder for the indexes instead of binary coding. For a more detailed explanation readers can refer to section 3.2.1. The workflow of the JPEG-inspired DNA codec is depicted in figure 3.1. It is important to mention that because the JPEG-inspired algorithm for DNA uses a variable-length Hufmman-based coder, it is necessary to also transmit the frequency tables of the categories, along with the rest of the data, to the decoder. More specifically, they are encoded with a predefined fixed-length coder using Paircode.

The main advantage of this coding schema compared to the fixed length VQ-based solution is that it provides higher compression at a given distortion. For that reason, we chose this encoding to store a color image, more specifically the image "kodak23" (figure 5.1 right).

Starting with the RGB data, the image was first decomposed into the luminance and chroma components (YUV). As human perception is more sensitive to the intensity than to the color, the color information —stored in the UV components— can be sub-sampled without a significant loss. More concretely, we applied the 4:2:2 YCbCr subsampling scheme * by which the two chroma components are sampled at half the horizontal sample rate of luma (Y). As a result, the horizontal

*. https://en.wikipedia.org/wiki/Chroma_subsampling#4:2:2

chroma resolution is halved. The luminance component was encoded at a rate of 10.26 bits/nt (0.78 nt/pixel) with a PSNR of 41.5 dB, obtaining quality comparable to the case of the same image encoded using the fixed-length solution. Figure 5.7 depicts how the image was processed prior to the encoding of each of its components independently. Figure 5.8 shows the encoded/decoded color image as well as its YUV components.

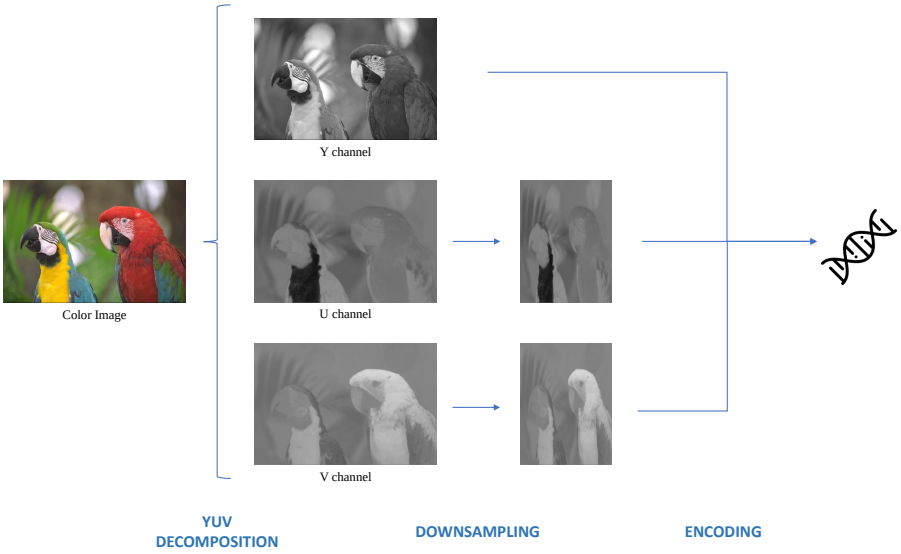
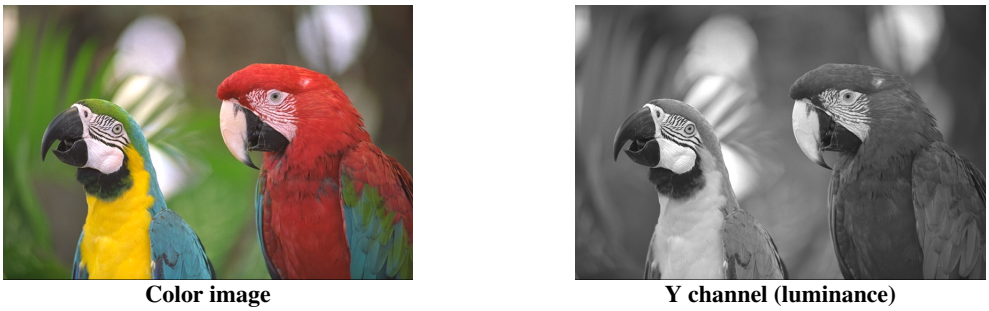


Figure 5.7 – Decomposition of the color image into the YUV components before encoding.

Formatting

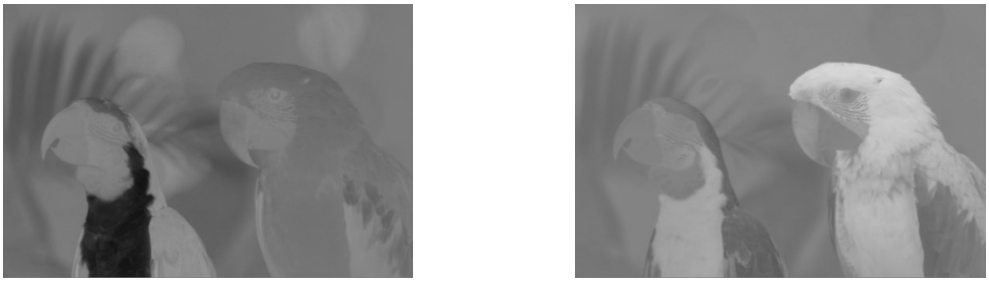
Once encoded, the sequences were formatted into short oligos of 200 nts length. The oligos follow the same general format explained earlier for the fixed-length solution. We defined three types of oligos in this case:

1. **Global Oligo (GO)**. It contains general information about the stored image:
   — The YUV component from which the oligo stores information (channel)
   — Header specifying that the oligo corresponds to a GO (G)
   — Block size of the DCT (block size)
   — Number of rows/columns in the stored image (Img size)
   — Maximum category of DC indices available for this encoding (DC max cat)
   — Maximum index of run/category for the AC indices of the encoding (AC max cat)
   — Maximum offset of data oligos to determine the end of the encoded data (max offset)
   — Length of the quaternary codewords used to encode the DC and AC frequencies (DC cw len/AC cw len)
   — Dynamic of the encoded image (α int/α float)
   — Padding with random sequence of nucleotides to reach the required oligo length (Rnd).
2. **Frequency Oligo (FO)**. They contain the frequencies of 3-ary Huffman used for the encoding of AC and DC indices. They store the following information:
   — The YUV component from which the oligo stores information (channel)
   — Header specifying that the oligo corresponds to a FO (F)
   — Flag indicating if the oligo contains AC or DC frequencies (AC/DC)



Color image
Rate = 20.51 bits/nt (0.39 nt/pixel)
PSNR = 37.3 dB

Y channel (luminance)
Rate = 10.26 bits/nt (0.78 nt/pixel)
PSNR = 41.5 dB



Up-sampled U channel
Rate = 41.32 bits/nt (0.19 nt/pixel)
PSNR = 45.06 dB

Up-sampled V channel
Rate = 40.7 bits/nt (0.2 nt/pixel)
PSNR = 44.17 dB

Figure 5.8 – Encoded/decoded images using the JPEG-inspired DNA coder.

— Offset that indicates the sorting of the FO to allow the de-formatting of the oligos and reconstruction of the encoded data
— The encoded frequencies

3. **Data Oligo (DO)**. They encode the data itself (AC and DC values):
— The YUV component from which the oligo stores information (channel)
— Header specifying that the oligo corresponds to a DO (D)
— Offset that indicates the sorting of the DO to allow the de-formatting (offset)
— The encoded values (data)

Figure 5.9 shows a schema of the oligo formatting just described.

Barcoding

All the important fields were also barcoded to further protect against sequencing noise and ensure the decodability of the stored data. In this case we used a barcode set with length 7 nts resistant to 2 errors (C) to encode the channel and the oligo type. The AC/DC flag was encoded with barcodes of 4 nts length resistant to 1 error (A). The rest of the barcoded fields were protected using barcodes of length 5 nts resistant to 1 error (B). These correspond to barcode sets C, A and B from table 5.1 respectively. Table 5.3 shows the total amount of oligos of each type generated after formatting.

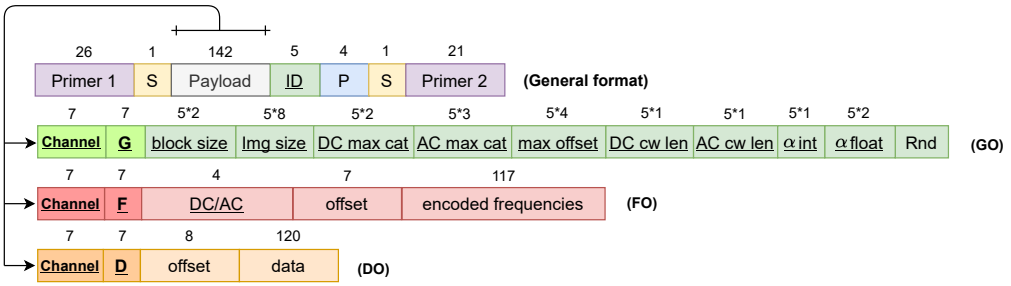


Figure 5.9 – Formatting for JPEG-inspired encoding. Underlined fields are barcoded. The values on top of each field indicate the length.

Table 5.3 – Number of oligos of each type obtained after formatting using the variable-length JPEG-inspired solution.

Oligo Type	Y	U	V	Color
GO	1	1	1	3
FO	14	13	13	40
DO	2556	635	645	3836
Total	2571	649	659	3879

5.2.3 Transcoding of JPEG-compressed data

The algorithm

During the past years, most of the works on DNA data storage have been based on directly transcoding binary sequences into DNA (Goldman et al., 2013 ; Blawat et al., 2016 ; Erlich & Zielinski, 2016). These encodings mainly differ from each other on the quaternary dictionary used to translate the binary data into a DNA-like sequence. Hence, in this experiment we compare the two encodings proposed by the team —VQ-based encoding and JPEG-inspired DNA codec— with a simpler transcoding solution. This method encodes the binary output of a classical JPEG using a quaternary fixed length coding for encoding each byte of the JPEG binary stream. More specifically, each byte is translated into a quaternary codeword of 5 nts length constructed with Paircode. As mentioned in section 5.2.2, in the case of the JPEG-inspired DNA coder, the decoder requires the transmission of the frequency tables that have been previously encoded with a predefined fixed-length coder. To have a fair and common ground for comparison, in the case of transcoding we adapted the JPEG algorithm to use the frequencies computed from the binary source. Hence, these frequencies must also be encoded and transmitted to the decoder.

Although this solution proved to give the best compression results (Dimopoulou et al., 2021a), its main disadvantage is the high impact of errors. With the proposed transcoding method, a single error in the quaternary strand can cause the wrong decoding of the 8 bits stored by the erroneous codeword. Even in the case of a substitution error, which are the easiest to deal with, the wrong decoding of a single byte of the stored data can propagate to the following values due to the fact that JPEG compression follows a variable-length schema. Figure 5.10 shows the workflow of the transcoding solution.

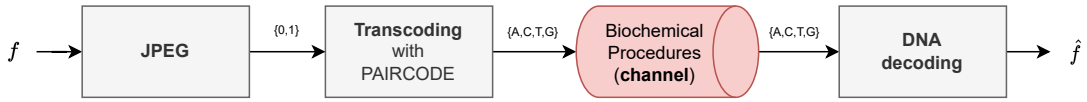


Figure 5.10 – Transcoding of the binary JPEG-compressed data using Paircode.

Following workflow above described, we compressed and encoded the image "kodim23" in grayscale at a rate of 11.25 bits/nt (0.71 nt/pixel) and a PSNR of 41.5 dB.

Formatting

Once encoded, the sequences were formatted into short oligos of 200 nts length. The oligos follow the same general format than the other two encodings. We defined three types of oligos:

1. **Global Oligo (GO)**. It contains general information about the stored image:
   — Header specifying that the oligo corresponds to a GO (G)
   — Block size of the DCT (block size)
   — Number of rows/columns in the stored image (Img size)
   — Maximum category of DC indices available for this encoding (DC max cat)
   — Maximum index of run/category for the AC indices of the encoding (AC max cat)
   — Length of the quaternary codewords used to encode the DC and AC frequencies (DC cw len/AC cw len)
   — Number of bits encoded by the last DNA codewords in the case that is less than 8 (Rem digits)
   — Dynamic of the encoded image (α int/α float)
   — Padding with random sequence of nucleotides to reach the required oligo length (Rnd).
2. **Frequency Oligo (FO)**. They contain the frequencies used for the encoding of AC and DC indices. They store the following information:
   — Header specifying that the oligo corresponds to a FO (F)
   — Flag indicating if the oligo contains AC or DC frequencies (AC/DC)
   — Offset that indicates the sorting of the FO to allow the de-formatting of the oligos and reconstruction of the encoded data
   — The encoded frequencies
3. **Data Oligo (DO)**. They encode the data istelf (AC and DC values):
   — Header specifying that the oligo corresponds to a DO (D)
   — Offset that indicates the sorting of the DO to allow the de-formatting (offset)
   — The encoded values (data)

Figure 5.12 shows a schema of the oligo formatting just described.

Barcoding

All the important fields were also barcoded to further protect against sequencing noise and ensure the decodability of the stored data. In this case we used a barcode set with length 7 nts resistant to 2 errors (C) to encode the oligo type. The AC/DC flag was encoded with barcodes of 4 nts length resistant to 1 error (A). The rest of the barcoded fields were protected using barcodes of length 5 nts resistant to 1 error (B). These correspond to barcode sets C, A and B from table 5.1 respectively. Table 5.4 shows the total amount of oligos of each type generated after formatting.

kodim23.png
Rate = 11.25 bits/nt (0.71 nt/pixel)
PSNR = 41.5 dB

Figure 5.11 – Compressed and transcoded image.

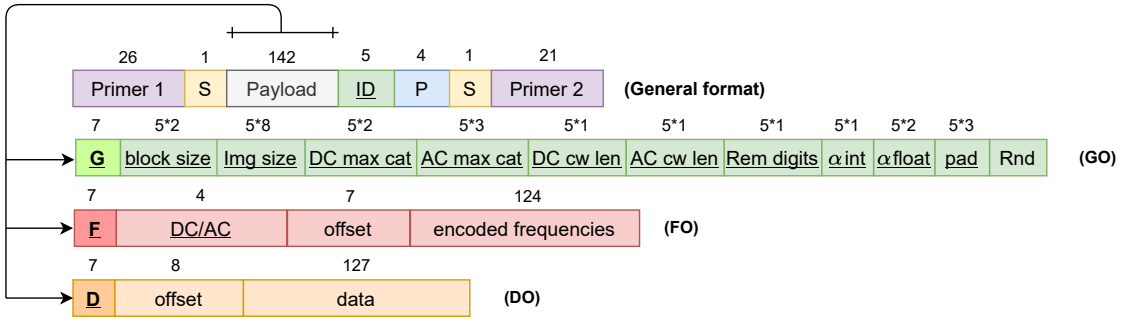


Figure 5.12 – Formatting for transcoding. Underlined fields are barcoded. The values on top of each field indicate the length.

Table 5.4 – Number of oligos of each type obtained after formatting with JPEG cmpression and transcoding.

Oligo Type	kodim23
GO	1
FO	13
DO	2201
Total	2215

5.2.4 Test sequences

Additionally, we created an extra oligo set including pattern repetition and homopolymers with the goal of analysing how the problematic sequences identified in the state of the art affect synthetic nanopore-sequenced DNA. More concretely, we created oligos with:

— **Homopolymers** for each of the four bases (A, C, T and G) and lengths ranging from 5 to 10 nts.

— **Patterns** using the codewords constructed with Paircode (our quaternary constrained dictionary) with lengths ranging from 4 to 10 nts.

The results of this experiment are presented in section 5.8.

5.3 DNA synthesis and storage

After encoding and formatting, a total amount of 18000 oligos were generated, including the four encoded images and the test sequences. The synthesis of the oligos was done by Twist Bioscience †, a well-known company based in San Francisco. Oligos were synthesized with phosphoramidite chemistry (described in section 2.1.1.1). The manufacturing process developed by Twist Bioscience allows the miniaturization of the synthesis chemistry, which compared to traditional synthesis methods reduces the reaction volumes by a factor if 1 million and increases the throughput by a factor of 1000. The miniaturization of chemical reactions presents other advantages such as the use of smaller volumes of reagents, which ultimately reduces the cost as well as the environmental impact as it generates less product waste. The end product delivered by the company is a pool of single-stranded oligos (i.e. the desired sequences) provided in its dried form.

Although dried DNA ensures stability for 3 months (at room temperature) to up to a few years (if stored at -80ºC), this project aims for the long-term storage of digital images into DNA. Therefore, in order to ensure the stability of the DNA molecules in the longer term, the pool of oligos was encapsulated inside an airtight stainless steel minicapsule called DNAshell ‡, by the company Imagene §, located in Evry (France). As explained in section 2.1.3, the DNAshell protects DNA from alteration factors such as water, oxygen or light by preserving the dried DNA molecules under an inert atmosphere. Several studies have demonstrated that the encapsulation in DNAshells allows to preserve safely and cost-effectively high quality DNA in the long-term at room temperature (Washetine et al., 2019 ; Clermont et al., 2014 ; Liu et al., 2015).

5.4 DNA sequencing

The sequencing of the oligos was carried out by the Institute Pharmacology Moléculaire Et Cellulaire (IPMCC - University Côte d'Azur and CNRS) located in Sophia Antipolis (France), also part of the OligoArchive consortium. Initially, three different sequencing runs were performed: one with Illumina NextSeq and two with MinION. More concretely, the second sequencing run with MinION followed a different protocol called R2C2.

Rolling Circle Amplification to Concatemeric Consensus (R2C2) is a method proposed in (Volden et al., 2018) to improve the accuracy of nanopore-sequenced reads. The main idea behind this protocol is to apply the circular consensus principle used in PacBio sequencers ¶. The first step is to circularise the DNA molecules using a DNA splint. The splint binds the target sequences thanks to some overlapping regions. In our case, all the DNA strands contain the Illumina primers in both ends, thus, these known extremities can be used to bind the splints via Gibson assembly ‖. Once the DNA is circularised, it is amplified by Rolling Circle Amplification (RCA), described in figure 5.13. The resulting raw reads are split into subreads containing full-length or partial sequences, which are combined into an more accurate consensus sequence using C3POa ** (Con-

†. https://www.twistbioscience.com/
‡. http://www.imagene.eu/dnashell-rnashell-en/dnashell-en/
§. http://www.imagene.eu/
¶. https://www.pacb.com/technology/hifi-sequencing/how-it-works/
‖. https://international.neb.com/applications/cloning-and-synthetic-biology/dna-assembly-and-cloning/gibson-assembly
**. https://github.com/rvolden/C3POa

catemeric Consensus Caller using partial order alignments) to generate consensus reads from the raw reads. Figure 5.14 depicts the workflow of the R2C2 protocol.

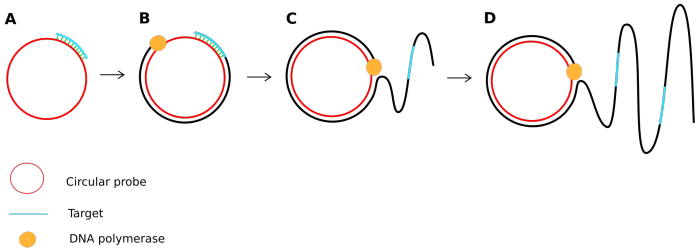


Figure 5.13 – Schematic outline of Rolling Circle Amplification (RCA). (A) A primer complementary to a region of a circular probe anneals to the circular template. (B) DNA polymerase initiates the DNA synthesis. (C) Strand displacement allows the continuation of DNA synthesis along the circular template. (D) DNA synthesis continues to generate a long DNA product. Source (Lau & Botella, 2017)

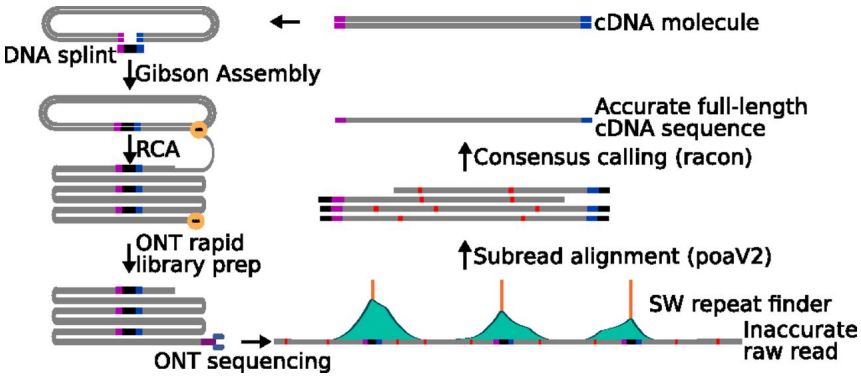


Figure 5.14 – R2C2 method overview. Source (Volden et al., 2018)

After one year approximately, ONT released a new chemistry, only available for PromethION so far, which has proven to reach the best read accuracy up to the date for nanopore sequencing. Thus, a fourth run of nanopore sequencing was performed using PromethION instead of MinION.

As explained in section 2.1.4.2, the main difference between MinION and PromethION lies on the throughput, with PromethION having a higher number of nanopore channels. In this experiment, the differences on the results obtained with MinION and PromethION are due to the flowcell used (see table 5.5) and not the sequencing device itself. However, for simplification purposes, we will refer to each sequencing run using the name of the device that was used. For further information, table 5.5 contains the technical and chemical specifications for each sequencing run, including the Flowcell type, sequencing kit and basecaller used in the experiment.

It is important to note that prior to sequencing, PCR amplification is required (see section 2.1.2). The starting amount of DNA and the number of PCR cycles —which directly affect the

amplification rate— are adjusted according to the starting concentration of the sample. After amplification, the amount of DNA loaded on the sequencer flow cell is the one required by the library preparation protocol specific for each sequencing platform. Table 5.6 contains the information regarding the PCR amplification step for each experimental setup.

Table 5.5 – Specifications of the sequencing platforms

	Illumina NextSeq 500	MinION-R2C2	MinION	PromethION
Flowcell type	Mid 150 (read 1: 100, read 2: 68)	FLO-MIN106 (R9.4.1)	FLO-MIN106 (R9.4.1)	FLO-PRO114M (R10.4.1)
Chemistry	Kit Nextflex Small RNAseq V3	SQK-LSK110	SQK-LSK110	SQK-LSK114
Basecaller	bcl2fastq2	Guppy 4.2.2	Guppy 5.0.11	Guppy 6.2.11

Table 5.6 – DNA concentrations and PCR cycles.

	Illumina NextSeq 500	MinION-R2C2	MinION	PromethION
Starting amount of DNA	10 ng	10 ng	10 ng	10 ng
PCR cycles	10	20	10	15
Moles of solute	10 fmoles	96 fmoles	360 fmoles	26 fmoles
DNA loaded on flowcell	1.75 ng	160 ng	54 ng	5 ng

The sequencing throughput highly varies among the different sequencing platforms. Table 5.7 contains the average number of reads per reference obtained from the different sequencers for each encoded image, reaching the highest values for the case of Illumina, which performs sequencing by synthesis as explained in section 2.1.1.1. The difference on the throughput between PromethION and MinION is due to the fact that PromethION contains 2671 nanopore channels for sequencing while MinION contains just 512. The lowest values appear for MinION-R2C2. As explained earlier, R2C2 protocol sequences each circularized DNA molecule several times to compute a consensus read after and obtain higher quality reads, which explains the abnormal low coverage.

Coverage results are consistent among the different encoded images for all the sequencers. However, the behaviour of the test sequences is opposite for Illumina and nanopore devices. In the case of nanopore sequencers, the coverage for the test oligos is significantly higher than for the rest of the oligos. One likely option is that the yield of synthesis is better for these less complex sequences, resulting in a higher concentration of these molecules in the synthesized pool. On the other hand, when using Illumina the coverage halves in the case of the problematic strands. This could be explained by the high error rates induced on these sequences during the reading process, making impossible to align them to their reference and, hence, lost in the process.

Another important aspect to consider is the frequency distribution of the reads. PCR amplification does not ensure an homogeneous replication of the reads, resulting on significant differences on the frequency of the reads. The extreme cases correspond to those oligos with a final coverage

Table 5.7 – Average number of reads for each reference

Image	Illumina	MinION-R2C2	MinION	PromethION
VQ-kodim15	9489	60	442	1355
VQ-kodim23	11265	71	538	1645
JPEG-DNA	12368	73	479	1701
Transcoding	10479	52	349	1635
Test sequences	5888	313	1527	3560

lower than 10 (i.e. less than 10 copies). Table 5.8 contains the number of under-represented reference oligos after sequencing, showing the number of sequences with less than 10 reads. Note that these results were obtained by mapping the noisy reads to the reference oligos, therefore not only perfectly sequenced oligos are considered but also the ones who were unequivocally mapped to a reference.

For the specific case of Illumina, which is expected to provide the best accuracy, just 12 reference oligos are under-represented with less than 10 copies. However, if we look into the perfectly sequenced reads, we find that out of the 18,000 oligos, 17698 are found in the reads in their full sequence without errors. Among the remaining 278 oligos, 260 correspond to test sequences with homopolymers and patterns and the rest belong to the VQ-encoded image "kodak15". To further understand the reason why the remaining 18 sequences had a higher error rate than the rest, one can look at their content, depicted in figure 5.15. All of the 18 sequences contain short patterns, most of them of length 2 nts. Although our encoding applies an hybrid mapping in which the most frequent values are mapped to more than one word, there are some extreme cases in which this approach fails to avoid the creation of patterns. These cases appear mostly in the high frequency wavelet subbands, which commonly contain high amount of zero values.

Among the nanopore sequencing runs, the best results are obtained with PromethION, with no under-represented reads, followed by MinION with 23 low-coverage sequences. Worst results correspond to MinION-R2C2 with 297 oligos with less than 10 copies, explained by the fact that this protocol provides reads with higher accuracy at expenses of a lower coverage as previously shown in table 5.7.

```
CTGTGTGTGTGTGTGTGTGTGTGTGTGTGTGTGTGTGTGTGTGTGTGTGTGTGTGTGTGTGTGTGTTTACATCGAAGTGG
CGTGTGTGTGTGTGTGTGTGTGTGTGTGTGTGTGTGTGTGTGTGTGTGTGTGTGTGTGTGTGTGTGTTACAACACAGTGG
CTGTGTGTGTGTGTGTGTGTGTGTGTGTGTGTGTGTGTGTGTGTGTGTGTGTGTGTGTGTGTGTGTTTACAAAAAAGTGG
CGTGTGTGTGTGTGTGTGTGTGTGTGTGTGTGTGTGTGTGTGTGTGTGTGTGTGTGTGTGTGTGTGTTACAACGTAGTGG
CGAGAGAGAGAGAGAGAGAGAGAGAGAGAGAGAGAGAGAGAGAGAGAGAGAGAGAGAGAGAGAGAGTTACATTGCAGTGG
CAGAGAGAGAGAGAGAGAGAGAGAGAGAGAGAGAGAGAGAGAGAGAGAGAGAGAGAGAGAGAGAGATTACACTCTAGTGG
CGAGAGAGAGAGAGAGAGAGAGAGAGAGAGAGAGAGAGAGAGAGAGAGAGAGAGAGAGAGAGAGAGTTACATCTCAGTGG
CAGAGAGAGAGAGAGAGAGAGAGAGAGAGAGAGAGAGAGAGAGAGAGAGAGAGAGAGAGAGAGAGATTACACTAGAGTGG
CGTGTGTGTGTGTGTGTGTGTGTGTGTGTGTGTGTGTGTGTGTGTGTGTGTGTGTGTGTGTGTGTGTTACAAGGTAGTGG
CTGTGTGATATATATGTTAATATATATATATGTTAATATGTTAATATGTTAGTTAATATATATATATGTTTTACAGTCTAGTGG
CCTCGTCATCCATCCATCCATCCATCCATCCATCCATCCATCCATCCATCTCGTCATCCATCCATCTTACATTATAGTGG
CGAGTGAGTGAGTGAGTGAGTGAGTGAGTGAGTGAGTGAGTGAGTGAGTGAGTGAGTGAGTGAGTGTTACACTTCTACTC
CCACACACACACACACACACACACACACACACACACACACACACACACACACACACACACACACACTTACACTTGTACTC
CACACACACACACACACACACACACACACACACACACACACACACACACACACACACACACTAGTATTACAACCTTACTC
CCACACACACACACACACACACACACACACACACACACACACACACACACACACACACACACACACTTACAGAAGTACTC
CACACACACACACACACACACACACACACACACACACACACACACACACACACACACACACACACATTACAATCGTACTC
CCACACACACACACACACACACACACACACACACACACACACACACACACACACACACACACACACTTACACTGTTACTC
CACACACACACACACACACACACACACACACACTAGTTAGTTAGTACACACTCATGAGAACACGATTTACAAGCTTACTC
```

Figure 5.15 – Example of patterns in missing reads after Illumina sequencing.

Table 5.8 – Frequency of underrepresented oligos.

Read Frequency	Illumina	MinION-R2C2	MinION	PromethION
0	4	19	6	0
1	1	15	4	0
2	1	11	1	0
3	0	22	4	0
4	2	24	2	0
5	0	30	0	0
6	0	32	2	0
7	0	46	3	0
8	4	41	0	0
9	0	57	1	0
$\geq$10	17988	17703	17977	18000

5.5 Clustering and consensus

The output of sequencing is a pool containing millions of noisy copies of the original oligos. To be able to infer the initial sequences from them, first they must be grouped into clusters such that each cluster contains all the noisy copies of the same reference oligo. On top of that, these algorithms should not rely on any a priori information other than the expected length of the sequences. However, Levenshtein (or edit) distance has a complexity that is quadratic with the string length, resulting on algorithms with excessive computational usage.

Until very recently, clustering remained a bottleneck on the DNA data storage pipeline. However, with the still increasing interest on DNA data storage, some recent works have proposed very promising solutions.

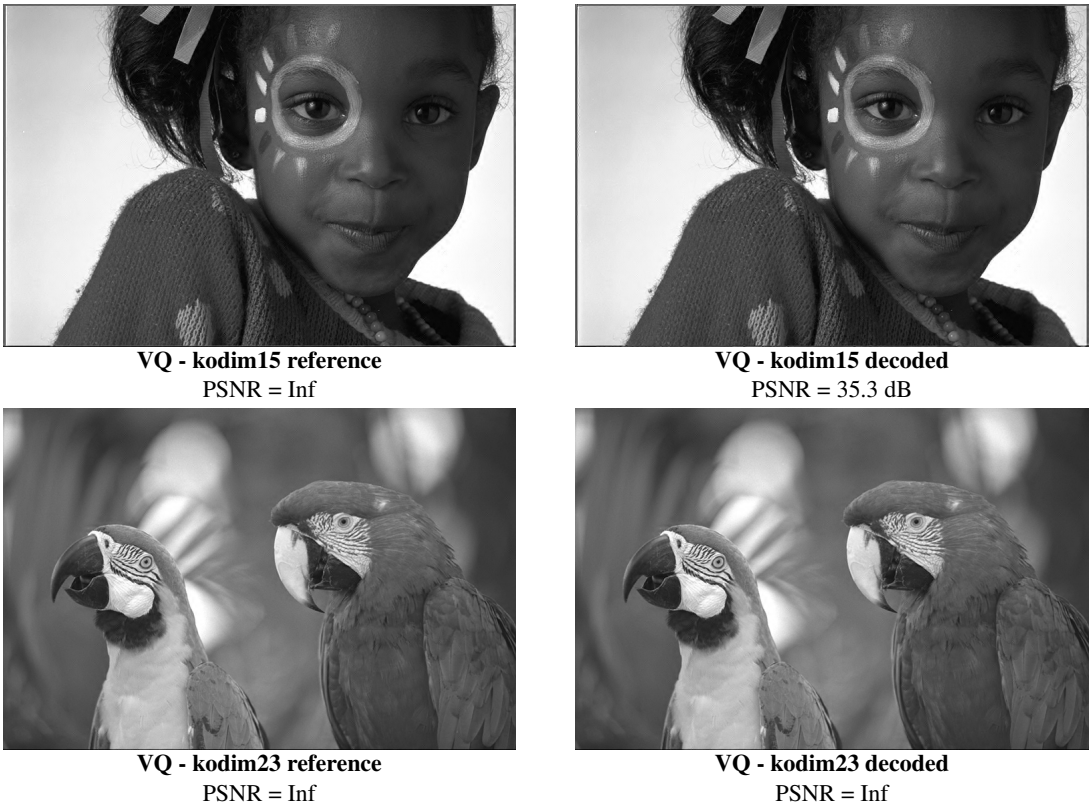
Concretely, in this wet-lab experiment we used OneConsensus (Marinelli et al., 2022) developed by EURECOM, in Sophia Antopolis (France), and Imperial College London, both of them part of the OligoArchive consortium. The algorithm relies on CKG-Embedding and LSH. These tools allow to drastically reduce the computational time and cost, which remains one of the main challenges when developing clustering algorithms for long strings. CKG-Embedding can map problems from an edit space into a Hamming space, significantly reducing the computational cost of the problem. Although the embedding does not produce perfect results, the Hamming distance of the embedded strings will accurately track the edit distance of the original reads. Afterwards, Hamming LSH is used over the embedded reads to group together these sequences into pools that, with a very high probability, contain reads which are similar to each other. To clean the false positives produced by LSH, the reads are sorted by length, moving up to the front the ones with correct length that, with high probability, are less noisy. The reads from the back are sequentially aligned to the ones in the front. This alignment progressively splits the reads into different clusters. A read is accepted into a cluster as long as the edit distance is lower than a threshold. Once the clusters are formed, the duplicated ones and these which are too small are discarded. Finally, a classical base-by-base consensus is inferred form the aligned reads in each cluster assigning to each position the most frequent base.

OneConsensus does not need the exact number of reference oligos as input. Instead, the program predicts the number of oligos from the pool of reads according to the input parameters such

as the maximum edit distance among reads to be allowed in the same cluster. As a result, the output of OneConsensus might provide more than one consensus for the same reference oligo. In such cases, some selection criteria is necessary to keep only one of them for decoding. Considering that most of the reads will have a small distance to the reference, with high probability the cluster containing the copies with low error rate will be of bigger size. On the contrary, due to the random distribution of sequencing errors, noisier reads will have higher distance among each other and will be grouped into several clusters of smaller size. Therefore, in the case of having more than one possible consensus for a specific reference, we select the consensus inferred from the cluster with bigger size.

5.6 Decoding results

5.6.1 Illumina NextSeq 500



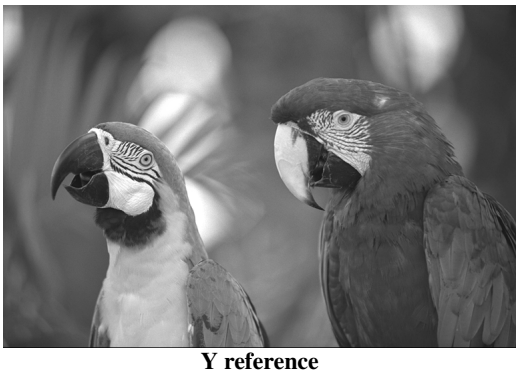
VQ - kodim15 reference
PSNR = Inf

VQ - kodim15 decoded
PSNR = 35.3 dB

VQ - kodim23 reference
PSNR = Inf

VQ - kodim23 decoded
PSNR = Inf

Figure 5.16 – Illumina VQ results

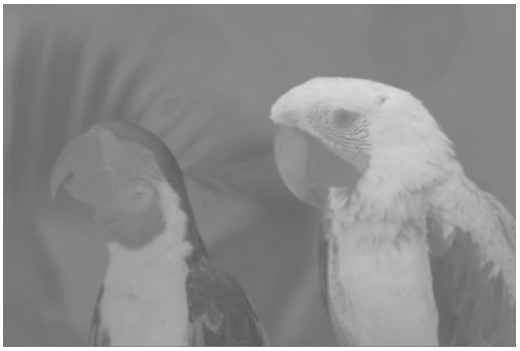


Y reference
PSNR = Inf

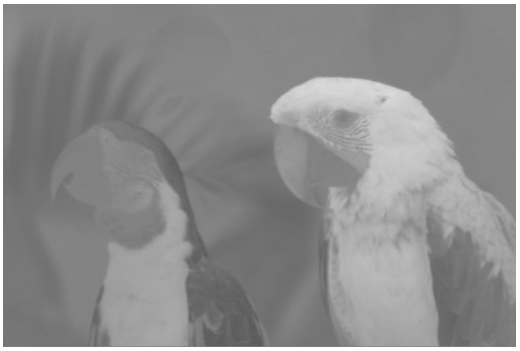
Y decoded
PSNR = 28.23 dB

U reference
PSNR = Inf

U decoded
PSNR = Inf dB



V reference
PSNR = Inf



V decoded
PSNR = Inf dB

RGB reference
PSNR = Inf

RGB decoded
PSNR = 28.34 dB

Figure 5.17 – Illumina JPEG-DNA results

Reference
PSNR = Inf

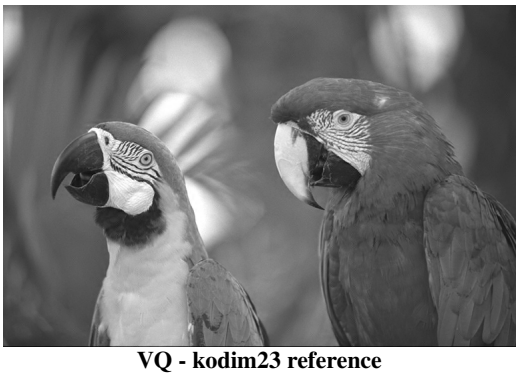
Decoded
PSNR = 59.37 dB

Figure 5.18 – Illumina transcoding results

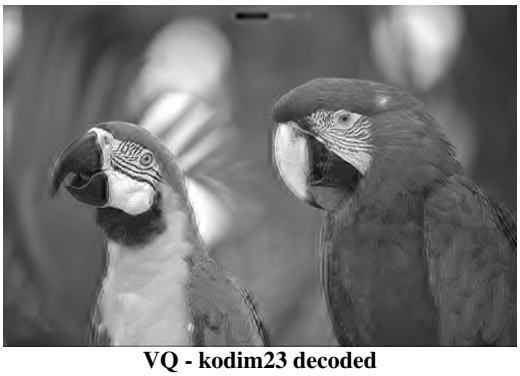
5.6.2 MinION-R2C2

VQ - kodim15 reference
PSNR = Inf

VQ - kodim15 decoded
PSNR = 39.65 dB



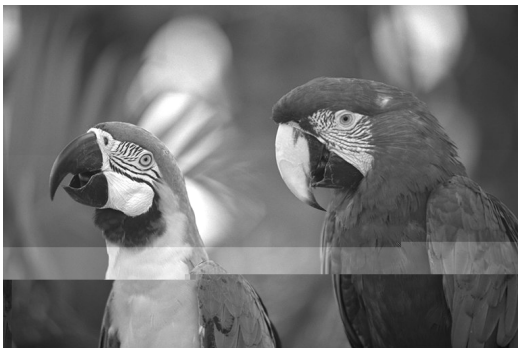
VQ - kodim23 reference
PSNR = Inf



VQ - kodim23 decoded
PSNR = 31.8 dB

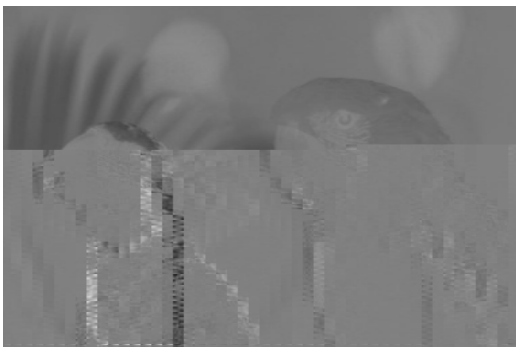
Figure 5.19 – MinION-R2C2 VQ results

Y reference
PSNR = Inf

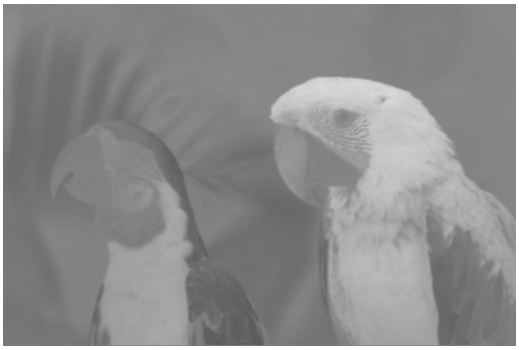


Y decoded
PSNR = 23 dB

U reference
PSNR = Inf



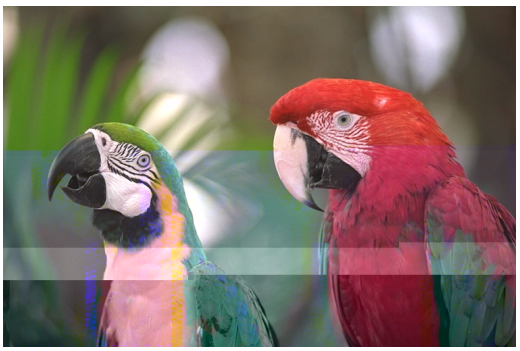
U decoded
PSNR = 19.77 dB



V reference
PSNR = Inf

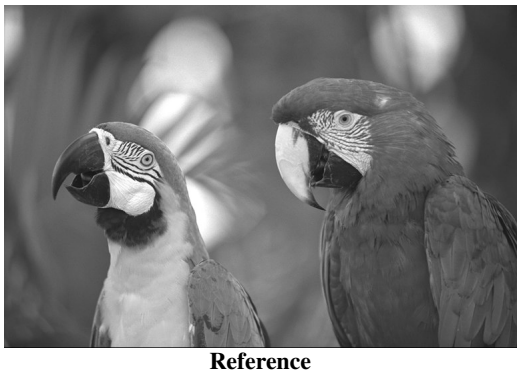
V decoded
PSNR = Inf dB

RGB reference
PSNR = Inf



RGB decoded
PSNR = 18.21 dB

Figure 5.20 – MinION-R2C2 JPEG-DNA results



Reference
PSNR = Inf

Decoded
PSNR = 11.81 dB

Figure 5.21 – MinION-R2C2 transcoding results

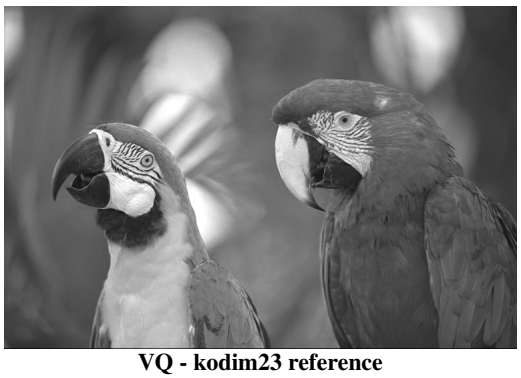
5.6.3 MinION

Non decodable.

5.6.4 PromethION

VQ - kodim15 reference
PSNR = Inf

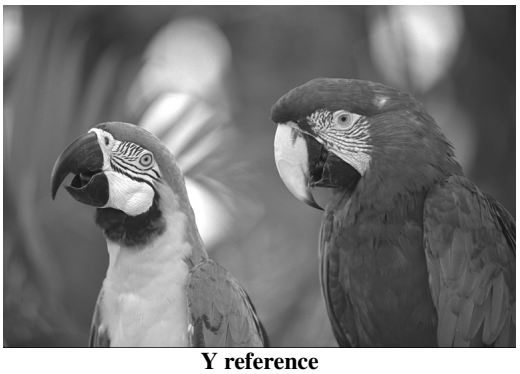
VQ - kodim15 decoded
PSNR = 36.55 dB



VQ - kodim23 reference
PSNR = Inf

VQ - kodim23
PSNR = 44.48 dB

Figure 5.22 – PromethION VQ results

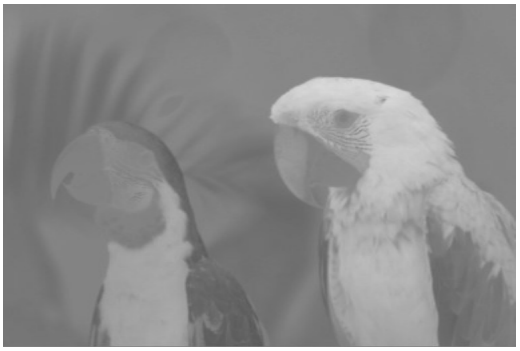


Y reference
PSNR = Inf

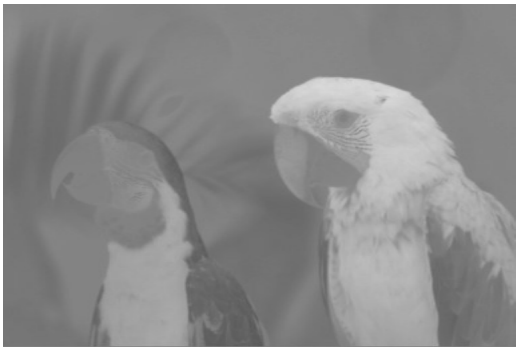
Y decoded
PSNR = Inf

U reference
PSNR = Inf

U decoded
PSNR = Inf



V reference
PSNR = Inf



V decoded
PSNR = Inf

Color reference
PSNR = Inf

RGB decoded
PSNR = Inf

Figure 5.23 – Promethion JPEG-DNA results

Reference
PSNR = Inf

Decoded
PSNR = 34 dB

Figure 5.24 – Promethion transcoding results

5.7 Comparison of nanopore sequencing

One of the goals of this thesis is to include nanopore sequencing into the DNA data storage pipeline, allowing faster and cheaper sequencing. Therefore, for the analysis and comparison of the decoding results of the experiment we focus on the nanopore-sequenced reads (MinION-R2C2, MinION and PromethION).

We computed the base error rates for the three methods including substitutions, insertions and deletions of nucleotides. These values can be found in table 5.9. Interestingly enough, the R2C2 protocol highly improves the quality of the reads, showing an error rate more than five times smaller than ordinary MinION and comparable to the one provided by the last chemistry tested with PromethION. It might also be noted that the ratio of each type of error remains consistent among the three experiments, being around 0.15 for substitutions, 0.48 for insertions and 0.37 for deletions.

Table 5.9 – Average error rates per base with nanopore.

	MinION-R2C2	MinION	PromethION
Error rate	0.0138	0.0702	0.012
Substitution rate	0.0022	0.01	0.0018
Insertion rate	0.0067	0.033	0.00575
Deletion rate	0.005	0.0272	0.00445

To further analyse the nature of the sequencing noise, we show in figure 5.25 the distribution of the edit distance of the reads to their references for the three cases. Note that the edit distance indicates the number of edits (substitutions, insertions and deletions of nucleotides) necessary to transform a read into its reference, which in this experiment is 200 nts long. The distribution of MinION-R2C2 and PromethION are comparable. In both cases, more than 80% of the reads present a small edit distance of 5 or less while in the case of MinION these reads account for just a 11%.

In order to compare the robustness of the three encoding methods tested in this experiment (VQ, JPEG-DNA and transcoding), we present in table 5.10 the average edit distance between the nanopore-sequenced reads and their reference for each of the encoded images. It is interesting to

focus on the case of MinION-R2C2, which shows the lowest average edit distance for transcoding, followed by JPEG-DNA and VQ. However, the highest PSNR among the decoded images was provided by the VQ-based solutions despite having higher average distances, followed by JPEG-DNA and transcoding. As explained at the beginning of this chapter, the selection of the encoding technique cannot be based only on the efficiency of the encoder but also its robustness against errors. It is important to mention that the low visual quality provided by VQ-kodim23 should be considered as a special case. The reason why the decoded image provides a low visual quality was the loss of one of the subband oligos (SO), which store some information crucial for the decoder. If one of these few oligos is lost, the decoding of the whole wavelet subband is jeopardised. Although such type of oligos are further protected against noise, the extremely low coverage provided by MinION-R2C2 caused the complete loss of some oligos, which is critical in the case of SOs. However, if we assume that the parameters stored in the missing SO are known by the decoder, the reconstructed image, depicted in figure 5.26, would have a PSNR of 34.12 dB. Overall, results confirm the importance of robustifying the DNA codes as we risk to suffer from data loss as in the case of transcoding, which despite having the smallest average edit distance to the reference, provided the worst results in terms of quality of the decoded image.

On the other hand, the decoding of PromethION reads provided better results in terms of quality of the decoded image despite having similar average distances to MinION-R2C2. This is due to the higher coverage of this sequencer, as previously seen (table 5.7), which improves the accuracy of the inferred consensus. If one compares the results for JPEG-DNA, both sequencers provided very similar average edit distance. However, the image was completely recovered with PromethION while with MinION-R2C2 the decoded image had a rather poor quality.

Table 5.10 – Average number of errors between raw reads and reference (Avg dis) and quality of the decoded image (PSNR) with nanopore sequencers.

	MinION-R2C2		MinION		PromethION	
	Avg dis	PSNR	Avg dis	PSNR	Avg dis	PSNR
VQ-kodim15	3.63	39.65 dB	14.28	-	2.41	36.55 dB
VQ-kodim23	2.4	31.8 dB	14.35	-	2.29	44.48 dB
JPEG-DNA	2.69	18.21 dB	13.55	-	2.33	Inf
Transcoding	1.64	11.81 dB	13.52	-	2.40	34 dB

5.8 Test sequences

Although the sequencing method selected to read the DNA strands directly affects the quality of the retrieved sequences, there are some coding constraints identified in the state of the art that, if not respected, can dramatically decrease the accuracy of the sequencing process. Two of the main error generating DNA codes identified up to the date are the repetition of short patterns and homopolymer runs. However, as the sequencing technologies evolve, these constraints should be revised and updated according to the limitations of the sequencers. As an example, Illumina allows a maximum homopolymer run length of up to 3 nts, while with the release of ONT nanopore sequencers this constrained could be ease, allowing homopolymers up to 5 nts. Easing the encod-

ing constraints allows the creation of more complete dictionaries, which ultimately reflects on a higher coding efficiency.

As mentioned in 5.2.4, in this experiment we also synthesized a set of oligos containing repetitions of patterns of different lengths, using the dictionaries constructed with Paircode, and homopolymer runs to better understand the limitations of the quite recent ONT nanopore sequencers. Given that the reads sequenced with MinION did not allow the decoding of the stored images due to the high error rate introduced during the sequencing process, this analysis focuses on the data obtained with MinION-R2C2 and PromethION.

5.8.1 Pattern repetition

The oligos to test pattern repetition were constructed by concatenating several times each of the different codewords generated with Paircode. We tested patterns for codeword lengths between 2 and 5 nts. The constructed patterns contained between 15 and 35 repetitions depending on the length of the codeword being tested. Table 5.11 shows the number codewords for each dictionary size that were correctly decoded codewords when repeated several times. Note that these values were obtained by analysing the content of the consensus inferred from the nanopore-sequenced reads and not the raw noisy reads themselves.

Table 5.11 – Number of codewords whose repetition do not create problems during sequencing.

Pattern length	Dictionary size	MinION-R2C2	PromethION
2 nts	10	0	0
3 nts	40	18	22
4 nts	100	63	71
5 nts	400	298	367

Results show that the creation of patterns is critical when the repeated string has a length of two nucleotides. However, for longer strings we found that not all patterns decrease the accuracy of sequencing as we successfully recovered the data in around 50% of the cases for codewords of 3 nts length and up to 90% for codewords of 5 nts.

Having the knowledge regarding which codewords do not affect the quality of the sequencing process when creating patterns offer promising possibilities. For example, one could improve the robustness of the encoded data by mapping the most frequent values to the words which do no create pattern issues to avoid data loss rather than using one-to-many mapping as described in section 5.2.1.

The codewords that allowed to correctly recover the sequenced data are listed in appendix A.

5.8.2 Homopolymer runs

The repetition of the same nucleotide several times in a row constitutes a major challenge in the sequencing process. Nanopore sequencers are able to decipher the content of DNA molecules by measuring changes on the ionic current transmitted through the nanopore. When the electrical signal does not suffer variations due to homopolymer runs, the basecalling process responsible for translating the measured signal into a sequence of nucleotides fails to correctly infer the length of

the homopolymer. This occurs mainly due to the variable speed at which the DNA strands pass through the pore.

The maximum length of homopolymer runs according to the state of the art is of 3 nts for Illumina sequencers and around 5 for nanopore devices. Nevertheless, the improvements on the basecalling process of the second ones currently allow for a higher tolerance of nanopore sequencing to runs of a same nucleotide.

In this experiment we created DNA sequences containing homopolymers for all the different bases (A, C, T and G) and lengths ranging from 4 to 10 nts. As mentioned in the previous section, results were obtained by analysing the consensus sequences and not the raw reads directly. Interestingly enough, PromethION showed a notable tolerance, allowing the correct retrieval of homopolymers with lengths up to 9 to 10 nts.

Table 5.12 – Maximum homopolymer length (in nts).

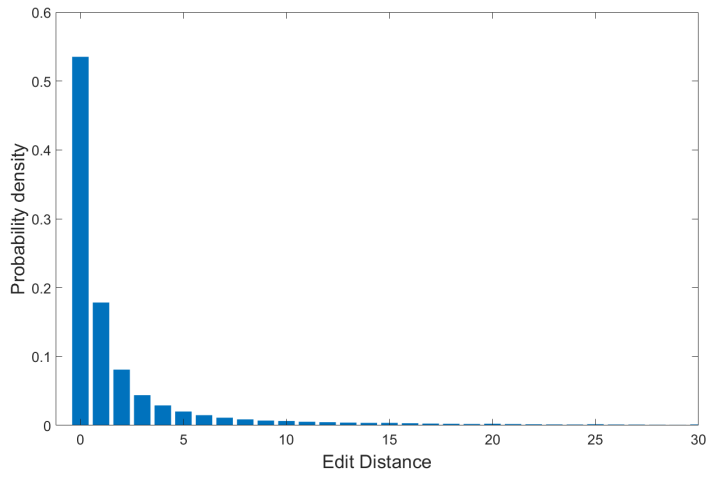
	MinION-R2C2	PromethION
A	6	10
T	9	9
C	7	9
G	6	10

Currently, most of the encoding schemes proposed in the state of the art allow homopolymer runs of 3 nts maximum, threshold that has been widely imposed due to Illumina limitations. However, with the increased interest on nanopore sequencing showed in more recent research, this constraint could be relaxed.

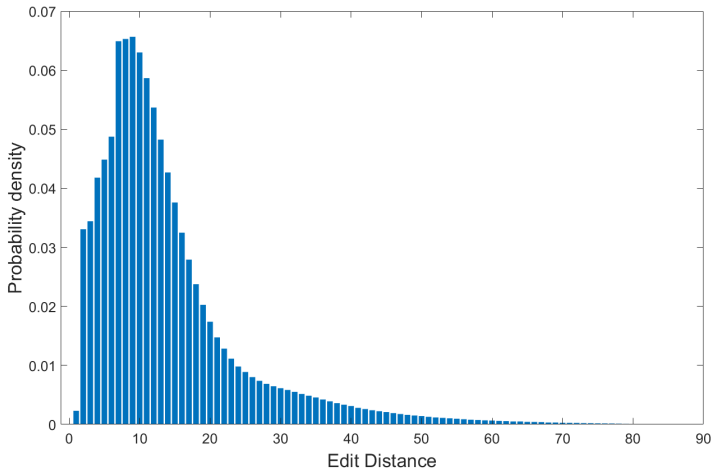
5.9 Conclusions

In this chapter we presented the results of a wet-lab experiment for the storage of digital images into synthetic DNA. We compare the robustness of three coding solutions using different sequencers. Results show that the fixed-length encoded based on vector quantization is the most robust to the remaining errors in the inferred consensus, followed by the variable-length JPEG-inspired codec. The worst results were obtained with simple transcoding, as even single nucleotide errors can affect the whole structure of the stored image. The results provided by the last ONT chemistry with PromethION prove that having enough redundancy to correct the errors in the raw reads is crucial to build an accurate consensus, even for low error rates.

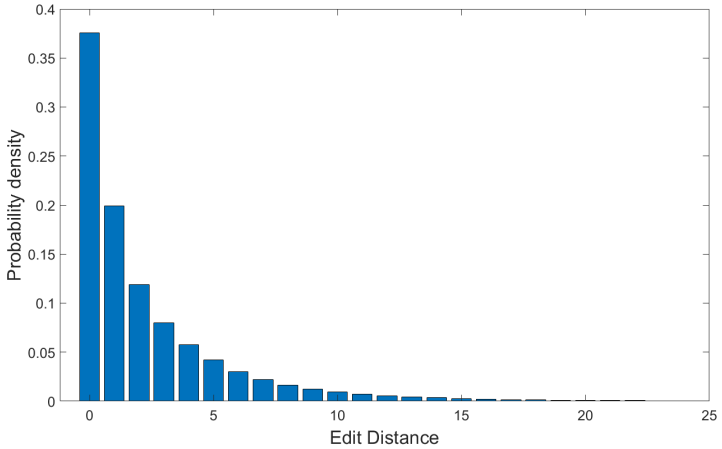
We also tested the accuracy of the different sequencing platforms when the reads contain problematic sequences such as pattern repetition or homopolymer runs. The results provided by nanopore sequencers, specially PromethION (R10.4.1), show that the coding constraints initially imposed by Illumina sequencing can be relaxed, being able to correctly recover sequences with patterns for up to 90% of the tested cases and accurately reading homopolymers of up to 10 nts length. The relaxation of the coding constraints according to the results of the experiments will allow the construction of more complete codebooks with the consequent improvement on the encoding efficiency.



MinION-R2C2



MinION



PromethION

Figure 5.25 – Distribution of the edit distance.

Figure 5.26 – VQ-kodim23 decoded image assuming the information contained in the missing SO is known.

CHAPTER 6

Encoding schema for synthesis by ligation

6.1 Introduction

Although DNA has proven to be an extremely promising candidate to replace current storage media, there are still some major challenges to be addressed before it becomes a reality. One of the most relevant limitations to the development of DNA data storage solutions is the high cost of the biochemical processes involved. It has been estimated that, with the use of current technologies, the cost of storing 1 terabyte of data into DNA would be of around 800 million USD while for LDO tape is 15 USD. Additionally to lowering the cost, the process of DNA writing has to fill a gap of 6 orders of magnitude in speed —from kilobytes to gigabytes per second— in ten years to be comparable to commercial cloud storage systems (Doricchi et al., 2022). The rapid advances on DNA sequencers, specially nanopore sequencing for long reads, have reduced the cost of DNA reading from 40-63 USD per Gigabase (Gb) in Illumina NextSeq 550 platform to around 21 USD per Gb with nanopore platform PromethION, while also offering improved speed with real-time applications.

However, synthesis remains a major limitation, preventing DNA data storage to be scalable for large data archival applications. Currently, the more extended synthesis method is the phosphoramidite chemistry, which builds the DNA strands adding a nucleotide at a time as detailed in section 2.1.1.1. Writing methods based on chemical synthesis have allowed to prove the feasibility of DNA data storage, but they present numerous inconveniences such as the price, speed, limited length of the growing strands and production of hazardous waste. The substantial efforts in this field during the past decades have brought to life novel writing methods. As an example, enzymatic synthesis has born as a potential alternative (see section 2.1.1.2). This method uses natural enzymes —or engineered versions of natural enzymes— to stitch individual nucleotides together into the desired sequence. The process is compatible with aqueous solutions, avoiding the creation of harmful waste byproducts. Although this method is in early stage and still too slow, it promises to become cheaper and faster than phosphoramidite synthesis. Nonetheless, in 2022, the french company DNA Script launched the first DNA printer powered by enzymatic DNA synthesis *.

*. https://www.dnascript.com/products/syntax/

Some start-ups such as HelixWorks †, Catalog ‡ or the DATANA project § from Biosistemica ¶ are working on methods for DNA data storage in which longer DNA molecules are built thanks to the enzymatic assembly of pre-synthesized oligonucleotide libraries.

As described in section 2.1.1.3, DNA synthesis by ligation of motifs is a method for synthesizing DNA molecules by joining together smaller DNA fragments, or "motifs," that have been chemically synthesized in a laboratory. The basic process involves the use of a DNA synthesizer to produce short, single-stranded DNA fragments, which are then purified and mixed together in the presence of an enzyme called DNA ligase. The ligase catalyzes the formation of covalent bonds between the ends of the motifs, effectively "gluing" them together to form a larger, contiguous DNA molecule.

Motifs are generally constructed using a process called solid-phase synthesis, which involves attaching the growing DNA strand to a solid support, such as a bead or a chip, and adding the next nucleotide to the strand in a series of chemical reactions. This synthesis method can be based either on phosphoramidite chemistry or enzymatic synthesis.

The evolution of current synthetic biology techniques as well as the development of new ones bring the need for novel encoding solutions to fill the gap between informatics and chemistry and allow the storage of digital information into DNA. In this chapter, we propose an encoding method adapted to DNA synthesis by ligation of motifs. Due to the infancy of this emerging technology, there are many unsolved questions such as the size of the motif library, the length of the growing strands or the synthesis yield, which makes unfeasible to assess the potential of new encoding solutions. Although no experiments have been carried out so far, this work rather establishes the foundations for a novel algorithm for the encoding of digital images into DNA.

6.2 VQ-based encoding with nucleotide allocation

As previously presented in section 5.2.1, in (Dimopoulou & Antonini, 2021), the Mediacoding group in I3S laboratory proposed a fixed-length encoding solution to store digital images into DNA. While most of the state of the art encoding solutions rely on already existing compression algorithms to provide a compressed representation of the target data and encode the result after, the algorithm presented by Mediacoding group includes compression and applies a source allocation algorithm —or nucleotide allocation— which allows to compute the optimal compression parameters given a user-defined nt rate for an estimation of the best quality/cost trade-off. In other words, it gives control over the compression rate which, ultimately, is equivalent to controlling the synthesis cost. Briefly, the input image is first decomposed into its different subbands using Discrete Wavelet Transform (DWT) and each subband is independently vector quantized. The nucleotide allocation algorithm computes the optimal quantization parameters for each of the subbands so to provide the best possible image quality for a given compression rate, which can be expressed in bits/nt or nt/pixel. After quantization, the indices of the quantization vectors are mapped to a dictionary of DNA codewords obtained with Paircode (see section 3.2.2).

†. https://helix.works/
‡. https://www.catalogdna.com/
§. https://datana-storage.com/
¶. https://biosistemika.com/datana/

6.3 Encoding solution for enzymatic synthesis by ligation of motifs

In this section, we propose an extension of the encoded method presented in (Dimopoulou & Antonini, 2021) to better suit synthesis by ligation of motifs. As previously mentioned, this technique is based on the enzymatic binding of short pre-synthesized oligonucleotides (or motifs). The motifs within a library must respect orthogonality, which ultimately affects the size of the library. Generally, libraries of motifs are pre-defined by the DNA synthesizer and the length of the motifs can vary, typically ranging from 5 to 25 nts. If the aforementioned encoding algorithm was to be used for the encoding into motifs, it could be inefficient to map each motif to a single quantization index, specially in the case of long motifs. An attempt to mitigate the negative impact on the encoding rate would be to adapt the quantization parameters by increasing the length and the number of quantization vectors. However, this approach has its limitations as motif libraries are restricted by two main constraints: the need for respecting the orthogonality of the sequences and technical challenges related to the physical manipulation of the DNA molecules. It is therefore clear that the size of the codebook is limited by the size of the libraries of motifs. On the other hand, increasing the length of the vectors could contribute to worsen the efficiency in terms of bit rate, and especially harmful in the higher level subbands.

The proposed workflow adapted for the encoding of digital image into motifs is depicted in figure 6.1. As in the original algorithm, the input image is first decomposed into its different subbands using Discrete Wavelet Transform (DWT) and compressed with vector quantization (VQ). Afterwards, the quantized wavelet subbands are encoded using motifs. To achieve better rates, the quantized subbands are divided into blocks of quantization vectors according to a "block dictionary", and each of them is assigned to a motif rather than mapping the quantization indices one by one.

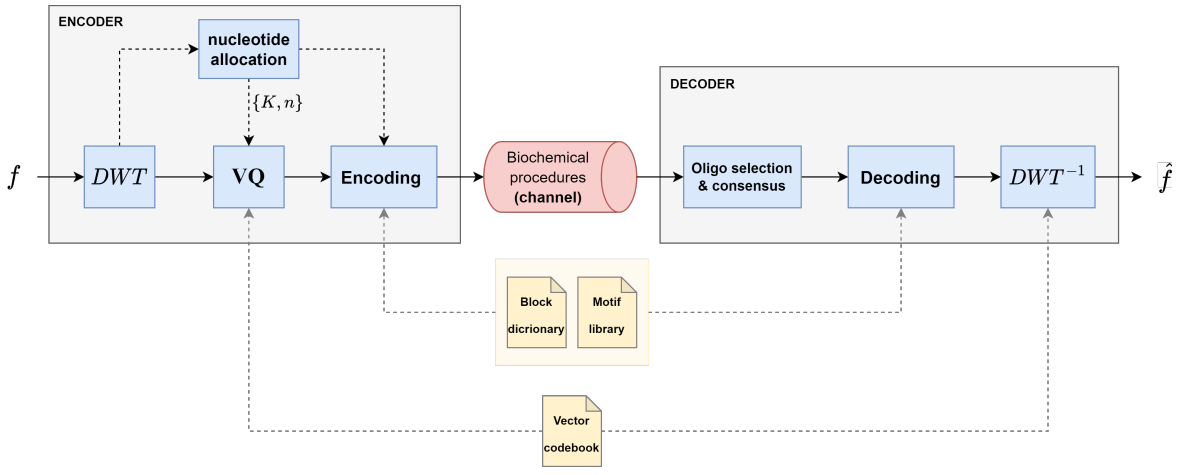


Figure 6.1 – Encoding workflow for DNA synthesis by ligation of motifs. The vector codebook for VQ and the block dictionary are trained prior to encoding and known at the decoder.

To better understand the encoding algorithm presented in this chapter, we will first introduce some notions. We will start by defining the creation of the block dictionary.

6.3.1 Creating the block dictionary

Let $\mathcal{V} = \{v_1, v_2, ..., v_K\}$ be a set of quantization vectors for VQ and $\mathcal{M} = \{m_1, m_2, \ldots, m_P\}$ a set of motifs m_p where $1 \leq p \leq P$. Starting from a training set of T images $\mathcal{I} = \{I^1, I^2, ..., I^T\}$, each image I^t with $1 \leq t \leq T$ is quantized with vector quantization and using the vectors from set $\mathcal{V}$. Let $\mathcal{I}_q = \{I_q^1, I_q^2, ..., I_q^T\}$ be the set of quantized images I_q^t composed by vectors $v \in \mathcal{V}$. We also define three sets B_1, B_2 and B_3, containing blocks composed of vectors $v \in \mathcal{V}$ that we extract from each $I_q^t \in \mathcal{I}_q$. More precisely, for each quantized element $I_{q_{i,j}}^t \in I_q^t$, we generate:

$$b_{1_{i,j}} = \{I_{q_{i,j}}^t, I_{q_{i+1,j}}^t, I_{q_{i,j+1}}^t, I_{q_{i+1,j+1}}^t\}$$

$$b_{2_{i,j}} = \{I_{q_{i,j}}^t, I_{q_{i+1,j}}^t\}$$

$$b_{3_{i,j}} = \{I_{q_{i,j}}^t, I_{q_{i,j+1}}^t\}$$

where the indices i, j denote the position of each quantization element in the image I_q^t. Then, the blocks $b_{1_{i,j}}$, $b_{2_{i,j}}$ and $b_{3_{i,j}}$ will be added to the three sets of blocks as follows (see figure 6.2):

$$B_1 \leftarrow B_1 \cup b_{1_{i,j}}$$

$$B_2 \leftarrow B_2 \cup b_{2_{i,j}}$$

$$B_3 \leftarrow B_3 \cup b_{3_{i,j}}$$

This step will be iterated for all elements i, j in all images $I_q^t \in \mathcal{I}_q$. Ultimately, we will have populated our three sets with blocks of three different shapes extracted from the images.

Considering that each block correspond to a combination of quantization vectors, we can estimate the joint probabilities of the vectors thanks to the sets B_1, B_2 and B_3. The most probable blocks (or combination of vectors) will be stored into a block dictionary $\mathcal{D}_b$ that will be used to encode the images. This dictionary will also contain all the vectors in $\mathcal{V}$. A bijective function $\Gamma_b : \mathcal{D}_b \rightarrow \mathcal{M}$ maps the blocks d_b to motifs m. More precisely, $m_p = \Gamma_b(d_b)$ provides the motif to encode the block d_b. Figure 6.2 depicts a schema of the creation of the block dictionary.

Then a set of motifs will encode the image according to the blocks it contains as described in the following section.

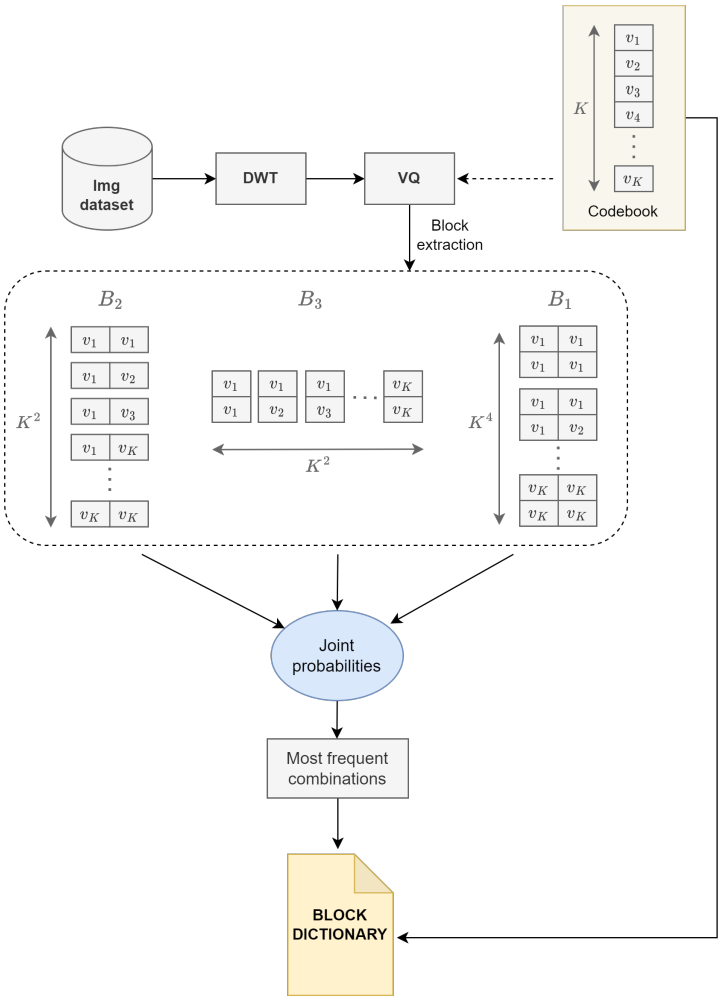


Figure 6.2 – Creation of block dictionary. Starting from a training set of images, each of them is decomposed following Discrete Wavelet Transform (DWT) and quantized. Blocks of vectors are extracted from the quantized subbands and the most frequent combinations added to the block dictionary together with the vector codebook.

6.3.2 Encoding an image with motifs

The problem can be simplified by dividing the image into smaller portions and finding the optimal representation for each of them independently. A simple proposition to address this task could be to start by decomposing the image into tiles of indices of size nxn. For simplification purposes, in this example we will consider $n = 2$.

Starting with an input image I of, the first operation is to first decompose it into subbands I_{sb} of coefficients using the Discrete Wavelet Transform. Each subband I_{sb} is then vector quantized as mentioned earlier. Each quantized subband $I_{q_{sb}}$ of size MxN is then encoded as follows:

1. The quantized subband $I_{q_{sb}}$ is divided into 2x2-sized tiles $X_{i,j}$ where the indices i, j indicate the position of the tile in $I_{q_{sb}}$ with $1 \leq i \leq M/2$ and $1 \leq j \leq N/2$.

   Given the block dictionary $\mathcal{D}_b$, any tile $X_{i,j}$ can be represented in at least one way (using four one-element blocks) and eight at most (depicted in figure 6.3):

— a single block of size 2x2 (6.3a)
— two vertical blocks (6.3b)
— two horizontal blocks (6.3e)
— one vertical block on the left and two one-element blocks on the right (6.3c)
— one vertical block on the right and two one-element blocks on the left (6.3d)
— one horizontal block on top and two one-element blocks at the bottom (6.3f)
— one horizontal block at the bottom and two one-element blocks on top (6.3g)
— four one-element blocks (6.3h)

2. For each $X_{i,j} \in I_{q_{sb}}$ if $X_{i,j} \in \mathcal{D}_b$ then the tile $X_{i,j}$ is encoded using the motif $m_p = \Gamma_b(X_{i,j})$ (case depicted in 6.3a).
3. Otherwise, we divide the tile $X_{i,j}$ into all possible combinations of subsets $X_s \subset X_{i,j}$ where s denotes the number of vectors in the subset (as depicted in figure 6.3b-h). We then proceed to encode the tile into motifs according to the following logic. We define as eligible S_E every combination whose subsets exist in $\mathcal{D}_b$. Among the eligible combinations S_E we select the one containing the least number of subsets and we encode the tile $X_{i,j}$ into as many motifs as subsets in the combination (we obtain $m_p = \Gamma_b(X_s)$ for each subset X_s).

In other words, the encoding into motifs is performed by trying to encode the bigger tiles first and whenever it is not possible because the specific tile $X_{i,j}$ does not exist in the block dictionary $\mathcal{D}_b$, it will be encoded as a combination of smaller ones. As each block (or subset) is assigned to a motif, representing the tile with the least possible number of blocks (i.e. selecting the representation which uses the least number of motifs) will provide the best rate-distortion possible.

Figure 6.4 shows an example of how an image subband is divided into a set of blocks, each of them corresponding to a motif.

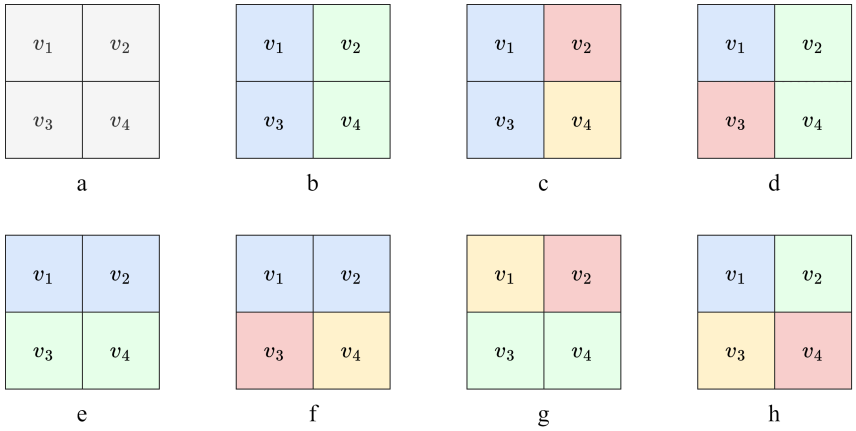


Figure 6.3 – Schema of the possible ways for representing a single tile using blocks. The different colors in each representation indicate the number of motifs needed for its encoding.

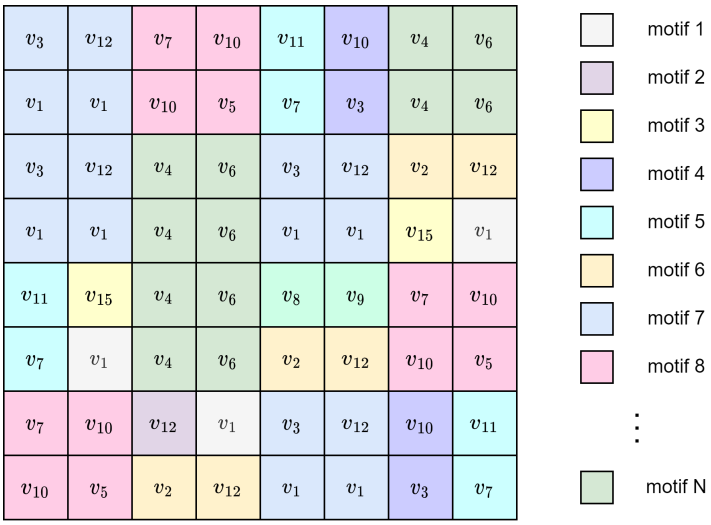


Figure 6.4 – Example of encoding an image into motifs. Initially, the quantized image is divided into tiles of 2x2 elements. According to the block dictionary $\mathcal{D}_b$ previously computed, the encoding into motifs will be performed by trying to encode the whole tiles first. Whenever it is not possible because the tile does not exist in the $\mathcal{D}_b$, the tile will be progressively divided into smaller blocks until all of them are assigned to a motif. In this example, each color correspond to a motif.

Additionally, a source allocation algorithm is used to find the optimal set of parameters for the quantization of each subband for a given compression rate with the goal of maximising the image quality. The source allocation algorithm requires pre-computing the block dictionary, which also includes the quantization codebook as described in section 6.3.1.

6.3.3 Theoretical results

Despite the rapid advancements in the field, the automation of DNA synthesis for large libraries or motifs is still in its developmental stages. DNA sequencing requires very delicate manipulations, and this poses significant challenges in the handling of large libraries. However, the growing interest and research in the field of synthetic biology has sparked new outbreaks and developments in this area.

In this thesis, we take into account the potential advancements in synthetic biology and assume that in the future, the evolution of this field will allow for the manipulation of large motif libraries like the one we consider in our study, with a size of 240k motifs of length 25 nts. Out of the 240k, we select 200k for the creation of a motif dictionary and leave the extra 40k for the encoding of other important fields or headers.

We compare the theoretical rate-distortion curve of the described algorithm to other encoding techniques including:

— Compression using JPEG 2000 & transcoding of the binary stream into motifs (JPEG2000+MOTIFS)
— Compression using JPEG 2000 & transcoding of the binary stream with Paircode (JPEG2000+Paircode)
— Compression using JPEG & transcoding of the binary stream into motifs (JPEG+MOTIFS)
— Compression using JPEG & transcoding of the binary stream with Paircode (JPEG+Paircode)

— Compression using DWT+VQ using a source allocation algorithm & encoding with motifs (VQ+MOTIFS —new proposed encoding)
— Compression using DWT+VQ using a source allocation algorithm & encoding with Paircode (VQ+Paircode)
— JPEG-inspired coder for DNA (JPEG-DNA)

Results are depicted in figure 6.5. The proposed encoding solution is not only comparable to the other tested solutions, but also outperforms some of them, specially for higher rates.

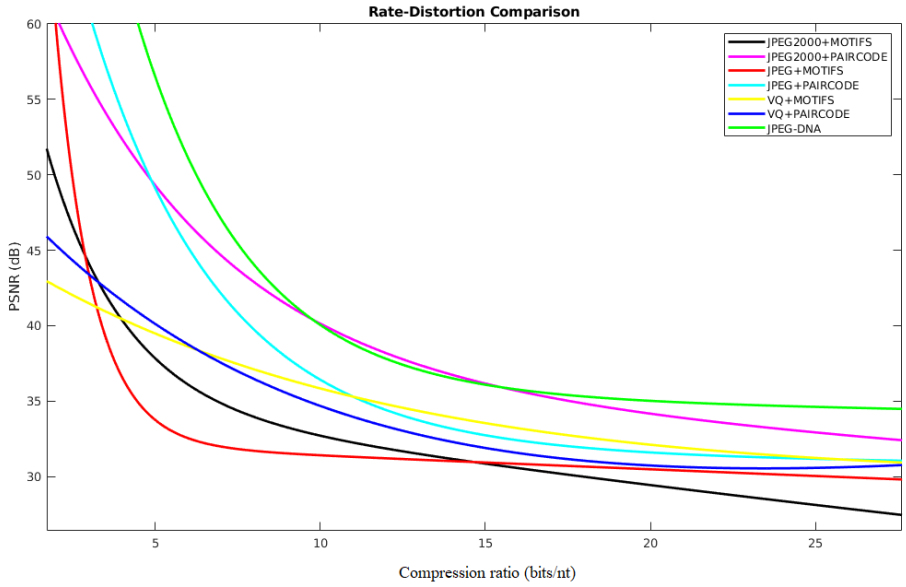


Figure 6.5 – RD comparison for different encoding solutions

6.4 Next steps

In this chapter, we proposed an encoding method adapted to DNA synthesis by ligation of motifs. This method involves synthesizing small, single-stranded DNA fragments, or "motifs," in a laboratory and then using an enzyme to join them together to form a larger, contiguous DNA molecule. This method is still in an early stage of development and there are many unsolved questions about the size of the motif library, the length of the growing strands, and the synthesis yield. This makes it unfeasible to accurately assess the potential of new encoding solutions. However, this work establishes the foundations for a novel algorithm for encoding digital images into DNA.

The method presented is very simple and can provide good solutions in a short amount of time. However, it has the potential to be further improved by extending the scope of the search from individual blocks to the whole image space and the use of spatial optimization algorithms for finding even more efficient solutions to this type of problem.

The algorithm should be also tested in a wet-lab experiment to prove its feasibility. Although handling motif libraries of thousands of motifs it is still an ongoing research, toy experiments could be made considering small libraries of tens to hundreds of motifs.

CHAPTER 7

General conclusions and Perspectives

7.1 Conclusions

Continuing previous works carried out by the Mediacoding group in I3S laboratory, this thesis has focused on proposing mechanisms for robustness and error correction which ensure the decodability of data, more precisely images, stored into DNA. Aiming to reduce the cost and speed up the sequencing step, we introduce nanopore sequencers in the decoding workflow despite the higher error rate introduced by those during the sequencing process compared to more accurate sequencers like Illumina. More precisely, in this thesis we first proposed an alternative encoding schema for the JPEG-inspired DNA coder developed by the Mediacoding group. Although the originally proposed algorithm is very promising in terms of nt-rate, some studies have shown that even small amounts of errors can greatly impact the decodability of the data and, in some cases, the structure of the image can be completely lost. Hence, we propose a method to robustify and ensure the decodability of the JPEG-inspired codec for DNA by encoding the DC coefficients using VQ. We further improve the decoding quality of the data by using NoRM, the noise resistant mapping previously proposed by the Mediacoding group, to map the quantization vectors to the DNA codewords so to reduce the impact of substitution errors on the visual quality of the decoded data. Results show improvements on the PSNR up to 25.4 dB on the decoded data. While the proposed adapted encoding does not guarantee perfect reconstruction of the images, it allows to preserve the structure of the image even in the presence of errors.

Secondly, due to the fact that errors might persist even when applying a robust encoding and error correction mechanisms, we also propose a method for the automatic detection of errors and correction with inpainting. In prior works of the Mediacoding group, it was presented an encoding algorithm that uses DWT followed by vector quantization and the encoding of each wavelet subband independently to obtain a DNA-like representation. As the errors introduced during the biochemical processes will affect the different subbands independently, the noise will appear in different shapes and sizes depending on the type of error and the subband in which it occurred and it can be propagated to the reconstructed image in different ways: large spots and lines or noisy checkerboard and crisscross patterns. However, most inpainting algorithms are built to handle either large occluded areas or smaller, thinner damages. To sidestep all this, we introduce an algorithm for the automatic detection of the damage which acts on the wavelet level, which allows to employ traditional inpainting approaches in a more constrained environment and facilitates the damage detection.

In a previous wet-lab experiment carried out by the Mediacoding group in which two images were stored into DNA and sequenced with the Illumina. Results showed a perfect reconstruction of the data by selecting the most frequent reads for the decoding. Part of the sample was preserved into a stainless still capsule for two years. Following this experiment, the stored DNA was sequenced with MinION (nanopore). The higher error rate introduced by nanopore sequencers brings the need for more sophisticated and adapted decoding schemes so to be able to recover the stored data. The third contribution in this thesis is a decoding workflow, further improved with a consensus algorithm adapted to non-complete dictionaries. The proposed methods not only allowed the decoding of the stored data, but also achieved a very good quality of the reconstructed images.

Fourth, we extracted the error rates from the aforementioned experimental data and use them to validate a new DNA data storage channel simulator proposed by our partners from Imperial College London. The development and use of noise simulators ease the implementation of new algorithms targeting DNA data storage, and more concretely nanopore sequencing. However, their error models are mainly generated from the sequencing of biological data and thus, in most cases not adapted to short synthetic DNA. This experiment constituted a proof of concept of the reliability of the sequencing module of the simulator, providing results with a very similar visual quality and PSNR compared to the experimental results.

Fifth, in this thesis we have carried out a wet-lab experiment to test the robustness of various solutions proposed by the Mediacoding group for the storage of digital images into DNA. More concretely, we compared:

1. A fixed-length solution using DWT and VQ on each quantized subband independently which includes a nucleotide allocation algorithm for the optimization of the compression.
2. A variable-length solution inspired by the main workflow of classical JPEG and adapted to DNA coding.
3. A simple transcoding of the binary output of JPEG into a quaternary-like sequence.

In this wet-lab experiment, we gathered sequencing data from different platforms such as Illumina, MinION and PromethION. Additionally to comparing the quality of the decoded images, we extracted the noise statistics and combined both information to draft some conclusions regarding the robustness for the tested encodings as well as the reliability of the different sequencing platforms. Results showed that the improved accuracy of nanopore sequencing platforms makes them suitable for DNA data storage, bringing this new storage paradigm one step closer to be a reality.

Sixth, we established the theoretical foundations for a novel encoding schema addressing oligonucleotide synthesis by the enzymatic ligation of motifs. Although this emerging technology is in its early stage and still too slow, it promises to become cheaper and faster than phosphoramidite synthesis.

7.2 Discussion

DNA data storage is a rapidly evolving field with a lot of potential. For the past years, DNA has been considered a well suited candidate to replace current storage media due to its remarkable storage capacity and durability. However, there are still several challenges to overcome in order to make DNA data storage practical and economically viable.

One of the major challenges is the pollution caused by the chemicals used in the synthesis of DNA. However, new methods based on harmless aqueous solutions have been developed to reduce this environmental impact. The writing and reading latency of DNA data is still too high, making it more suitable for cold data storage.

The cost of the biochemical processes involved in DNA data storage, including synthesis and sequencing, is still very high, with the current cost of storing 1MB estimated to be €1000 or more. This has prevented the real-world application of this technology for massive data storage. However, there is hope for the future. The cost of sequencing the human genome has dramatically decreased from hundreds of millions of dollars to less than a thousand dollars in just a few years. This trend is expected to continue with the rapid progress and development of synthetic biology.

According to IARPA and the French Academy of Technology, by 2023 the cost for storing 1MB is expected to drop to 1 euro. This means that the first applications for DNA storage will be limited to valuable data that is infrequently accessed. By 2027, the cost is expected to drop to 1 euro per GB, making it a viable solution for general archival. By 2030, the cost is expected to drop to 1 euro per TB, making it a potential solution for large scale data storage.

DNA storage technology is still in its early stages and there are technological challenges to make it a viable alternative data storage solution. However, advances have been made in encoding algorithms, barcoding, and miniaturization and parallelization of DNA synthesis, which has reduced the cost of chemical DNA synthesis. With continued research and development, the future of DNA data storage looks promising and has the potential to revolutionize the way we store digital data.

7.3 Perspectives

7.3.1 Ongoing works

Filtering of DNA reads for robust decoding

In section 4.2.1 we proposed a method for decoding nanopore-sequenced reads. The clustering step of the proposed decoding groups the different reads into clusters according to some header. These headers are included during the formatting process and allow to uniquely identify each sequence so to be able to reconstruct the data encoded in the oligos.

The noise introduced by the biochemical processes involved on DNA data storage —mainly during sequencing— come in the form of substitutions, insertions and deletions of nucleotides. While substitution errors are easier to treat and have a lower impact on the decodability of the data, insertions and deletions (indels) cause shifts on the sequences of nucleotides, often making not possible to correctly identify the oligo headers. In this scenario there are two possible outcomes:

1. The read is misclassified and assigned to a wrong cluster.
2. The read cannot be located into any cluster and, therefore, it is discarded.

In addition, reads with high levels of corruption —even when correctly classified— can negatively affect the quality of the consensus. As a result of the aforementioned, the consensus inferred from the clusters may fail to provide an accurate representation of the reference oligos, interfering in the decoding of the stored images.

Motivated by the idea of improving the clustering process so to be able to infer more accurate consensus, we propose the use of neural networks for the classification of raw reads based on

their quality. The idea is to discard highly corrupted reads that would have a negative impact on the consensus results. This classification step is integrated in the workflow after sequencing and before decoding as depicted in figure 7.1.

The classification task is achieved by training a multilayer perceptron (MLP) classifier with three hidden layers. The outcome of the classification is a label for each processed read with two possible values: noisy or noiseless. Note that reads classified as noiseless will not necessarily be error free but with an error rate lower than a predefined threshold. To make the data compatible with the neural network, the DNA strings are embedded using one-hot encoding, which represents each categorical variable (A, C, T and G) with a binary vector that has one element for each unique label and marks the class label with a 1 and all other elements with 0.

The training was carried out using cross-validation with stochastic gradient descent, sigmoid activation in the hidden layers and softmax on the output layer. The learning rate is adaptive with an initial value of 0.01.

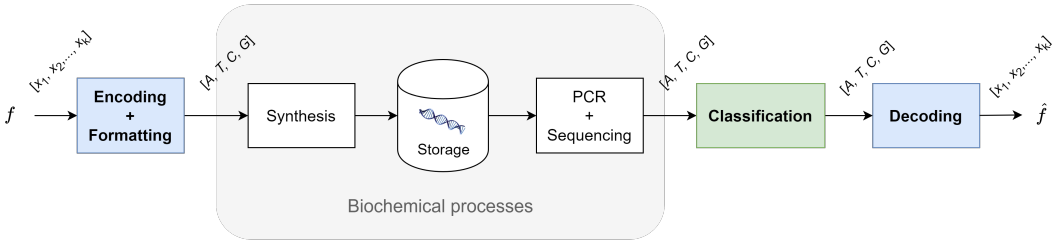


Figure 7.1 – Encoding/decoding workflow including a classification step to filter out highly corrupted reads before decoding.

One of the main constraints for the use of supervised learning is the need for a labeled data set to train the model, which can be challenging in emerging fields as it is the case of DNA data storage. One solution could be the use of DNA noise simulators to generate these data sets without the need of spending vasts amounts of money on wet-lab experiments. For the preliminary experiments, we used the nanopore-sequenced reads from the wet-lab experiment presented in chapter 5. More concretely, we used the sequencing data obtained with MinION-R2C2 (see section 5.4). We selected the two images encoded with the VQ-based fixed-length solution (see section 5.2.1), as it is the only encoding for which we have data from two different images. The original images as well as the decoding results are depicted in figure 5.4. In this experiment, we used the reads from image "kodim23" for training and from "kodim15" for testing.

As explained in section 5.4, MinION-R2C2 provided and extremely low coverage and a high number of under-represented sequences (see table 5.8). As a consequence, considering a read to be noiseless only if it is identical to its reference ($d_L = 0$) might be too restrictive, adding the risk of data loss. Hence, we tested two scenarios. First scenario considers a read to be noiseless when $d_L = 0$ and the second one when $d_L \leq 2$.

The output of the classification step, is a filtered pool of reads, containing only those with high accuracy. The decoding is then carried out as described in section 4.2.1. Additionally, we tested the decoding for the raw reads (i.e. without read filtering) to assess the improvement of the decoding when adding this extra classification step in the workflow. Figure 7.2 shows the decoded results.

Noiseless if $d_L = 0$
PSNR = 40.9 dB

Noiseless if $d_L \leq 2$
PSNR = 47.4 dB

Figure 7.2 – Decoding results after read classification and filtering.

While the image could not be decoded following the method presented in section 4.2.1 using the raw nanopore reads (i.e. with no classification step to filter out the noisy reads), the results when adding this extra classification step in the workflow not only show that the image could be decoded, but also provided a high visual quality in both cases. Although it is not easy to spot the differences between both decoded images, being less restrictive on the classification by considering also reads with low error levels (up to 2 errors per read in this case), contributed to an improvement of 6.5 dB on the PSNR.

Although in this thesis we just show some preliminary results and further study on the topic needs to be done, results show the importance of cleaning the pool of reads to achieve cleaner clusters and better consensus accuracy.

Another asset of the proposed method is that it can be integrated in almost any decoding workflow as an independent module. While it is true that the model should be trained according to the target data, it could be done by simulating the sequencing noise if experimental data is not available. As an example, it would be interesting to combine the proposed classifier with OneConsensus, the clustering and consensus software used in the wet-lab experiment carried out during this thesis (see section 5.5). The consensus sequences obtained with OneConsensus provided a PSNR of 39.65 dB on the decoded image (figure 5.19), which potentially could be further improved if integrating the classification module.

Studies about the ability of the model to generalise to various encoding schemes must also be done. A good starting point would be to try to generalise the model so to handle different encoding algorithms which use the same dictionary. Such is the case of the three encoding methods tested in the wet-lab experiment presented in section 5.2, which ultimately use dictionaries constructed with Paircode.

7.3.2 Future steps

In this thesis we have proposed various methods for enhancing the quality of DNA-stored image retrieval and for handling nanopore sequencing noise to ensure the decodability of the data as well as improve the decoding accuracy. Although the experiments presented in this thesis already show promising results, there is still room for improvement and we believe that with further research and development, they could be further optimized. In the following paragraphs, we will discuss some potential avenues for improvement.

In chapter 3, we propose a method for ensuring the decodability of the JPEG-inspired DNA coder in the presence of sequencing noise. More specifically, we propose to use VQ for the encoding of the more critical information, so to ensure the decodability of the stored images. We further robustify the encoder by using NoRM algorithm for efficiently mapping the quantization vectors to DNA codewords. As explained in section 3.2.3, this algorithm has the goal of mapping vectors which are close in terms of Euclidean distance to DNA codewords which have a Hamming distance equal to 1 from each other (i.e. they differ on 1 position). Although basing this method on the Hamming distance has the advantage of lowering the complexity of the computations, it only considers substitution errors. While this is a good starting point, the mapping algorithm could be further optimized for the DNA data storage channel by also considering insertions and deletions. This could be achieve by using other string similarity metrics such as Levenshtein distance.

Also in chapter 3, we introduce an automatic error detection algorithm with error correction based on inpainting. The algorithm is specialised to act on the wavelet domain, prior to the reconstruction of the image. Although this approach has been tested on simulated data and provided interesting results, it could be further tested and tuned using the experimental data presented in chapters 4 and 5, both of them containing DWT-compressed images.

Regarding the wet-lab experiment presented in chapter 5, a more detailed analysis of the errors introduced during sequenced could provide important insights on the nature of such errors. Information such as the error distribution or the error transition probabilities on both, single bases and k-mers, could provide a deeper understanding of the noisy channel our data goes through. With this information available, the error detection and correction mechanisms could be optimized as well as the encoding algorithms further robustify. Additionally, the creation of accurate noise models would allow to improve the reliability of noise simulators such as the one described in section 4.3.

During the past years, a growing interest on machine learning techniques for DNA coding has led to the publication of various works. As an example, one study applied deep learning algorithms to the process of encoding and decoding digital images stored in DNA, using convolutional neural networks to improve the accuracy of image retrieval (Buterez, 2021). Another study proposed the use of generative adversarial networks (GANs) to improve the robustness of DNA-based data storage against errors and mutations (Pan et al., 2022). These studies demonstrate the potential of machine learning techniques for enhancing the performance of DNA data storage systems, and suggest that further research in this area could lead to new and more effective methods for error correction in DNA data storage.

Bibliography

Alberts, B., Johnson, A., Lewis, J., Raff, M., Roberts, K., & Walter, P. (2002). The structure and function of DNA.

Allentoft, M. E., Collins, M., Harker, D., Haile, J., Oskam, C. L., Hale, M. L., ... Bunce, M. (2012). The half-life of DNA in bone: measuring decay kinetics in 158 dated fossils. Proceedings of the Royal Society B: Biological Sciences, 279(1748), 4724-4733. Consulté sur https://royalsocietypublishing.org/doi/abs/10.1098/rspb.2012.1745 doi: 10.1098/rspb.2012.1745

Alliance, D. D. S. (2021). Preserving Our Digital Legacy: an Introduction to DNA Data Storage (Rapport technique). tech. rep. June.

Alnasir, J. J., Heinis, T., & Carteron, L. (2022). DNA Storage Error Simulator: A Tool for Simulating Errors in Synthesis, Storage, PCR and Sequencing. arXiv preprint arXiv:2205.14437.

Antonini, M., Cruz, L., da Silva, E., Dimopoulou, M., Ebrahimi, T., Foessel, S., ... others (2022, April). DNA-based Media Storage: State-of-the-art, challenges, use cases and requirements version 7.0. Consulté sur https://jpeg.org/jpegdna/documentation.htm

Ashlock, D., & Houghten, S. K. (2009). DNA error correcting codes: No crossover. In 2009 IEEE symposium on computational intelligence in bioinformatics and computational biology (pp. 38–45).

Baker, E. A. G., Goodwin, S., McCombie, W. R., & Ramos, O. M. (2016). Silico: a simulator of long read sequencing in pacbio and oxford nanopore. BioRxiv, 076901.

Beaucage, S., & Caruthers, M. (1981). Deoxynucleoside phosphoramidites—a new class of key intermediates for deoxypolynucleotide synthesis. Tetrahedron letters, 22(20), 1859–1862.

Blawat, M., Gaedke, K., Huetter, I., Chen, X.-M., Turczyk, B., Inverso, S., ... Church, G. M. (2016). Forward error correction for DNA data storage. Procedia Computer Science, 80, 1011–1022.

Bornemann, F., & März, T. (2007). Fast image inpainting based on coherence transport. Journal of Mathematical Imaging and Vision, 28(3), 259–278.

Bornholt, J., Lopez, R., Carmean, D. M., Ceze, L., Seelig, G., & Strauss, K. (2016). A DNA-based archival storage system. ACM SIGOPS Operating Systems Review, 50(2), 637–649.

Borodina, T. A., Lehrach, H., & Soldatov, A. V. (2003). Ligation-based synthesis of oligonucleotides with block structure. Analytical biochemistry, 318(2), 309–313.

Buschmann, T., & Bystrykh, L. V. (2013). Levenshtein error-correcting barcodes for multiplexed DNA sequencing. BMC bioinformatics, 14(1), 272.

Buterez, D. (2021). Scaling up DNA digital data storage by efficiently predicting DNA hybridisation using deep learning. Scientific Reports, 11(1), 20517.

Callaway, E. (2021). Million-year-old mammoth genomes shatter record for oldest ancient DNA. Nature, 590(7847), 537–539.

Church, G. M., Gao, Y., & Kosuri, S. (2012). Next-generation digital information storage in DNA. Science, 1226355.

Clermont, D., Santoni, S., Saker, S., Gomard, M., Gardais, E., & Bizet, C. (2014). Assessment of DNA encapsulation, a new room-temperature DNA storage method. Biopreservation and Biobanking, 12(3), 176–183.

Criminisi, A., Perez, P., & Toyama, K. (2004). Region filling and object removal by exemplar-based image inpainting. IEEE Transactions on Image Processing, 13(9), 1200-1212.

De Marca, J. B., Jayant, N., et al. (1987). An algorithm for assigning binary indices to the codevectors of a multi-dimensional quantizer. In 1987 IEEE International Conference on Communications (ICC'87) (pp. 1128–1132).

Dimopoulou, M. (2020). Encoding techniques for long-term storage of digital images into synthetic DNA (Thèse de doctorat non publiée). Université Côte d'Azur.

Dimopoulou, M., & Antonini, M. (2021). Image storage in DNA using Vector Quantization. In 2020 28th European Signal Processing Conference (EUSIPCO) (pp. 516–520).

Dimopoulou, M., Antonini, M., Barbry, P., & Appuswamy, R. (2019). A biologically constrained encoding solution for long-term storage of images onto synthetic DNA. In 2019 27th European Signal Processing Conference (EUSIPCO).

Dimopoulou, M., Gil San Antonio, E., & Antonini, M. (2020). A quaternary code mapping resistant to the sequencing noise for DNA image coding. In 2020 IEEE 22nd International Workshop on Multimedia Signal Processing (MMSP) (pp. 1–6).

Dimopoulou, M., Gil San Antonio, E., & Antonini, M. (2021a). A JPEG-based image coding solution for data storage on DNA. In 2021 29th European Signal Processing Conference (EUSIPCO) (pp. 786–790).

Dimopoulou, M., Gil San Antonio, E., & Antonini, M. (2021b). A sequencing noise resistant code mapping algorithm for image storage in DNA. In CORESA 2020.

Dimopoulou, M., Gil San Antonio, E., Antonini, M., Barbry, P., & Appuswamy, R. (2019). On the reduction of the cost for encoding/decoding digital images stored on synthetic DNA. In GRETSI 2019.

Doricchi, A., Platnich, C. M., Gimpel, A., Horn, F., Earle, M., Lanzavecchia, G., ... others (2022). Emerging Approaches to DNA Data Storage: Challenges and Prospects. ACS nano, 16(11), 17552–17571.

Erlich, Y., & Zielinski, D. (2016). Capacity-approaching DNA storage. bioRxiv, 074237.

Faucon, P. C., Balachandran, P., & Crook, S. (2017). Snaresim: synthetic nanopore read simulator. In 2017 IEEE International Conference on Healthcare Informatics (ICHI) (pp. 338–344).

Ferguson, J. M., & Smith, M. A. (2019). Squigglekit: a toolkit for manipulating nanopore signal data. Bioinformatics, 35(24), 5372–5373.

Gil San Antonio, E., Dimopoulou, M., Antonini, M., Barbry, P., & Appuswamy, R. (2021). Decoding of nanopore-sequenced synthetic DNA storing digital images. In 2021 IEEE International Conference on Image Processing (ICIP).

Gil San Antonio, E., Heinis, T., Carteron, L., Dimopoulou, M., & Antonini, M. (2021). Nanopore Sequencing Simulator for DNA Data Storage. In 2021 International Conference on Visual Communications and Image Processing (VCIP) (pp. 1–5).

Gil San Antonio, E., Piretti, M., Dimopoulou, M., & Antonini, M. (2021). Robust image coding on synthetic DNA: Reducing sequencing noise with inpainting. In 2020 25th International Conference on Pattern Recognition (ICPR) (pp. 9859–9865).

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

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

Hawkins, J. A., Jones, S. K., Finkelstein, I. J., & Press, W. H. (2018). Error-correcting DNA barcodes for high-throughput sequencing. bioRxiv, 315002.

Heckel, R., Mikutis, G., & Grass, R. N. (2019, 04 Jul). A Characterization of the DNA Data Storage Channel. Scientific Reports, 9(1).

Hughes, R. A., & Ellington, A. D. (2017). Synthetic DNA synthesis and assembly: putting the synthetic in synthetic biology. Cold Spring Harbor perspectives in biology, 9(1), a023812.

Jain, M., Fiddes, I. T., Miga, K. H., Olsen, H. E., Paten, B., & Akeson, M. (2015). Improved data analysis for the minion nanopore sequencer. Nature methods, 12(4), 351–356.

Lau, H. Y., & Botella, J. R. (2017). Advanced DNA-based point-of-care diagnostic methods for plant diseases detection. Frontiers in plant science, 8, 2016.

Leggett, R. M., Heavens, D., Caccamo, M., Clark, M. D., & Davey, R. P. (2016). NanoOK: multi-reference alignment analysis of nanopore sequencing data, quality and error profiles. Bioinformatics, 32(1), 142–144.

Li, H. (2018). Minimap2: pairwise alignment for nucleotide sequences. Bioinformatics, 34(18), 3094–3100.

Li, Y., Han, R., Bi, C., Li, M., Wang, S., & Gao, X. (2018). DeepSimulator: a deep simulator for nanopore sequencing. Bioinformatics, 34(17), 2899–2908.

Li, Y., Wang, S., Bi, C., Qiu, Z., Li, M., & Gao, X. (2020). DeepSimulator1. 5: a more powerful, quicker and lighter simulator for nanopore sequencing. Bioinformatics, 36(8), 2578–2580. doi: 10.1093/bioinformatics/btz963

Liu, X., Li, Q., Wang, X., Zhou, X., He, X., Liao, Q., ... Zhang, Y. (2015). Evaluation of DNA/RNAshells for room temperature nucleic acids storage. Biopreservation and biobanking, 13(1), 49–55.

Magi, A., Semeraro, R., Mingrino, A., Giusti, B., & D'aurizio, R. (2018). Nanopore sequencing data analysis: state of the art, applications and challenges. Briefings in bioinformatics, 19(6), 1256–1272.

Marinelli, E., & Appuswamy, R. (2021). Onejoin: Cross-architecture, scalable edit similarity join for DNA data storage using oneapi.

Marinelli, E., Ghabach, E., Yan, Y., Bolbroe, T., Sella, O., Heinis, T., & Appuswamy, R. (2022). Digital preservation with synthetic DNA. In Transactions on large-scale data-and knowledge-centered systems li (pp. 119–135). Springer.

Nguyen, B., Sinistore, J., Smith, J. A., Arshi, P. S., Johnson, L. M., Kidman, T., ... Strauss, K. (2020). Architecting datacenters for sustainability: Greener data storage using synthetic DNA. Electronics Goes Green 2020.

Ono, Y., Asai, K., & Hamada, M. (2021). PBSIM2: a simulator for long-read sequencers with a novel generative model of quality scores. Bioinformatics, 37(5), 589–595.

Pan, C., Tabatabaei, S. K., Tabatabaei Yazdi, S. H., Hernandez, A. G., Schroeder, C. M., & Milenkovic, O. (2022). Rewritable two-dimensional DNA-based data storage with machine learning reconstruction. Nature communications, 13(1), 2984.

Pan, C., Yazdi, S., Tabatabaei, S. K., Hernandez, A. G., Schroeder, C., & Milenkovic, O. (2019). Image processing in DNA. arXiv preprint arXiv:1910.10095.

Pereira, R., Oliveira, J., & Sousa, M. (2020). Bioinformatics and computational tools for next-generation sequencing analysis in clinical genetics. Journal of clinical medicine, 9(1), 132.

Porechop, R. W. (2018). https://github.com/rrwick/Porechop..

Rohrandt, C., Kraft, N., Gießelmann, P., Brändl, B., Schuldt, B. M., Jetzek, U., & Müller, F.-J. (2018). Nanopore SimulatION — a raw data simulator for Nanopore Sequencing. In 2018 IEEE International Conference on Bioinformatics and Biomedicine (BIBM) (pp. 1–8).

Sanger, F., Nicklen, S., & Coulson, A. R. (1977). DNA sequencing with chain-terminating inhibitors. Proceedings of the national academy of sciences, 74(12), 5463–5467.

Schwarz, M., Welzel, M., Kabdullayeva, T., Becker, A., Freisleben, B., & Heider, D. (2020). MESA: automated assessment of synthetic DNA fragments and simulation of DNA synthesis, storage, sequencing and PCR errors. Bioinformatics, 36(11), 3322–3326.

Takahashi, C. N., Nguyen, B. H., Strauss, K., & Ceze, L. (2019). Demonstration of end-to-end automation of DNA data storage. Scientific reports, 9(1), 1–5.

Volden, R., Palmer, T., Byrne, A., Cole, C., Schmitz, R. J., Green, R. E., & Vollmers, C. (2018). Improving nanopore read accuracy with the R2C2 method enables the sequencing of highly multiplexed full-length single-cell cDNA. Proceedings of the National Academy of Sciences, 115(39), 9726–9731.

Wang, Y., Zhao, Y., Bollas, A., Wang, Y., & Au, K. F. (2021). Nanopore sequencing technology, bioinformatics and applications. Nature biotechnology, 39(11), 1348–1365.

Washetine, K., Heeke, S., Ribeyre, C., Bourreau, C., Normand, C., Blons, H., ... others (2019). Dnashell protects DNA stored at room temperature for downstream next-generation sequencing studies. Biopreservation and biobanking, 17(4), 352–354.

Wick, R. R. (2019). Badread: simulation of error-prone long reads. Journal of Open Source Software, 4(36), 1316.

Yang, C., Chu, J., Warren, R. L., & Birol, I. (2017). NanoSim: nanopore sequence read simulator based on statistical characterization. GigaScience, 6(4), gix010.

Young, P., Lai, A., Hodosh, M., & Hockenmaier, J. (2014). From image descriptions to visual denotations: New similarity metrics for semantic inference over event descriptions. Transactions of the Association for Computational Linguistics, 2, 67–78.

Zeng, J., Cai, H., Peng, H., Wang, H., Zhang, Y., & Akutsu, T. (2020). Causalcall: Nanopore basecalling using a temporal convolutional network. Frontiers in Genetics, 10, 1332.

Zhang, H., & Zhang, Q. (2017). Embedjoin: Efficient edit similarity joins via embeddings. In Proceedings of the 23rd ACM SIGKDD International conference on knowledge discovery and data mining (pp. 585–594).

Zorita, E., Cusco, P., & Filion, G. J. (2015). Starcode: sequence clustering based on all-pairs search. Bioinformatics, 31(12), 1913–1919.

Web pages

DNAScript. (2022). Enzymatic DNA synthesis (EDS): Transforming molecular biology (DNA script). Consulté le 2023-01-12, sur https://www.dnascript.com/technology/

Illumina. (s. d.). Illumina NextSeq 550 System. Consulté le 2023-01-12, sur https://www.illumina.com/systems/sequencing-platforms/nextseq/order-nextseq-550.html

Imagene. (s. d.). DNASHELL (Imagene). Consulté le 2023-01-12, sur http://www.imagene.eu/dnashell-rnashell-en/dnashell-en/

McNerney, M. (2019). The Data Center Dilemma: Is Our Data Destroying the Environment? (Data Center Knowledge). Consulté le 2023-01-12, sur https://www.datacenterknowledge.com/industry-perspectives/data-center-dilemma-our-data-destroying-environment

ONT. (s. d.). ONT products. Consulté le 2023-01-12, sur https://nanoporetech.com/products

PraxiLabs. (s. d.). DNA Sequencing: Definition, Importance, Methods, Facts, and More (Praxilabs). Consulté le 2023-01-12, sur http://en.biomarker.com.cn/platforms/illumina

Taylor, P. (2021). Amount of data created, consumed, and stored 2010-2020, with forecasts to 2025 (Statista). Consulté le 2023-01-12, sur https://www.statista.com/statistics/871513/worldwide-data-created/#:~:text=The%20total%20amount%20of%20data,to%20more%20than%20180%20zettabytes

Vivadifferences. (s. d.). 5 Differences Between Frameshift and Point Mutations (Vivadifferences). Consulté le 2023-01-12, sur https://vivadifferences.com/frameshift-vs-point-mutations-the-underlying-differences/

List of Figures

List of Algorithms

Appendix

A Test sequences with repetition of patterns

A.1 Correctly sequenced codewerds when pattern repetition with MiniON-R2C2

3 nucleotides

'ACA' 'CAA' 'GTA' 'ACT' 'AGT' 'TCT' 'CAT' 'CTT' 'GAT' 'ACC' 'TCC' 'TGC' 'CAC' 'ATG' 'ACG' 'TCG' 'TGG' 'GTG'

4 nucleotides

'ATTC' 'ATTG' 'ATCA' 'ATCT' 'ATGA' 'ATGT' 'ACTA' 'ACTC' 'ACTG' 'ACCA' 'ACGA' 'AGAT' 'AGAC' 'AGTC' 'AGTG' 'AGGA' 'AGGT' 'TAAT' 'TAAC' 'TATC' 'TATG' 'TACA' 'TAGA' 'TAGT' 'TCAT' 'TCAC' 'TCAG' 'TCTA' 'TCCA' 'TCCT' 'TCGT' 'TGAT' 'TGAC' 'TGTA' 'TGTC' 'TGGA' 'TGGT' 'CAAT' 'CAAC' 'CATA' 'CATC' 'CAGA' 'CAGT' 'CTAT' 'CTTA' 'CTTC' 'CTCA' 'CTGA' 'CTGT' 'GAAT' 'GAAC' 'GAAG' 'GATA' 'GATG' 'GACA' 'GACT' 'GAGT' 'GTAT' 'GTAG' 'GTTA' 'GTTC' 'GTTG' 'GTGA'

5 nucleotides

'ATATC' 'ATATG' 'ATACA' 'ATACT' 'ATACC' 'ATACG' 'ATAGA' 'ATAGC' 'ATTAC' 'ATTAG' 'ATTCA' 'ATTCT' 'ATTGA' 'ATTGT' 'ATTGC' 'ATTGG' 'ATCAA' 'ATCAT' 'ATCAC' 'ATCAG' 'ATCTA' 'ATCTT' 'ATCTG' 'ATGAA' 'ATGAT' 'ATGAC' 'ATGAG' 'ATGTA' 'ATGTT' 'ATGTC' 'ATGTG' 'ACATA' 'ACATT' 'ACATC' 'ACATG' 'ACACA' 'ACACT' 'ACACC' 'ACACG' 'ACAGA' 'ACAGT' 'ACAGC' 'ACAGG' 'ACTAA' 'ACTAT' 'ACTAC' 'ACTAG' 'ACTCT' 'ACTGA' 'ACTGT' 'ACTGC' 'ACTGG' 'ACCAA' 'ACCAT' 'ACCAG' 'ACCTT' 'ACCTG' 'ACGAA' 'ACGAT' 'ACGAC' 'ACGAG' 'AGATC' 'AGATG' 'AGACA' 'AGACT' 'AGACC' 'AGAGC' 'AGTAA' 'AGTAT' 'AGTAC' 'AGTAG' 'AGTCT' 'AGTCC' 'AGTCG' 'AGTGA' 'AGTGT' 'AGTGC' 'AGCAA' 'AGCAT' 'AGCAC' 'AGCAG' 'AGGAA' 'AGGAT' 'AGGAC' 'AGGTT' 'AGGTC' 'TAATC' 'TAATG' 'TAACA' 'TAACG' 'TAAGA' 'TAAGT' 'TAAGC' 'TAAGG' 'TATCA' 'TATCT' 'TATCG' 'TATGA' 'TATGT' 'TATGC' 'TATGG' 'TACAA' 'TACAC' 'TACAG' 'TACTA' 'TACTT' 'TACTG' 'TAGAA' 'TAGAT' 'TAGAC' 'TAGAG' 'TAGTA' 'TAGTT' 'TAGTC' 'TCATA' 'TCATT' 'TCATG' 'TCACA' 'TCACT' 'TCAGA' 'TCAGT' 'TCAGC' 'TCAGG' 'TCTAA' 'TCTAT' 'TCTAG' 'TCTGA' 'TCTGT' 'TCTGC' 'TCTGG' 'TCCAA' 'TCCAT' 'TCCAG' 'TCCTG' 'TCGTA' 'TCGTT' 'TCGTC' 'TCGTG' 'TGATA' 'TGATT' 'TGATC' 'TGATG' 'TGACA' 'TGACT' 'TGACC' 'TGACG' 'TGAGA' 'TGAGT' 'TGAGC' 'TGAGG' 'TGTAA' 'TGTAT' 'TGTAC' 'TGTAG' 'TGTCA' 'TGTCT' 'TGTCC' 'TGTCG' 'TGTGA' 'TGTGC' 'TGCTA' 'TGCTT' 'TGCTC' 'TGCTG' 'TGGAA' 'TGGAT' 'TGGAC' 'TGGAG' 'TGGTA' 'TGGTT' 'TGGTC' 'TGGTG' 'CAATC' 'CAACA' 'CAACT' 'CAACC' 'CAACG' 'CAAGA' 'CAAGT' 'CAAGC' 'CAAGG' 'CATAA' 'CATAT' 'CATAC' 'CATAG' 'CATCA' 'CATCT' 'CATCG' 'CATGC'

'CATGG' 'CACAA' 'CACAT' 'CACAC' 'CACAG' 'CACTA' 'CACTT' 'CACTG' 'CAGAT' 'CAGAC' 'CAGAG' 'CAGTA' 'CAGTT' 'CAGTC' 'CAGTG' 'CTATA' 'CTATT' 'CTATC' 'CTATG' 'CTACA' 'CTACT' 'CTACG' 'CTTAT' 'CTTAC' 'CTTAG' 'CTTCA' 'CTTCC' 'CTTGA' 'CTTGT' 'CTTGC' 'CTTGG' 'CTCAA' 'CTCAT' 'CTCAG' 'CTCTA' 'CTGAA' 'CTGAT' 'CTGAC' 'CTGAG' 'CTGTA' 'CTGTT' 'CTGTC' 'CTGTG' 'GAATA' 'GAATT' 'GAATC' 'GAACA' 'GAACT' 'GAACC' 'GAACG' 'GAAGT' 'GATAA' 'GATAT' 'GATAC' 'GATAG' 'GATCT' 'GATGA' 'GATGT' 'GATGC' 'GATGG' 'GACAA' 'GACAT' 'GACAC' 'GACAG' 'GACTA' 'GACTT' 'GACTC' 'GACTG' 'GAGAA' 'GAGAT' 'GAGAC' 'GAGAG' 'GAGTA' 'GAGTT' 'GAGTC' 'GTATA' 'GTATT' 'GTATC' 'GTATG' 'GTACA' 'GTACT' 'GTAGT' 'GTAGC' 'GTTAA' 'GTTAT' 'GTTAC' 'GTTAG' 'GTTCA' 'GTTCT' 'GTTCC' 'GTTCG' 'GTTGA' 'GTTGT' 'GTTGC' 'GTTGG' 'GTCAA' 'GTCAT' 'GTCAC' 'GTCAG' 'GTCTA' 'GTCTT' 'GTCTC' 'GTCTG' 'GTGAT' 'GTGAC' 'GTGAG' 'GTGTA' 'GTGTT' 'GTGTC'

A.2 Correctly sequenced codewerds when pattern repetition with PromethION

3 nucleotides

'TAA' 'TGA' 'ACT' 'TAT' 'TGT' 'CAT' 'CTT' 'GAT' 'ATC' 'ACC' 'AGC' 'TCC' 'CAC' 'CTC' 'GAC' 'GTC' 'ATG' 'ACG' 'TAG' 'TGG' 'GTG'

4 nucleotides

'ATAC' 'ATAG' 'ATTA' 'ATTC' 'ATTG' 'ATCA' 'ATCT' 'ATGA' 'ATGT' 'ACAG' 'ACTA' 'ACTC' 'ACTG' 'ACCA' 'ACCT' 'AGAT' 'AGAC' 'AGTA' 'AGTC' 'AGTG' 'AGCA' 'AGCT' 'AGGT' 'TAAT' 'TAAG' 'TATG' 'TACT' 'TAGT' 'TCAT' 'TCAC' 'TCTG' 'TCCA' 'TCGT' 'TGAT' 'TGAC' 'TGAG' 'TGTA' 'TGTC' 'TGCA' 'TGCT' 'TGGA' 'TGGT' 'CAAT' 'CAAC' 'CATA' 'CATC' 'CATG' 'CACA' 'CACT' 'CTAT' 'CTAG' 'CTTA' 'CTTC' 'CTTG' 'CTCA' 'CTGT' 'GAAT' 'GAAC' 'GAAG' 'GATA' 'GATG' 'GACA' 'GACT' 'GTAT' 'GTAG' 'GTTA' 'GTTG' 'GTCA' 'GTCT' 'GTGA' 'GTGT'

5 nucleotides

'ATATT' 'ATATC' 'ATATG' 'ATACA' 'ATACT' 'ATACC' 'ATACG' 'ATAGA' 'ATAGT' 'ATAGC' 'ATTAA' 'ATTAT' 'ATTAC' 'ATTAG' 'ATTCA' 'ATTCT' 'ATTCC' 'ATTCG' 'ATTGA' 'ATTGT' 'ATTGC' 'ATTGG' 'ATCAA' 'ATCAT' 'ATCAC' 'ATCTA' 'ATCTT' 'ATCTC' 'ATCTG' 'ATGAA' 'ATGAT' 'ATGAC' 'ATGAG' 'ATGTA' 'ATGTT' 'ATGTC' 'ATGTG' 'ACATA' 'ACATT' 'ACATC' 'ACATG' 'ACACA' 'ACACT' 'ACACC' 'ACACG' 'ACAGA' 'ACAGT' 'ACAGC' 'ACTAA' 'ACTAT' 'ACTAC' 'ACTAG' 'ACTCA' 'ACTCT' 'ACTCC' 'ACTGA' 'ACTGT' 'ACTGC' 'ACTGG' 'ACCAA' 'ACCAT' 'ACCAC' 'ACCTA' 'ACCTT' 'ACCTC' 'ACCTG' 'ACGAA' 'ACGAT' 'ACGAC' 'ACGTA' 'ACGTT' 'ACGTC' 'ACGTG' 'AGATA' 'AGATT' 'AGATC' 'AGACA' 'AGACT' 'AGACC' 'AGAGA' 'AGAGC' 'AGAGG' 'AGTAA' 'AGTAT' 'AGTAC' 'AGTAG' 'AGTCA' 'AGTCT' 'AGTCC' 'AGTGA' 'AGTGT' 'AGTGC' 'AGTGG' 'AGCAA' 'AGCAT' 'AGCAC' 'AGCAG' 'AGCTA' 'AGCTC' 'AGCTG' 'AGGAA' 'AGGAT' 'AGGAC' 'AGGAG' 'AGGTA' 'AGGTT' 'AGGTC' 'AGGTG'

’TAATA’ ’TAATT’ ’TAATC’ ’TAATG’ ’TAACA’ ’TAACT’ ’TAACC’ ’TAACG’ ’TAAGA’
’TAAGT’ ’TAAGC’ ’TAAGG’ ’TATAA’ ’TATAC’ ’TATAG’ ’TATCA’ ’TATCT’ ’TATCC’
’TATCG’ ’TATGA’ ’TATGT’ ’TATGC’ ’TATGG’ ’TACAA’ ’TACAT’ ’TACAC’ ’TACAG’
’TACTA’ ’TACTT’ ’TACTC’ ’TACTG’ ’TAGAA’ ’TAGAT’ ’TAGAG’ ’TAGTA’ ’TAGTT’
’TAGTC’ ’TAGTG’ ’TCATA’ ’TCATT’ ’TCATC’ ’TCATG’ ’TCACA’ ’TCACT’ ’TCACC’
’TCACG’ ’TCAGT’ ’TCAGC’ ’TCAGG’ ’TCTAA’ ’TCTAT’ ’TCTAG’ ’TCTCA’ ’TCTCT’
’TCTCC’ ’TCTGA’ ’TCTGT’ ’TCTGC’ ’TCTGG’ ’TCCAA’ ’TCCAT’ ’TCCAC’ ’TCCAG’
’TCCTA’ ’TCCTT’ ’TCCTC’ ’TCCTG’ ’TCGAA’ ’TCGAT’ ’TCGAC’ ’TCGTA’ ’TCGTT’
’TCGTC’ ’TCGTG’ ’TGATA’ ’TGATT’ ’TGATC’ ’TGATG’ ’TGACA’ ’TGACT’ ’TGACC’
’TGACG’ ’TGAGA’ ’TGAGT’ ’TGAGC’ ’TGAGG’ ’TGTAA’ ’TGTAT’ ’TGTAC’ ’TGTAG’
’TGTCA’ ’TGTCT’ ’TGTCC’ ’TGTCG’ ’TGTGA’ ’TGTGT’ ’TGTGC’ ’TGTGG’ ’TGCAA’
’TGCAT’ ’TGCAC’ ’TGCAG’ ’TGCTA’ ’TGCTT’ ’TGCTC’ ’TGCTG’ ’TGGAA’ ’TGGAT’
’TGGAC’ ’TGGAG’ ’TGGTA’ ’TGGTT’ ’TGGTC’ ’TGGTG’ ’CAATA’ ’CAATT’ ’CAATC’
’CAATG’ ’CAACA’ ’CAACT’ ’CAACC’ ’CAACG’ ’CAAGT’ ’CAAGC’ ’CATAA’ ’CATAT’
’CATAC’ ’CATAG’ ’CATCA’ ’CATCT’ ’CATCC’ ’CATCG’ ’CATGA’ ’CATGT’ ’CATGC’
’CATGG’ ’CACAA’ ’CACAT’ ’CACAC’ ’CACAG’ ’CACTA’ ’CACTT’ ’CACTC’ ’CACTG’
’CAGAA’ ’CAGAT’ ’CAGAC’ ’CAGTT’ ’CAGTC’ ’CAGTG’ ’CTATA’ ’CTATT’ ’CTATC’
’CTATG’ ’CTACT’ ’CTACC’ ’CTACG’ ’CTAGA’ ’CTAGT’ ’CTAGC’ ’CTTAA’ ’CTTAT’
’CTTAC’ ’CTTAG’ ’CTTCA’ ’CTTCC’ ’CTTCG’ ’CTTGA’ ’CTTGT’ ’CTTGC’ ’CTTGG’
’CTCAA’ ’CTCAT’ ’CTCAC’ ’CTCAG’ ’CTCTA’ ’CTCTT’ ’CTCTC’ ’CTCTG’ ’CTGAA’
’CTGAT’ ’CTGAC’ ’CTGAG’ ’CTGTA’ ’CTGTT’ ’CTGTC’ ’CTGTG’ ’GAATA’ ’GAATT’
’GAATC’ ’GAATG’ ’GAACA’ ’GAACT’ ’GAACG’ ’GAAGT’ ’GAAGC’ ’GAAGG’ ’GATAA’
’GATAT’ ’GATAC’ ’GATAG’ ’GATCA’ ’GATCT’ ’GATCC’ ’GATCG’ ’GATGA’ ’GATGT’
’GATGC’ ’GATGG’ ’GACAA’ ’GACAT’ ’GACAC’ ’GACTA’ ’GACTT’ ’GACTC’ ’GACTG’
’GAGAT’ ’GAGAC’ ’GAGAG’ ’GAGTA’ ’GAGTC’ ’GAGTG’ ’GTATA’ ’GTATT’ ’GTATC’
’GTATG’ ’GTACA’ ’GTACT’ ’GTACC’ ’GTAGA’ ’GTAGT’ ’GTAGC’ ’GTAGG’ ’GTTAA’
’GTTAT’ ’GTTAC’ ’GTTAG’ ’GTTCC’ ’GTTCG’ ’GTTGA’ ’GTTGT’ ’GTTGC’ ’GTTGG’
’GTCAA’ ’GTCAT’ ’GTCAC’ ’GTCAG’ ’GTCTA’ ’GTCTT’ ’GTCTC’ ’GTCTG’ ’GTGAA’
’GTGAT’ ’GTGAC’ ’GTGAG’ ’GTGTA’ ’GTGTT’ ’GTGTC’ ’GTGTG’

Stockage intelligent sur ADN synthetique pour l'archivage des images numeriques

Eva GIL SAN ANTONIO

Resume

La croissance rapide de la consommation de données numériques, communément appelée "l'explosion des données", présente un défi important pour le stockage des données. L'univers numérique devrait atteindre 175 zettaoctets d'ici 2025, une grande partie de ces données étant rarement consultées, mais nécessitant toujours un archivage sécurisé pour des raisons de sécurité et de conformité réglementaire. Les dispositifs de stockage conventionnels, tels que les disques durs, ont une durée de vie limitée de 10 à 20 ans, ce qui rend nécessaire de trouver des solutions alternatives pour la préservation des données à long terme qui soient non seulement rentables, mais également économes en énergie. Des études récentes ont montré que l'ADN est un candidat très prometteur pour l'archivage à long terme des données numériques. L'ADN a une capacité allant jusqu'à 215 pétaoctets par gramme et une durée de vie théorique allant jusqu'à 1 000 ans, ce qui en fait une option appropriée pour stocker de grandes quantités de données pendant des siècles, voire plus. Cependant, le processus d'encodage des données numériques dans un flux quaternaire composé des symboles A, T, C et G, qui représentent les quatre composants de la molécule d'ADN, tout en respectant d'importantes contraintes d'encodage, fait l'objet de recherches en cours. Des travaux pionniers ont proposé différents algorithmes pour le codage de l'ADN, mais des améliorations sont encore possibles. Dans ce contexte, une nouvelle génération de séquenceurs utilisant des nanopores offre la possibilité de lire des brins d'ADN beaucoup plus rapidement et à moindre coût, avec l'inconvénient d'un taux d'erreur plus élevé. Cette thèse porte sur l'étude de ces erreurs afin d'adapter et de rendre encore plus robuste le codage quaternaire des données. De plus, des techniques de post-traitement adaptées au contexte de stockage des données ADN sont proposées pour corriger les erreurs restantes après décodage. Les résultats d'expériences en laboratoire sont présentés dans lesquels diverses images ont été stockées dans l'ADN à l'aide de différentes méthodes de codage et séquencées à l'aide de différentes technologies telles que Illumina et nanopore. Nous présentons une étude des erreurs introduites avec chaque plate-forme de séquençage et la robustesse des différentes solutions de codage testées expérimentalement. L'objectif de cette recherche est de contribuer au développement de systèmes efficaces et fiables de stockage d'archives sur ADN.

Mots-clés: Stockage de données ADN, Correction des erreurs, Décodage robuste.

Abstract

The rapid growth of digital data, commonly referred to as the "data explosion," presents a significant challenge for data storage. The digital universe is projected to reach 175 zettabytes by 2025, with a large portion of this data being infrequently accessed, yet still requiring safe archival for security and regulatory compliance reasons. Conventional storage devices, such as hard drives, have a limited lifespan of 10-20 years, making it necessary to find new solutions for long-term data preservation that are not only cost-effective, but also energy-efficient. Recent studies have shown that DNA is a very promising candidate for the long-term archival storage of digital data. DNA has a capacity of up to 215 petabytes per gram and a theoretical lifespan of up to 1,000 years, making it a suitable option for storing large amounts of data for centuries or even longer. However, the process of encoding digital data into a quaternary stream made up of the symbols A, T, C and G, which represent the four components of the DNA molecule, while also respecting important encoding constraints, has been a subject of