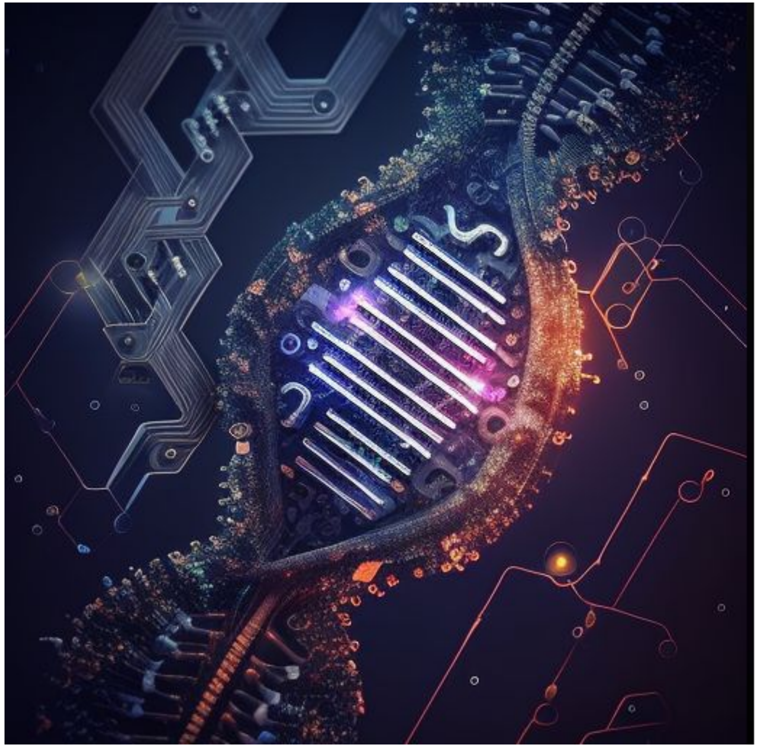
QUANTUM FORMALISM

quantumformalism.com

# Classical-to-Quantum Sequence Encoding in Genomics

Nouhaila Innan & Dr. Muhammed Al-Zafar Khan

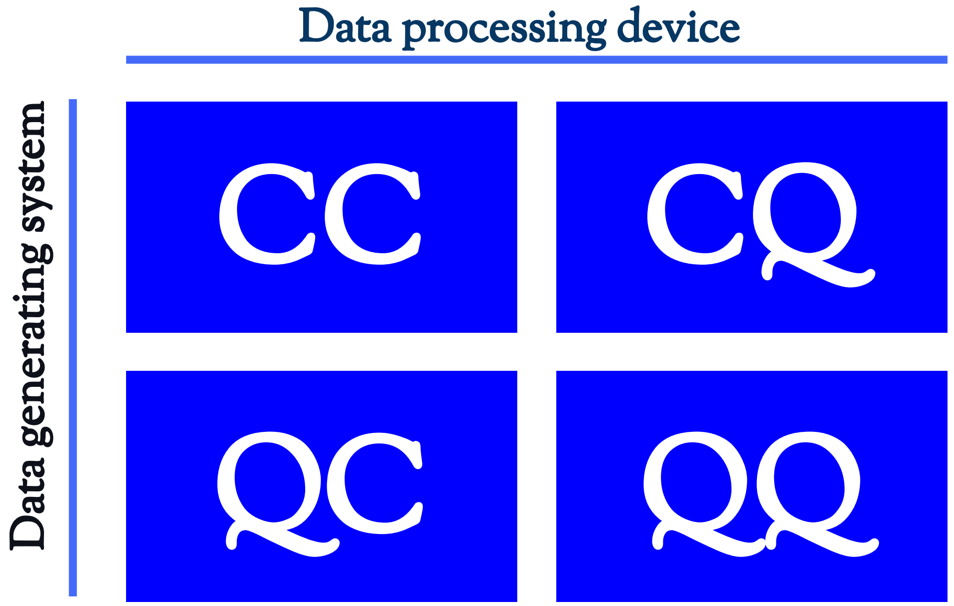


## Outline

- Introduction
- Existing Quantum Data Encoding Methods
- New Quantum Data Encoding Methods:
    - Lossless Compression-Inspired Encoding Schemes
    - Entropy-Based Methods
    - Energy-Based Methods For Testing Encoded Sequences
- Conclusion & Future Work

## Introduction

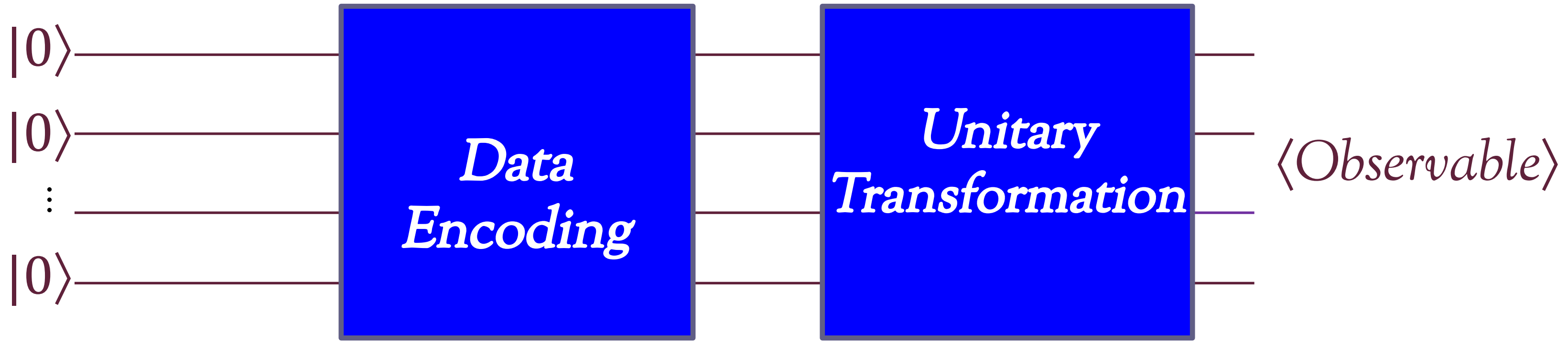
- Quantum Machine Learning (QML) is a research area that explores the interplay of ideas from Quantum Computing and Machine Learning.



*C - Classical, Q - Quantum*

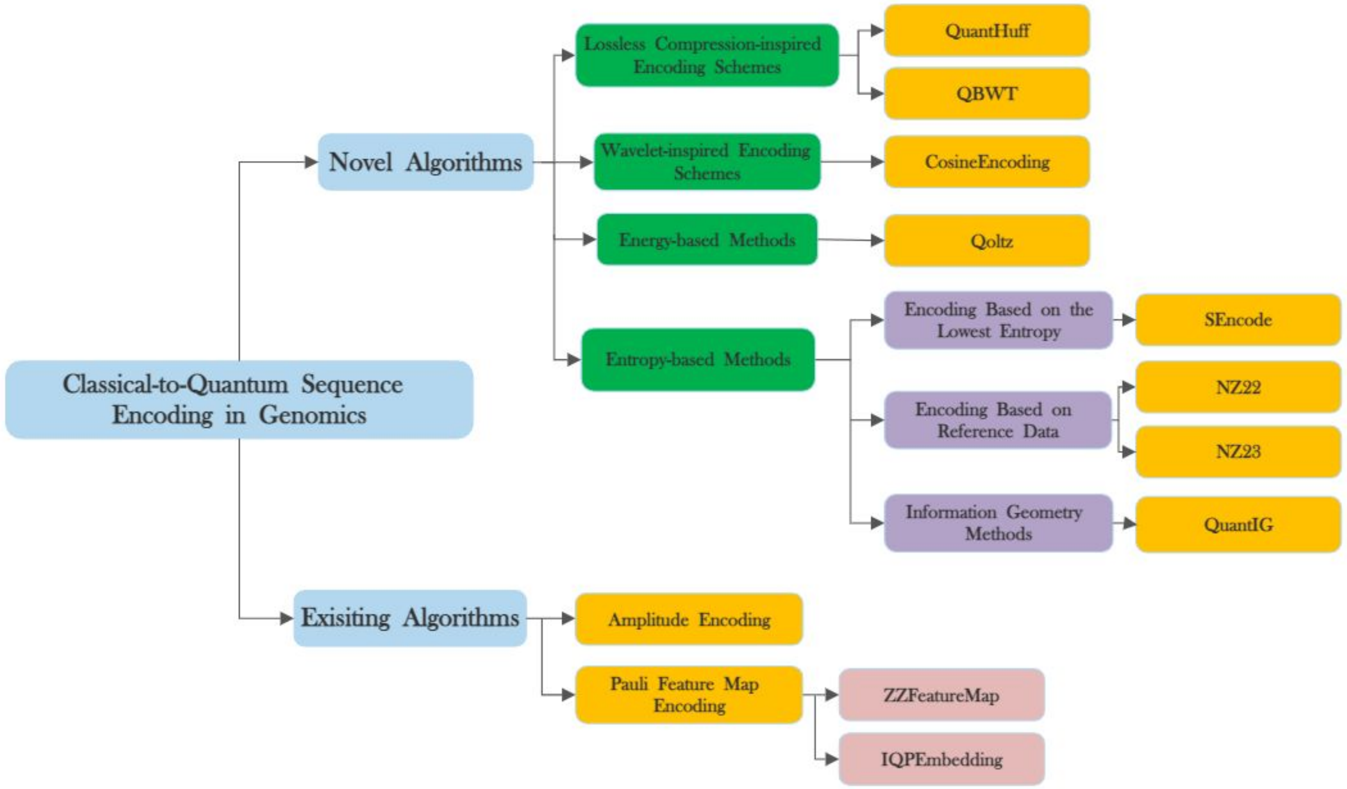
## Introduction (Cont.)

- A Quantum Machine Learning Model Circuit:



**Prepare some initial state** **Encode data** **Other operations** **Measure**

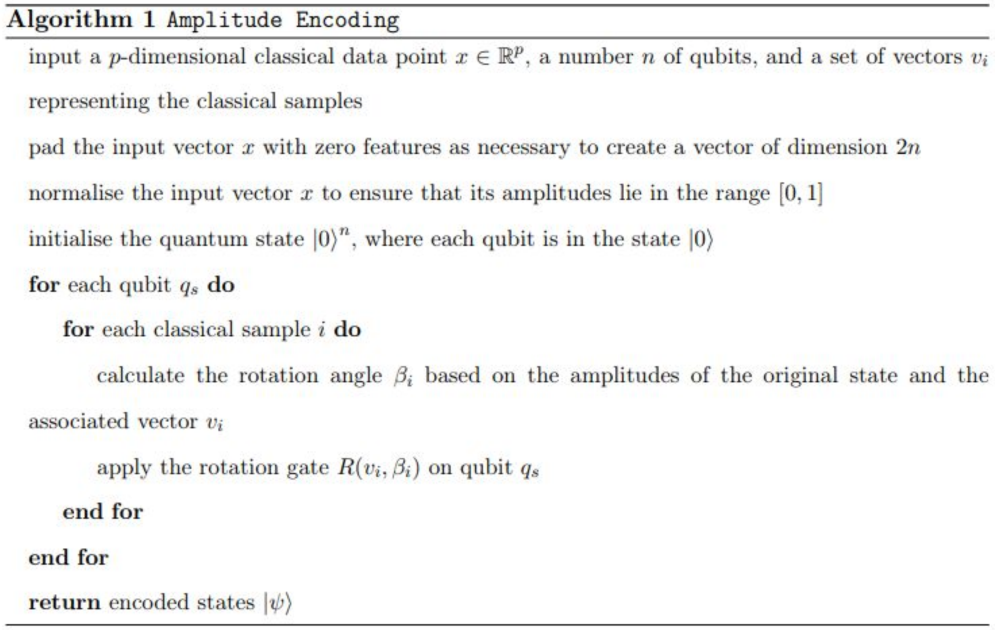
## All Methods


Classical-to-Quantum Sequence Encoding in Genomics
Novel Algorithms
Lossless Compression-inspired Encoding Schemes
QuantHuff
QBWT
Wavelet-inspired Encoding Schemes
CosineEncoding
Energy-based Methods
Qoltz
Entropy-based Methods
Encoding Based on the Lowest Entropy
SEncode
Encoding Based on Reference Data
NZ22
NZ23
Information Geometry Methods
QuantIG
Exisiting Algorithms
Amplitude Encoding
Pauli Feature Map Encoding
ZZFeatureMap
IQPEmbedding

## Existing Methods

## Amplitude Encoding

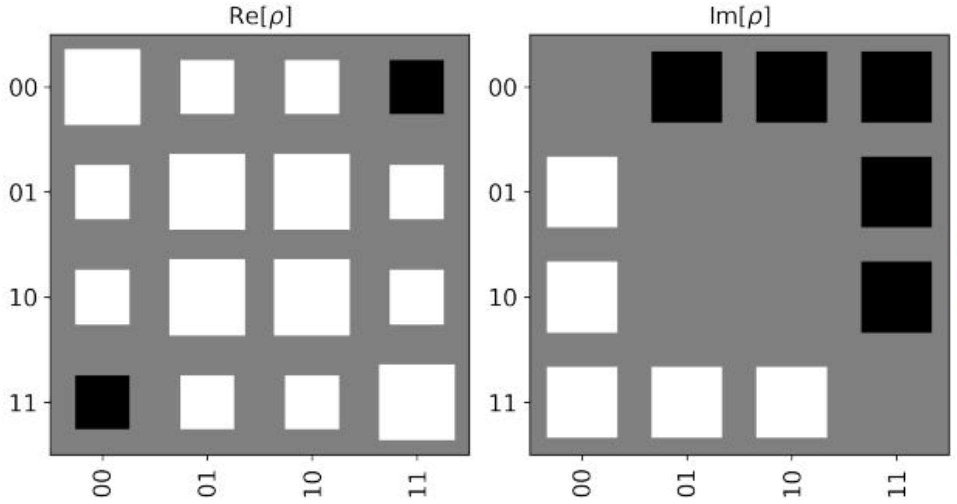
- Amplitude Encoding involves mapping classical data onto qubits through a series of controlled rotations to define the amplitudes of the quantum state.
- This technique can handle non-linear data distributions and is helpful for various tasks, including classification, regression, and clustering.



**Algorithm 1** Amplitude Encoding

```
input a $p$-dimensional classical data point $x \in \mathbb{R}^p$, a number $n$ of qubits, and a set of vectors $v_i$ representing the classical samples
pad the input vector $x$ with zero features as necessary to create a vector of dimension $2n$
normalise the input vector $x$ to ensure that its amplitudes lie in the range $[0, 1]$
initialise the quantum state $|0\rangle^n$, where each qubit is in the state $|0\rangle$
for each qubit $q_s$ do
    for each classical sample $i$ do
        calculate the rotation angle $\beta_i$ based on the amplitudes of the original state and the associated vector $v_i$
        apply the rotation gate $R(v_i, \beta_i)$ on qubit $q_s$
    end for
end for
return encoded states $|\psi\rangle$
```

## Example: A

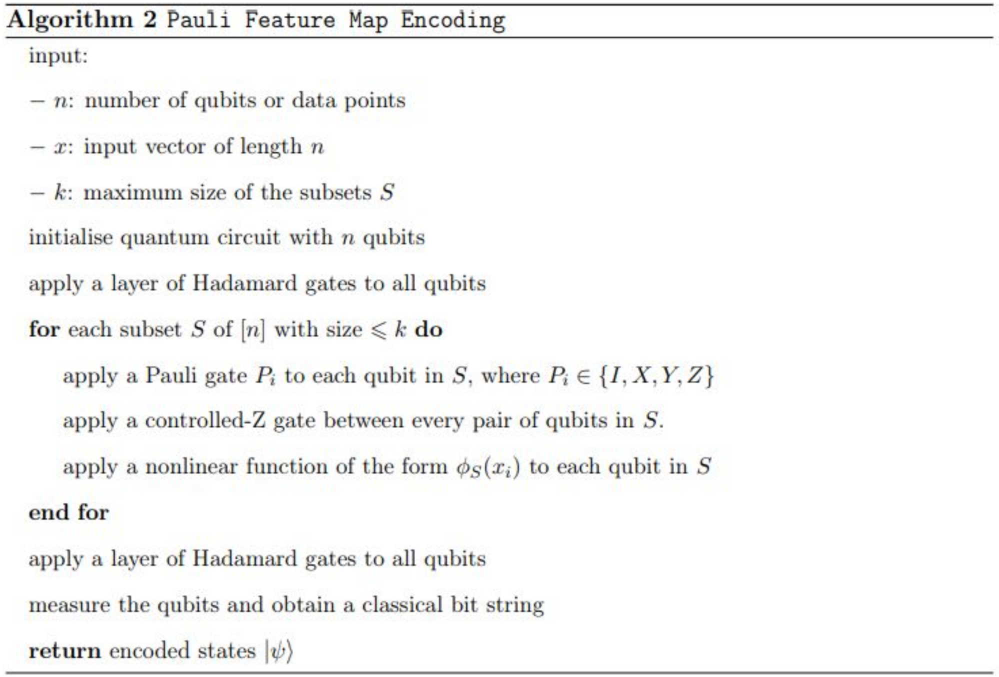


$$\begin{pmatrix} 0.25 - 4.33012702 \times 10^{-01} \jmath & 0.5 - 7.85046229 \times 10^{-17} \jmath \\ 0.5 - 8.32667268 \times 10^{-17} \jmath & 0.25 + 4.33012702 \times 10^{-01} \jmath \end{pmatrix}$$

*Hinton representation*

## Pauli Feature Map Encoding

- The Pauli feature map involves applying repeated sequences of Hadamard and Pauli gates to an initially prepared quantum system, with the Pauli gates acting as a set of nonlinear functions of the input data.
- There are several variations of the Pauli feature map, such as the ZZFeatureMap and IQPEmbedding, which can provide different trade-offs between expressiveness and efficiency for different applications.

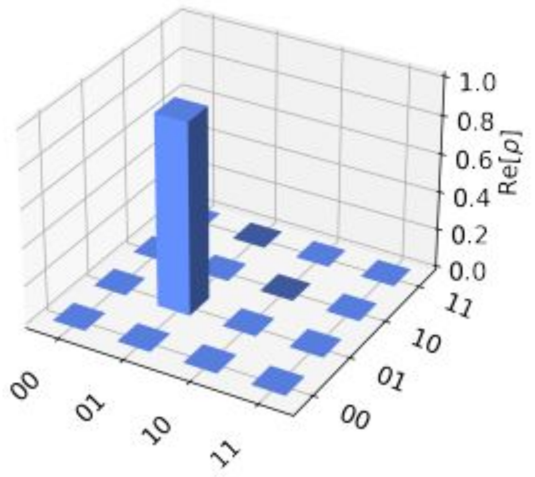


**Algorithm 2** Pauli Feature Map Encoding

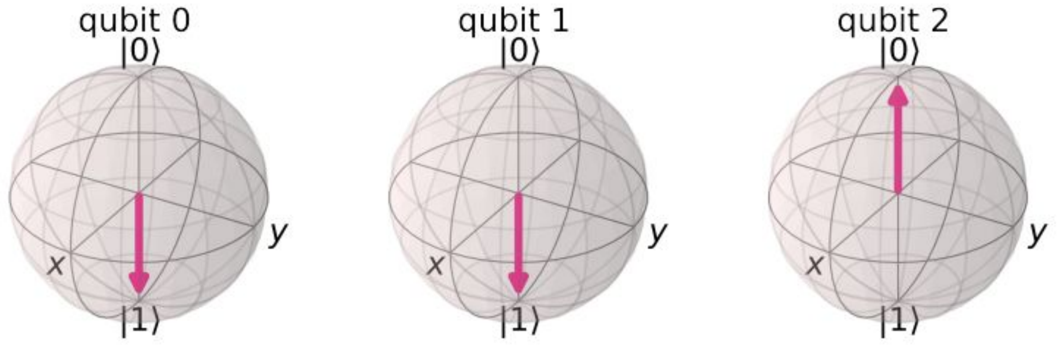
```
input:
– $n$: number of qubits or data points
– $x$: input vector of length $n$
– $k$: maximum size of the subsets $S$
initialise quantum circuit with $n$ qubits
apply a layer of Hadamard gates to all qubits
for each subset $S$ of $[n]$ with size $\leqslant k$ do
    apply a Pauli gate $P_i$ to each qubit in $S$, where $P_i \in \{I, X, Y, Z\}$
    apply a controlled-Z gate between every pair of qubits in $S$.
    apply a nonlinear function of the form $\phi_S(x_i)$ to each qubit in $S$
end for
apply a layer of Hadamard gates to all qubits
measure the qubits and obtain a classical bit string
return encoded states $|\psi\rangle$
```

## Example:

- AG Using ZZFeature Map
- AGC Using IQPEmbedding



$$\langle 0. + 0.\jmath, 1. + 0.\jmath, -0. + 0.\jmath, -0. + 0.\jmath \rangle$$



$$\begin{pmatrix} 1.83697020 \times 10^{-16} + 0\jmath & -2.24963967 \times 10^{-32} + 0\jmath & -1.22464680 \times 10^{-16} + 0\jmath \\ 1.00000000 + 0\jmath & 0.00000000 + 0\jmath & 0.00000000 + 0\jmath \\ 0.00000000 + 0\jmath & 0.00000000 + 0\jmath & 0.00000000 + 0\jmath \\ 0.00000000 + 0\jmath & 0.00000000 + 0\jmath & 0.00000000 + 0\jmath \end{pmatrix}$$

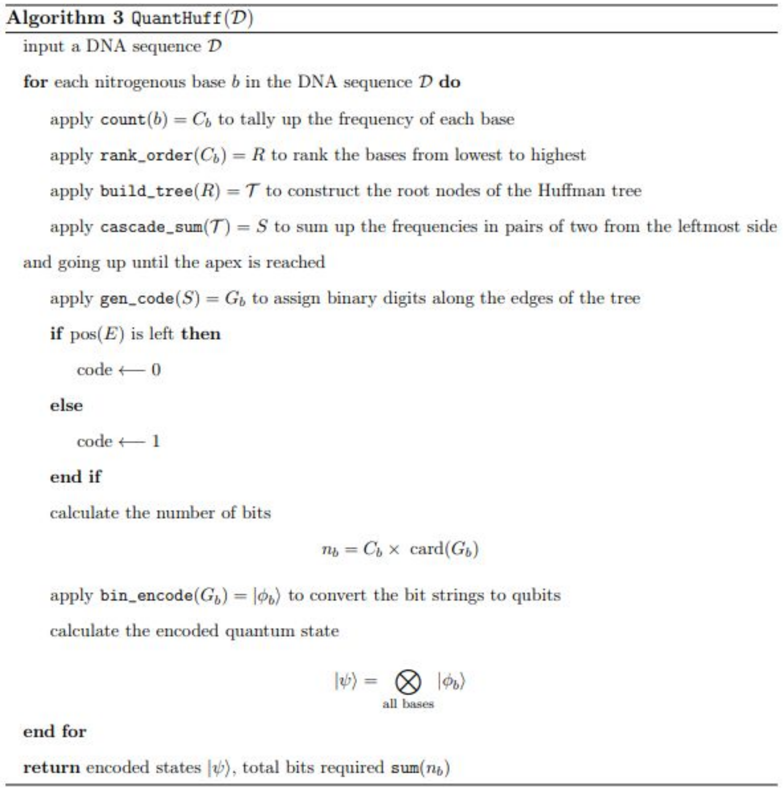
## Lossless Compression-Inspired Encoding Schemes

## Motivation for Design of Lossless Compression Algorithms

- They preserve the original DNA sequence without any information loss. It is important to note that even if one pyrimidine (nitrogenous) base is off, this affects the entire DNA sequence and, thus, the organism as well.
- Allows for the representation of more frequently occurring bases with shorter code.
- Repeated sequences can be encoded to one symbol, thus, reducing the number of qubits required. This is especially important in the NISQ-era, whereby we have limited qubits to work with. This facilitates scalability because real-world Genomic data is quite large.
- Lossless-based compression algorithms are quick to implement and efficient.
- Since lossless-based compression algorithms offer the benefit of reducing the amount of data for preprocessing, these algorithms can potentially reduce the number of errors sustained as current QC is highly error sensitive.

# Quantum-influenced Huffman Coding Stratagem

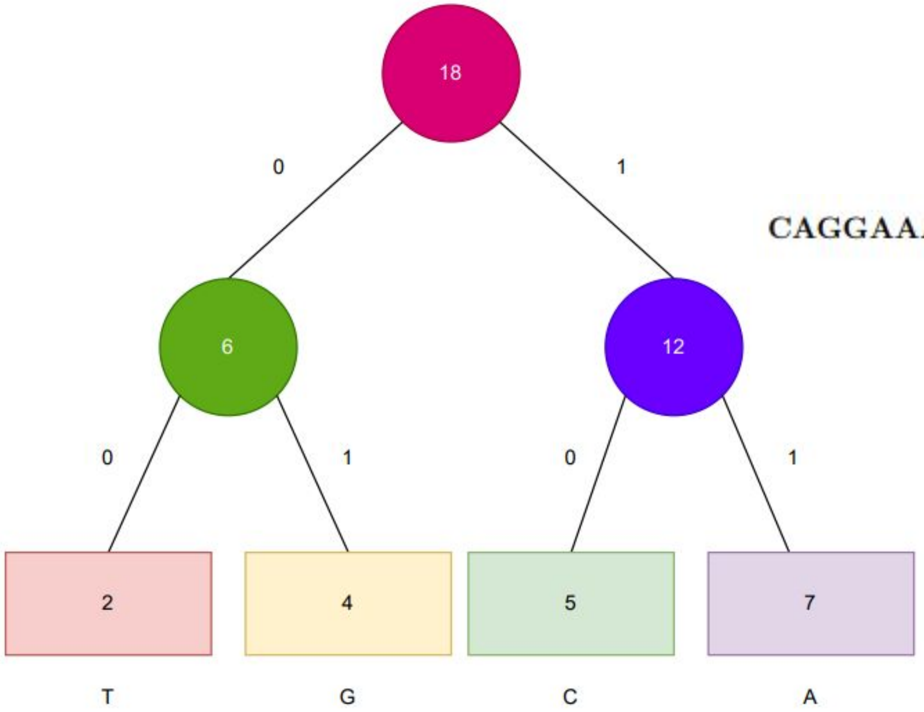
- Huffman coding is a data compression algorithm that assigns varying-length codes to symbols based on their frequency of occurrence in the input data.
- In DNA encoding applications, Huffman coding can represent DNA sequences with a minimal number of qubits, which is particularly useful in quantum computing applications.



**Algorithm 3** QuantHuff($\mathcal{D}$)

```
input a DNA sequence $\mathcal{D}$
for each nitrogenous base $b$ in the DNA sequence $\mathcal{D}$ do
    apply count($b$) $= C_b$ to tally up the frequency of each base
    apply rank_order($C_b$) $= R$ to rank the bases from lowest to highest
    apply build_tree($R$) $= \mathcal{T}$ to construct the root nodes of the Huffman tree
    apply cascade_sum($\mathcal{T}$) $= S$ to sum up the frequencies in pairs of two from the leftmost side
and going up until the apex is reached
    apply gen_code($S$) $= G_b$ to assign binary digits along the edges of the tree
    if pos($E$) is left then
        code $\longleftarrow 0$
    else
        code $\longleftarrow 1$
    end if
    calculate the number of bits
                    $n_b = C_b \times \text{card}(G_b)$
    apply bin_encode($G_b$) $= |\phi_b\rangle$ to convert the bit strings to qubits
    calculate the encoded quantum state
                    $|\psi\rangle = \bigotimes_{\text{all bases}} |\phi_b\rangle$
end for
return encoded states $|\psi\rangle$, total bits required sum($n_b$)
```

## Example: CAGGAAACAGCTATGACC



$$\begin{aligned}\text{CAGGAAACAGCTATGACC} \longrightarrow |\psi\rangle = {} & |10\rangle \otimes |11\rangle \otimes |01\rangle^{\otimes 2} \otimes |11\rangle^{\otimes 3} \otimes |10\rangle \\ & \otimes |11\rangle \otimes |01\rangle \otimes |10\rangle \otimes |00\rangle \otimes |11\rangle \\ & \otimes |00\rangle \otimes |01\rangle \otimes |11\rangle \otimes |10\rangle^{\otimes 2}. \qquad (10)\end{aligned}$$

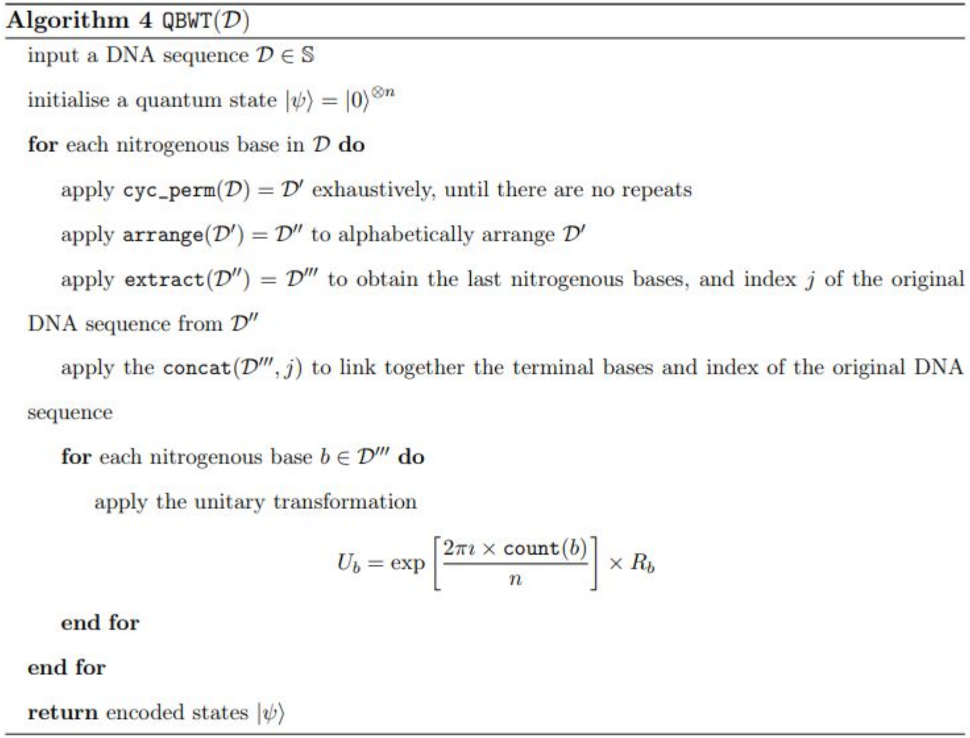
*Graphical representation of the nitrogenous bases with their counts and codes*

## Algorithm Deepdive

- We observe that the algorithm has a time computational complexity of O(k) steps. This means that despite all the seemingly difficult operations, it scales linearly in time as the size of the input increases!
- Additionally, the algorithm calculates and returns the number of qubits required to encode the DNA sequence.

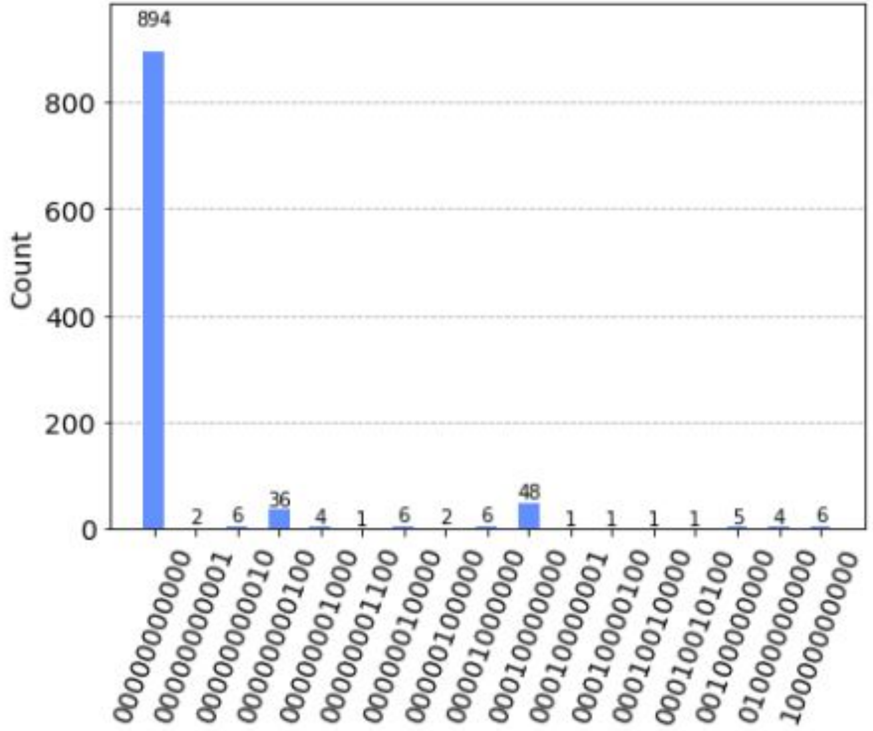
# Quantum-influenced Burrows-Wheeler Transform (QBWT)

- The QBWT uses quantum circuits to perform the matrix multiplication step in the BWT algorithm, which is typically the most time-consuming part of the computation.
- Like the classical BWT, the QBWT can be used to compress DNA sequences and exploit the redundancy in the sequence for storage and analysis purposes.



```
Algorithm 4 QBWT($\mathcal{D}$)
input a DNA sequence $\mathcal{D} \in \mathbb{S}$
initialise a quantum state $|\psi\rangle = |0\rangle^{\otimes n}$
for each nitrogenous base in $\mathcal{D}$ do
    apply cyc_perm($\mathcal{D}$) $= \mathcal{D}'$ exhaustively, until there are no repeats
    apply arrange($\mathcal{D}'$) $= \mathcal{D}''$ to alphabetically arrange $\mathcal{D}'$
    apply extract($\mathcal{D}''$) $= \mathcal{D}'''$ to obtain the last nitrogenous bases, and index $j$ of the original
DNA sequence from $\mathcal{D}''$
    apply the concat($\mathcal{D}''', j$) to link together the terminal bases and index of the original DNA
sequence
    for each nitrogenous base $b \in \mathcal{D}'''$ do
        apply the unitary transformation
            $U_b = \exp\left[\frac{2\pi\imath \times \mathtt{count}(b)}{n}\right] \times R_b$
    end for
end for
return encoded states $|\psi\rangle$
```

## Example: CTTTTGGAAAA

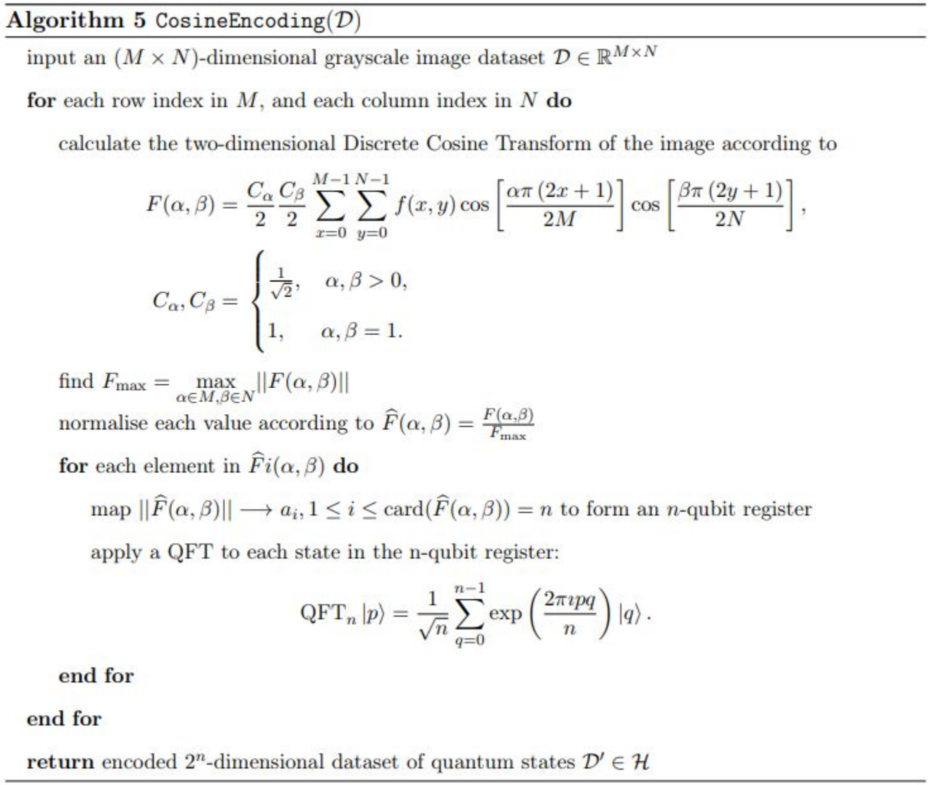


## Algorithm Deepdive

- The QBWT has a time computational complexity of O(k^2), i.e., the algorithm scales quadratically as the input size increases.
- This step is not included in the algorithm, but we could add some kind of special character, say ◆, at the end of the DNA sequence and then start cycling through the DNA sequence exhaustively until we reach the original sequence, not included. For example, ABC. We add ◆ at the end to get: ABC◆. Now, we cycle through to get: ◆ABC, A◆BC, AB◆C.
- The algorithm then alphabetically orders the sequence. In the above example, coincidently, it is ordered already.
- The algorithm then extracts the last column of the alphabetically-sorted sequence and the index from the original DNA sequence.
- Lastly, the algorithm applies a unitary transformation, U, to encode and return the quantum state.
- It is important to note that the unitary transformation given here is arbitrary, and one may choose any suitable unitary transformation.

## Cosine Encoding

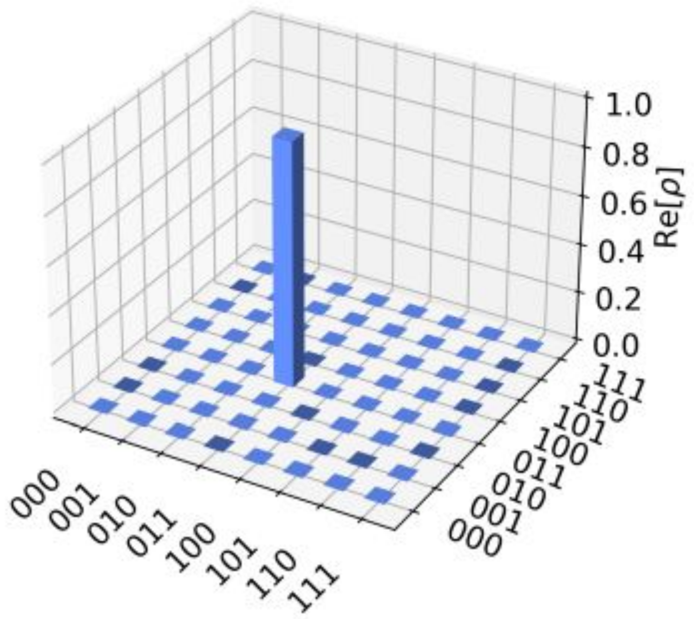
- The wavelet-inspired encoding scheme is designed to handle DNA sequence image data, which is a departure from the other schemes that attempt to encode the DNA sequences directly.
- This encoding scheme can also be used for data compression, processing, and analysis of DNA sequence images.



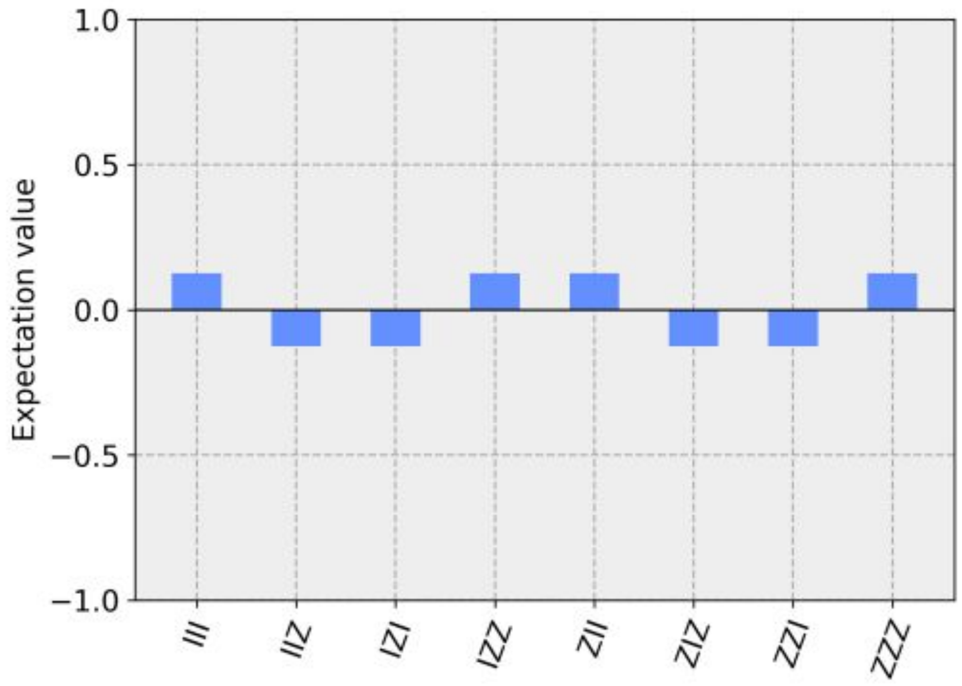
**Algorithm 5** CosineEncoding($\mathcal{D}$)

```
input an $(M \times N)$-dimensional grayscale image dataset $\mathcal{D} \in \mathbb{R}^{M \times N}$
for each row index in $M$, and each column index in $N$ do
    calculate the two-dimensional Discrete Cosine Transform of the image according to
        $F(\alpha,\beta) = \frac{C_\alpha}{2}\frac{C_\beta}{2}\sum_{x=0}^{M-1}\sum_{y=0}^{N-1} f(x,y)\cos\left[\frac{\alpha\pi(2x+1)}{2M}\right]\cos\left[\frac{\beta\pi(2y+1)}{2N}\right],$
        $C_\alpha, C_\beta = \begin{cases} \frac{1}{\sqrt{2}}, & \alpha,\beta > 0, \\ 1, & \alpha,\beta = 1. \end{cases}$
    find $F_{\max} = \max_{\alpha \in M, \beta \in N} ||F(\alpha,\beta)||$
    normalise each value according to $\widehat{F}(\alpha,\beta) = \frac{F(\alpha,\beta)}{F_{\max}}$
    for each element in $\widehat{F}i(\alpha,\beta)$ do
        map $||\widehat{F}(\alpha,\beta)|| \longrightarrow a_i, 1 \le i \le \mathrm{card}(\widehat{F}(\alpha,\beta)) = n$ to form an $n$-qubit register
        apply a QFT to each state in the n-qubit register:
            $\mathrm{QFT}_n |p\rangle = \frac{1}{\sqrt{n}}\sum_{q=0}^{n-1}\exp\left(\frac{2\pi\imath pq}{n}\right)|q\rangle .$
    end for
end for
return encoded $2^n$-dimensional dataset of quantum states $\mathcal{D}' \in \mathcal{H}$
```

## Example: ATC



*City representation*



*Pauli representation*

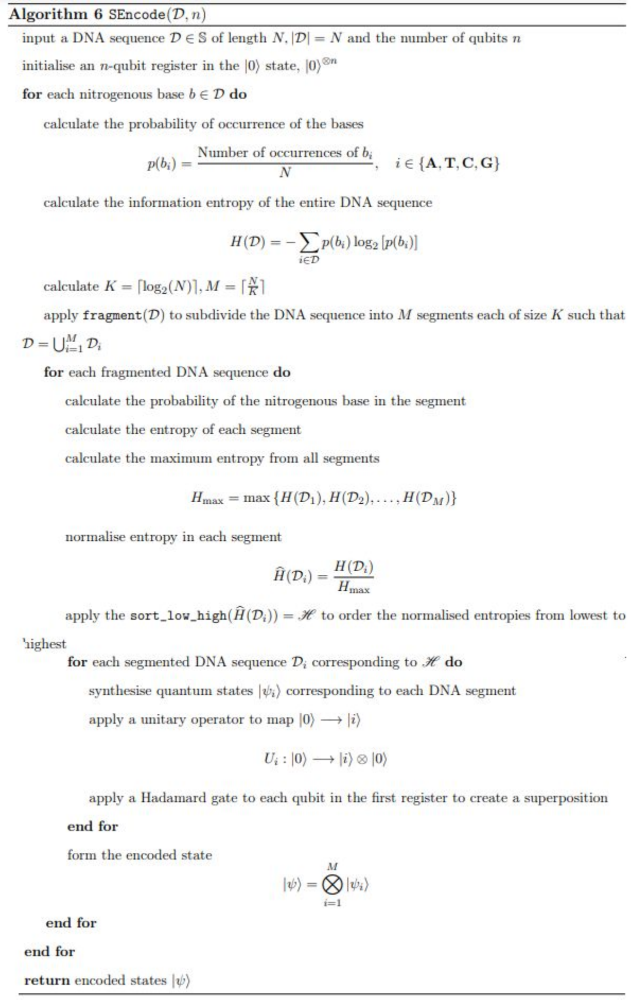
## Entropy-Based Methods

## Motivation for Entropy-based Algorithms

- In general, entropy-based methods, inspired by ML, extract the most relevant information from high-dimensional datasets and extract the most important features and relevant information efficiently.
- Entropy-based algorithms can identify patterns and relationships in the data that can provide insights into the underlying sequence; this can help us better understand the genomic data and make more informed decisions about experimental design and data analysis.
- They allow for checking information loss during encoding or how much of the original information has been retained, i.e., information gain.

## SEncode

- The SEncode is based on the theory of information and uses the fixed-length DNA subsequence with the lowest entropy to create highly entangled quantum states that encode the information in the DNA sequence.
- DNA sequences are often assumed to have a high degree of inherent randomness, making entropy-based methods a natural choice for comparing and encoding DNA sequences.

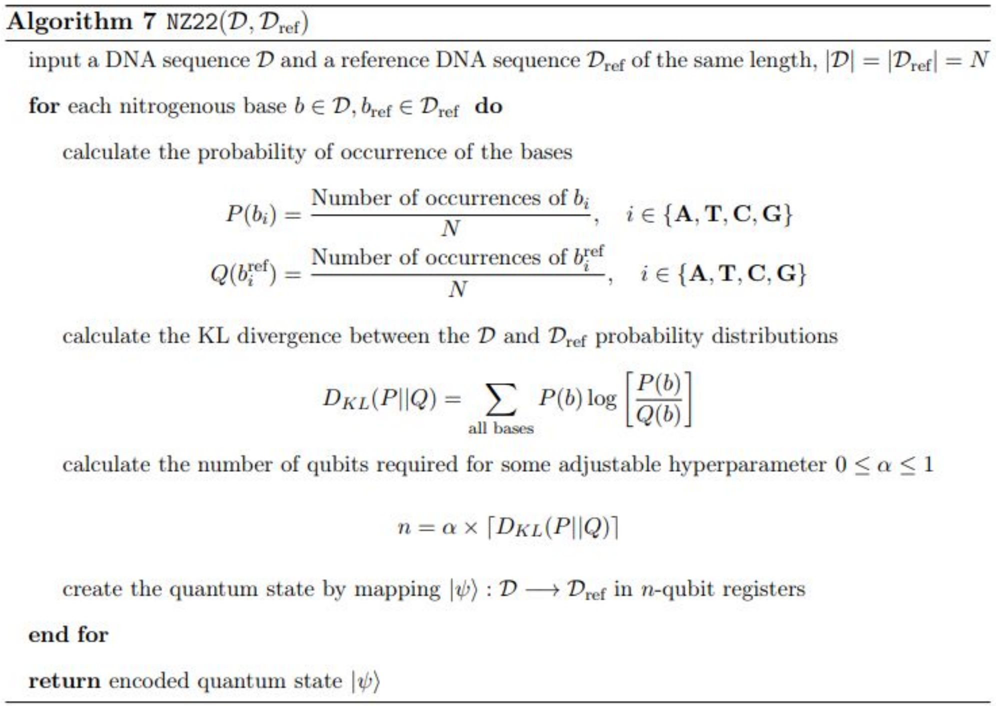


**Algorithm 6** SEncode($\mathcal{D}, n$)

```
input a DNA sequence $\mathcal{D} \in \mathbb{S}$ of length $N, |\mathcal{D}| = N$ and the number of qubits $n$
initialise an $n$-qubit register in the $|0\rangle$ state, $|0\rangle^{\otimes n}$
for each nitrogenous base $b \in \mathcal{D}$ do
    calculate the probability of occurrence of the bases
        $$p(b_i) = \frac{\text{Number of occurrences of } b_i}{N}, \quad i \in \{\mathbf{A}, \mathbf{T}, \mathbf{C}, \mathbf{G}\}$$
    calculate the information entropy of the entire DNA sequence
        $$H(\mathcal{D}) = -\sum_{i \in \mathcal{D}} p(b_i) \log_2 [p(b_i)]$$
    calculate $K = \lceil \log_2(N) \rceil, M = \lceil \frac{N}{K} \rceil$
    apply fragment($\mathcal{D}$) to subdivide the DNA sequence into $M$ segments each of size $K$ such that
$\mathcal{D} = \bigcup_{i=1}^{M} \mathcal{D}_i$
    for each fragmented DNA sequence do
        calculate the probability of the nitrogenous base in the segment
        calculate the entropy of each segment
        calculate the maximum entropy from all segments
            $$H_{\max} = \max \{H(\mathcal{D}_1), H(\mathcal{D}_2), \dots, H(\mathcal{D}_M)\}$$
        normalise entropy in each segment
            $$\widehat{H}(\mathcal{D}_i) = \frac{H(\mathcal{D}_i)}{H_{\max}}$$
        apply the sort_low_high($\widehat{H}(\mathcal{D}_i)$) $= \mathscr{H}$ to order the normalised entropies from lowest to
highest
        for each segmented DNA sequence $\mathcal{D}_i$ corresponding to $\mathscr{H}$ do
            synthesise quantum states $|\psi_i\rangle$ corresponding to each DNA segment
            apply a unitary operator to map $|0\rangle \longrightarrow |i\rangle$
                $$U_i : |0\rangle \longrightarrow |i\rangle \otimes |0\rangle$$
            apply a Hadamard gate to each qubit in the first register to create a superposition
        end for
        form the encoded state
            $$|\psi\rangle = \bigotimes_{i=1}^{M} |\psi_i\rangle$$
    end for
end for
return encoded states $|\psi\rangle$
```

## NZ22

- The NZ22 is based on quantum data compression, which involves encoding a large amount of classical data into a smaller quantum state.
- This algorithm uses KL divergence to determine the number of qubits needed to encode the input sequence into the reference sequence.

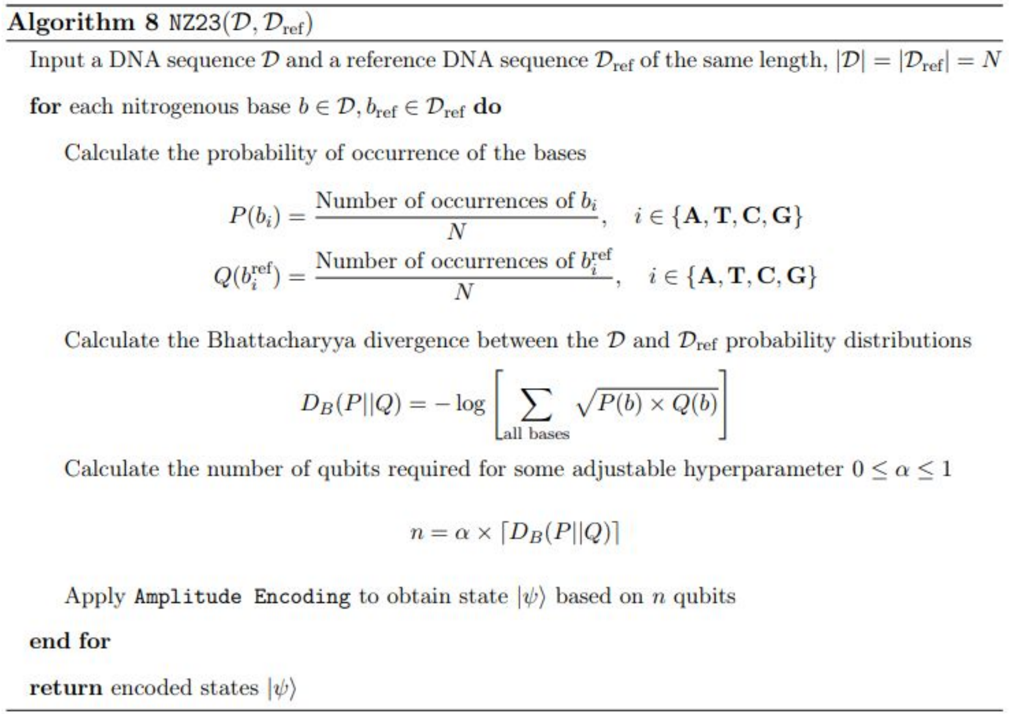


**Algorithm 7** NZ22($\mathcal{D}, \mathcal{D}_{\text{ref}}$)

```
input a DNA sequence $\mathcal{D}$ and a reference DNA sequence $\mathcal{D}_{\text{ref}}$ of the same length, $|\mathcal{D}| = |\mathcal{D}_{\text{ref}}| = N$
for each nitrogenous base $b \in \mathcal{D}, b_{\text{ref}} \in \mathcal{D}_{\text{ref}}$ do
    calculate the probability of occurrence of the bases
        $P(b_i) = \frac{\text{Number of occurrences of } b_i}{N}, \quad i \in \{\mathbf{A}, \mathbf{T}, \mathbf{C}, \mathbf{G}\}$
        $Q(b_i^{\text{ref}}) = \frac{\text{Number of occurrences of } b_i^{\text{ref}}}{N}, \quad i \in \{\mathbf{A}, \mathbf{T}, \mathbf{C}, \mathbf{G}\}$
    calculate the KL divergence between the $\mathcal{D}$ and $\mathcal{D}_{\text{ref}}$ probability distributions
        $D_{KL}(P||Q) = \sum_{\text{all bases}} P(b) \log \left[\frac{P(b)}{Q(b)}\right]$
    calculate the number of qubits required for some adjustable hyperparameter $0 \leq \alpha \leq 1$
        $n = \alpha \times \lceil D_{KL}(P||Q) \rceil$
    create the quantum state by mapping $|\psi\rangle : \mathcal{D} \longrightarrow \mathcal{D}_{\text{ref}}$ in $n$-qubit registers
end for
return encoded quantum state $|\psi\rangle$
```

## NZ23

- The NZ23 computes the Bhattacharyya divergence to determine the number of qubits required to encode the quantum state.
- The adjustable hyperparameter α allows us to balance the tradeoff between the accuracy of the encoding and the number of qubits required.

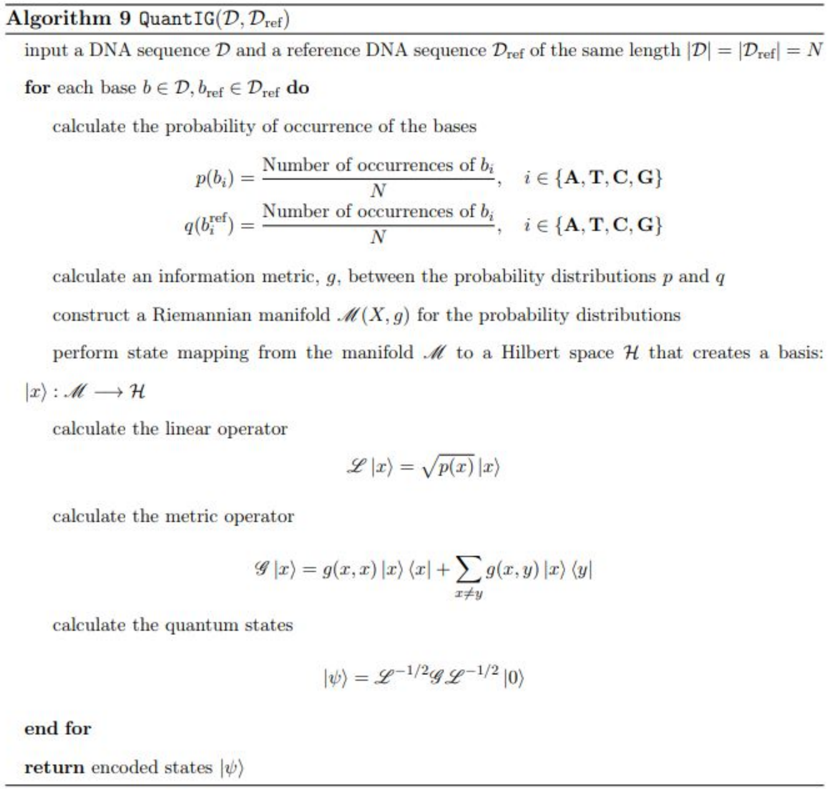


**Algorithm 8** NZ23($\mathcal{D}, \mathcal{D}_{\text{ref}}$)

```
Input a DNA sequence $\mathcal{D}$ and a reference DNA sequence $\mathcal{D}_{\text{ref}}$ of the same length, $|\mathcal{D}| = |\mathcal{D}_{\text{ref}}| = N$
for each nitrogenous base $b \in \mathcal{D}, b_{\text{ref}} \in \mathcal{D}_{\text{ref}}$ do
    Calculate the probability of occurrence of the bases
        $P(b_i) = \frac{\text{Number of occurrences of } b_i}{N}, \quad i \in \{\mathbf{A}, \mathbf{T}, \mathbf{C}, \mathbf{G}\}$
        $Q(b_i^{\text{ref}}) = \frac{\text{Number of occurrences of } b_i^{\text{ref}}}{N}, \quad i \in \{\mathbf{A}, \mathbf{T}, \mathbf{C}, \mathbf{G}\}$
    Calculate the Bhattacharyya divergence between the $\mathcal{D}$ and $\mathcal{D}_{\text{ref}}$ probability distributions
        $D_B(P||Q) = -\log \left[ \sum_{\text{all bases}} \sqrt{P(b) \times Q(b)} \right]$
    Calculate the number of qubits required for some adjustable hyperparameter $0 \leq \alpha \leq 1$
        $n = \alpha \times \lceil D_B(P||Q) \rceil$
    Apply Amplitude Encoding to obtain state $|\psi\rangle$ based on $n$ qubits
end for
return encoded states $|\psi\rangle$
```

## QuantIG

- QuantIG is a novel approach combining Differential Geometry and Information Theory concepts to analyze DNA sequences.
- This algorithm constructs a Riemannian manifold to represent the probability distributions of DNA bases and calculates quantum states using linear and metric operators.



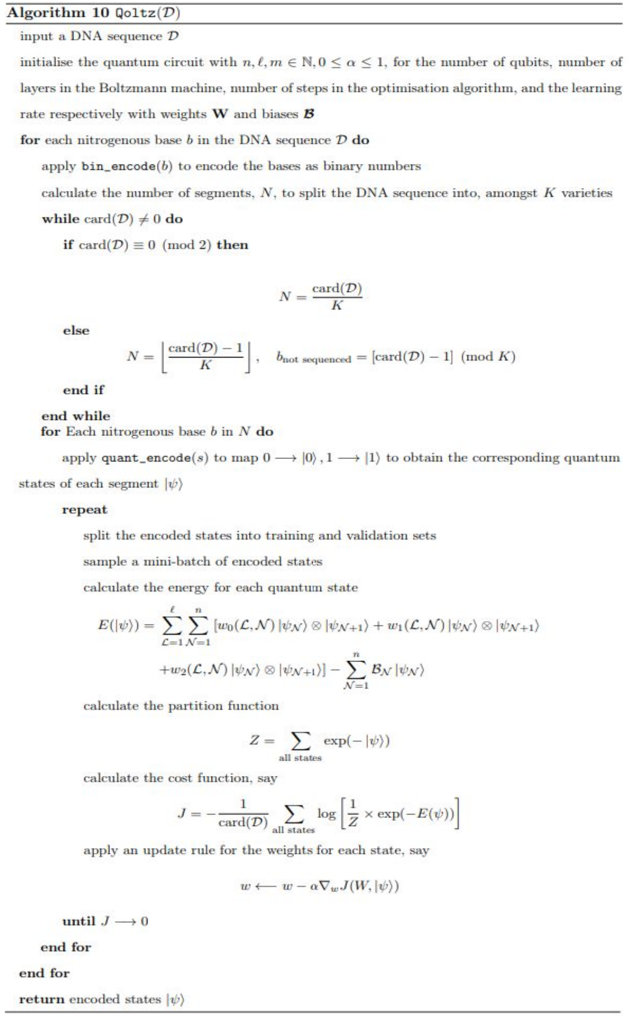
**Algorithm 9** QuantIG($\mathcal{D}, \mathcal{D}_{\text{ref}}$)

```
input a DNA sequence $\mathcal{D}$ and a reference DNA sequence $\mathcal{D}_{\text{ref}}$ of the same length $|\mathcal{D}| = |\mathcal{D}_{\text{ref}}| = N$
for each base $b \in \mathcal{D}, b_{\text{ref}} \in \mathcal{D}_{\text{ref}}$ do
    calculate the probability of occurrence of the bases
        $p(b_i) = \frac{\text{Number of occurrences of } b_i}{N}, \quad i \in \{\mathbf{A}, \mathbf{T}, \mathbf{C}, \mathbf{G}\}$
        $q(b_i^{\text{ref}}) = \frac{\text{Number of occurrences of } b_i}{N}, \quad i \in \{\mathbf{A}, \mathbf{T}, \mathbf{C}, \mathbf{G}\}$
    calculate an information metric, $g$, between the probability distributions $p$ and $q$
    construct a Riemannian manifold $\mathscr{M}(X, g)$ for the probability distributions
    perform state mapping from the manifold $\mathscr{M}$ to a Hilbert space $\mathcal{H}$ that creates a basis:
$|x\rangle : \mathscr{M} \longrightarrow \mathcal{H}$
    calculate the linear operator
        $\mathscr{L}|x\rangle = \sqrt{p(x)}|x\rangle$
    calculate the metric operator
        $\mathscr{G}|x\rangle = g(x,x)|x\rangle\langle x| + \sum_{x \neq y} g(x,y)|x\rangle\langle y|$
    calculate the quantum states
        $|\psi\rangle = \mathscr{L}^{-1/2}\mathscr{G}\mathscr{L}^{-1/2}|0\rangle$
end for
return encoded states $|\psi\rangle$
```

## Energy-Based Methods For Testing Encoded Sequences

## Qoltz

- The QBM can implement the probabilistic updates that drive the learning process, and the number of qubits and gates used will depend on the size and complexity of the input DNA sequence.
- QBMs have the potential to provide a quantum advantage for certain types of ML tasks, including DNA sequencing, due to their ability to perform computations in parallel and explore large search spaces more efficiently than classical methods.



**Algorithm 10** Qoltz($\mathcal{D}$)

```
input a DNA sequence $\mathcal{D}$
initialise the quantum circuit with $n, \ell, m \in \mathbb{N}, 0 \leq \alpha \leq 1$, for the number of qubits, number of layers in the Boltzmann machine, number of steps in the optimisation algorithm, and the learning rate respectively with weights $\mathbf{W}$ and biases $\boldsymbol{\mathcal{B}}$
for each nitrogenous base $b$ in the DNA sequence $\mathcal{D}$ do
  apply bin_encode($b$) to encode the bases as binary numbers
  calculate the number of segments, $N$, to split the DNA sequence into, amongst $K$ varieties
  while $\mathrm{card}(\mathcal{D}) \neq 0$ do
    if $\mathrm{card}(\mathcal{D}) \equiv 0 \pmod 2$ then
        $N = \frac{\mathrm{card}(\mathcal{D})}{K}$
    else
        $N = \left\lfloor \frac{\mathrm{card}(\mathcal{D}) - 1}{K} \right\rfloor, \quad b_{\text{not sequenced}} = [\mathrm{card}(\mathcal{D}) - 1] \pmod K$
    end if
  end while
  for Each nitrogenous base $b$ in $N$ do
    apply quant_encode($s$) to map $0 \longrightarrow |0\rangle, 1 \longrightarrow |1\rangle$ to obtain the corresponding quantum states of each segment $|\psi\rangle$
    repeat
      split the encoded states into training and validation sets
      sample a mini-batch of encoded states
      calculate the energy for each quantum state
        $E(|\psi\rangle) = \sum_{\mathcal{L}=1}^{\ell} \sum_{\mathcal{N}=1}^{n} [w_0(\mathcal{L},\mathcal{N}) |\psi_{\mathcal{N}}\rangle \otimes |\psi_{\mathcal{N}+1}\rangle + w_1(\mathcal{L},\mathcal{N}) |\psi_{\mathcal{N}}\rangle \otimes |\psi_{\mathcal{N}+1}\rangle + w_2(\mathcal{L},\mathcal{N}) |\psi_{\mathcal{N}}\rangle \otimes |\psi_{\mathcal{N}+1}\rangle] - \sum_{\mathcal{N}=1}^{n} \mathcal{B}_{\mathcal{N}} |\psi_{\mathcal{N}}\rangle$
      calculate the partition function
        $Z = \sum_{\text{all states}} \exp(-|\psi\rangle)$
      calculate the cost function, say
        $J = -\frac{1}{\mathrm{card}(\mathcal{D})} \sum_{\text{all states}} \log \left[ \frac{1}{Z} \times \exp(-E(\psi)) \right]$
      apply an update rule for the weights for each state, say
        $w \longleftarrow w - \alpha \nabla_w J(W, |\psi\rangle)$
    until $J \longrightarrow 0$
  end for
end for
return encoded states $|\psi\rangle$
```

## Conclusion & Future Work

- We have developed new algorithms for classical-to-quantum data encoding in bioinformatics by employing various quantum computing techniques.
- We have demonstrated the efficacy of proposed algorithms, such as QuantHuff, QBWT, CosineEncoding, SEncode, and Qoltz, in producing encoded quantum states.
- Further testing and validation are needed to assess effectiveness in larger datasets and real-world scenarios.
- The study provides promising avenues for the development of bioinformatics applications based on quantum computing.

★ Investigate the performance of proposed algorithms on larger datasets and compare them with existing methods.

★ Test algorithms on real quantum hardware to evaluate the performance and effectiveness in practical applications.

★ Explore the potential of quantum computing for other bioinformatics tasks, such as protein folding prediction and drug discovery, using our algorithms.

★ Examine the limitations and challenges of the proposed methods, such as the impact of noise on encoding and algorithm scalability.

★ Contribute to the advancement of bioinformatics applications of quantum computing.

## Thank you!

Nouhaila Innan & Dr. Muhammed Al-Zafar Khan
nouhailainnan@gmail.com, muhammadalzafark@gmail.com