Information Models for Prediction
(Assignment 2 - TAI)
https://github.com/Miragaia/PROJECT2_TAI

Miguel Miragaia
Algorithmic Theory of Information
Student 108317, DETI
Universidade de Aveiro, Portugal
miguelmiragaia@ua.pt

Miguel Cruzeiro
Algorithmic Theory of Information
Student 107660, DETI
Universidade de Aveiro, Portugal
miguelcruzeiro@ua.pt

Diogo Silva
Algorithmic Theory of Information
Student 107647, DETI
Universidade de Aveiro, Portugal
diogo.manuel@ua.pt

April 11, 2025

1 Introduction

This report presents the development and analysis of two C++ programs: `FCM` and `MetaClass`, designed to perform similarity analysis on DNA sequences using Finite Context Models (FCM) and the Normalized Relative Compression (NRC) metric.

The `FCM` program implements a probabilistic model that learns context-based symbol distributions from an input sequence. It supports multiple functionalities:

- Training an FCM model based on user-defined parameters such as context length (k) and smoothing factor (α).
- Computing the number of bits required to encode a sequence given a trained model.
- Estimating the Normalized Relative Compression (NRC) between two sequences to measure their similarity.
- Producing a complexity profile for a given sequence, indicating the information content of each symbol relative to its context.

The `MetaClass` program extends this functionality to the domain of metagenomics. It allows the comparison of a metagenomic DNA sample against a reference database of sequences. The main goal is to identify the most similar sequences to the sample using NRC as a similarity metric. The workflow includes training an FCM on the sample, evaluating the compression cost of encoding each reference sequence with the sample model, and ranking them based on the resulting NRC scores.

This approach leverages the idea that sequences more similar to the sample will be better predicted—and thus more efficiently compressed—by the model trained on the sample. As such, NRC provides an alignment-free, information-theoretic method for sequence comparison.

The objective of this project is to explore the application of FCMs and NRC in the context of DNA sequence analysis, with a focus on compression-based similarity. This report details the implementation strategies, underlying theory, and the potential of this approach for metagenomic classification and comparative genomics.

2 Implementation

2.1 Developing the FCM Model (`fcm.h`)

2.1.1 Program Input and Output:

- **Input:** The `FCM` class is initialized with two parameters: the context length k (an integer determining the number of preceding symbols used for prediction) and the smoothing factor α (a double controlling Laplace smoothing). It receives a training sequence (e.g., the metagenomic sample or a reference sequence) as a string, typically containing raw DNA data.
- **Output:** The class provides three types of output:
  - **Compression Bits**: The total number of bits required to encode a sequence using the trained model, computed by the `compute_compression_bits` method.
  - **Normalized Relative Compression (NRC)**: A similarity metric between the trained model and a target sequence, computed by the `compute_nrc` method.
  - **Complexity Profile**: A per-symbol vector of information content (in bits) for a sequence, computed by the `compute_complexity_profile` method, useful for local complexity analysis.

2.1.2 Context and Symbol Counts

- **context_counts** is a nested map that stores how many times each symbol appears after a specific context.
- **total_counts** keeps track of the total number of times each context appears in the training data.
- **alphabet** contains all distinct characters seen during training, used to determine the size of the smoothing factor.

Context	Symbol	Count
"GCT"	'A'	4
"GCT"	'G'	2
"ATC"	'T'	5
"ATC"	'G'	3

Table 1: Example of the Context Counts Data Structure

2.1.3 Methods:

The `FCM` class encapsulates all logic required for finite-context modeling. The following methods are implemented:

- **train:** This method builds the context model by scanning the input sequence and populating the `context_counts`, `total_counts`, and `alphabet` based on a user-defined context length k.
- **compute_compression_bits:** This method calculates the total number of bits needed to encode a given sequence using the trained model. Probabilities are estimated using Laplace smoothing, and each symbol's contribution is computed as $-\log_2(P(x_i|context))$.
- **compute_nrc:** Computes the *Normalized Relative Compression (NRC)* between a trained model and a new sequence. It is defined as:

$$\mathcal{NRC}(x^i \| y) = \frac{C(x^i \| y)}{2|x^i|}.$$

  where $C(x^i||y)$ is the number of bits to compress sequence x using a model trained on y.
- **compute_complexity_profile:** Returns a per-symbol complexity vector that shows how much information (in bits) each symbol contributes given its context. This can be used for local complexity analysis and pattern discovery.

2.2 Developing the MetaClass (`MetaClass.cpp`)

This program can analyze metagenomic DNA sequences using a Finite Context Model. The main goal is to compare a given metagenomic sample against a reference database of DNA sequences and identify the most similar sequences based on a metric called Normalized Relative Compression (NRC).

2.2.1 Program Input and Output

The program is executed with several command-line parameters:

- `-d <db_file>`: Path to the reference database file.
- `-s <sample_file>`: Path to the metagenomic sample file.
- `-k <order>`: The order of the Markov model (FCM).
- `-a <alpha>`: Smoothing parameter for FCM probabilities.
- `-t <top_n>`: Number of top similar sequences to display.
- `-c <matrix_output_file>`: Optional file to output an NRC comparison matrix between the top N sequences.
- `-cp`: Generates files wiht the NRC per position for each sequence in the database.

The metagenomic sample is loaded and used to train an FCM model with order k and smoothing parameter α. This model learns the probability distribution of nucleotide contexts within the sample.

For each sequence in the reference database, we evaluate them by computing the corresponding NRC value using the trained model. The NRC indicates how well the sample model can represent the database sequence (lower values indicate better similarity).

The sequences are then sorted by their NRC values, and the top N most similar sequences are printed to the console, showing their NRC values and identifiers.

2.2.2 Generating an NRC Comparison Matrix

An optional feature of the program is the generation of a pairwise **NRC comparison matrix** for the top N most similar sequences identified. If the user provides the `-c <matrix_output_file>` flag, the program will create a CSV file containing NRC values between each pair of the top N sequences.

The logic proceeds as follows:

- The top N sequences are selected based on their NRC scores relative to the sample.
- For each sequence s_i in the top N, a new `FCM` model is trained using s_i as training data.
- The program then evaluates the NRC of every other sequence s_j in the top N using the model trained on s_i, resulting in an asymmetric similarity matrix.
- The result is saved as a CSV file, where each row represents a reference sequence acting as a model and each column shows the NRC score for compressing another sequence using that model.

This allows for comparative analysis among the top candidates, helping to highlight intra-group similarities and structure.

	Seq1	Seq2	Seq3
Seq1	0.000	0.324	0.298
Seq2	0.317	0.000	0.289
Seq3	0.310	0.276	0.000

Table 2: Example of an NRC comparison matrix for top 3 sequences

This matrix can be used for downstream clustering, visualization, or simply to understand how reference sequences relate to each other based on their compression-based similarity.

2.2.3 Generating Complexity Profiles for Top Sequences

An additional feature of the `MetaClass` program is the generation of complexity profiles for the top N sequences identified as most similar to the metagenomic sample.

When the `-cp` flag is provided, the program uses the model previously trained on the metagenomic sample (`meta.txt`) to calculate the information content of each symbol in the sequence.

The complexity for position i is computed as $-\log_2(P(x[i] \mid \text{context}))$ and stored in a vector representing the profile.

The complexity profiles offer a granular view of the information content across each sequence, relative to the FCM trained on the metagenomic sample.

High complexity values indicate symbols that are less predictable given their context, potentially highlighting regions of interest such as mutations, unique patterns, or sequence anomalies.

2.3 Results and Analysis

This section presents the experimental results obtained during testing.

2.3.1 Results

As part of this project, we analyzed the behavior of our `MetaClass` implementation using a provided `meta.txt` file containing a metagenomic DNA sequence, along with a `db.txt` file that included multiple reference DNA sequences.

Listing 1: Running MetaClass with NRC matrix output and complexity profiles

```
./MetaClass -cp -d db.txt -s meta.txt -k 15 -a 0.01 -t 20 -c similarity_matrix.csv
```

```
Top 20 similar sequences based on NRC:
Rank    NRC Value       Sequence ID
----------------------------------------
1       0.042034        OR353425.1 Octopus vulgaris mitochondrion, complete genome
2       0.045196        NC_001348.1 Human herpesvirus 3, complete genome
3       0.046122        NC_005831.2 Human Coronavirus NL63, complete genome
4       0.126456        Super ISS Si1240
5       0.217786        Super MUL 720
6       0.446922        gi|49169782|ref|NC_005831.2| Human Coronavirus NL63, complete genome
7       0.999757        gi|20522139|ref|NC_003810.1| Spinach latent virus putative movement protein and putative coat protein genes, complete cds
8       0.999914        gi|98960844|ref|NC_008037.1| Prune dwarf virus segment RNA2, complete sequence
9       1.000000        ContaXT bug
10      1.000000        gi|1003725824|ref|NC_029621.1| Lake Sarah-associated circular molecule 8 isolate LSaCM-8-LSGA-2013, complete sequence
11      1.000000        gi|1003725858|ref|NC_029626.1| Lake Sarah-associated circular virus-49 isolate LSaCV-49-LSGA-2013, complete sequence
12      1.000000        gi|1003725940|ref|NC_029635.1| Lake Sarah-associated circular virus-46 isolate LSaCV-46-LSSO-2013, complete sequence
13      1.000000        gi|1003725950|ref|NC_029636.1| Lake Sarah-associated circular virus-22 isolate LSaCV-22-LSMU-2013, complete sequence
14      1.000000        gi|1003725973|ref|NC_029638.1| Lake Sarah-associated circular virus-47 isolate LSaCV-47-LSCO-2013, complete sequence
15      1.000000        gi|1006610892|ref|NC_029645.1| Norovirus GIII, complete genome
16      1.000000        gi|19744910|ref|NC_003464.1| Apple mosaic virus RNA 1 putative methyltransferase/helicase gene, complete cds
17      1.000000        gi|19744912|ref|NC_003465.1| Apple mosaic virus RNA 2 putative polymerase gene, complete cds
18      1.000000        gi|19919963|ref|NC_003507.1| Beet soil-borne mosaic virus RNA3 29K protein gene, complete cds
19      1.000000        gi|20428596|ref|NC_003750.1| Rice ragged stunt virus segment S2 P2 gene, complete cds
20      1.000000        gi|20428604|ref|NC_003758.1| Rice ragged stunt virus autocatalytic protease precurser gene, complete cds
```

Figure 1: Top 20 similar sequences based on NRC for the given sample and db. Lower values indicate higher similarity.

Observations

- The top three results have an NRC value below 0.05, indicating a strong similarity between the metagenomic sample (`meta.txt`) and these sequences.
- The **Octopus vulgaris mitochondrion** sequence appears in first place, which means that is the most similar sequence to the one present in the meta file.
- Both **Human herpesvirus 3** and **Human Coronavirus NL63** could mean human contamination on the sample.
- Notably, two sequences related to human coronaviruses appear in the top 20. However, this does not necessarily imply that both sequences are strongly related to the metagenomic sample. This observation will be further explored later sections of the report.
- All values below the NRC threshold of 0.5 are considered irrelevant for identifying significant similarity with the metagenomic sample.

2.3.2 Optimal K and Alpha values

To optimize the performance of the `MetaClass` program in identifying sequences similar to the metagenomic sample, we conducted a parameter tuning experiment by varying the context length k and the smoothing parameter α of the Finite Context Model (FCM). The goal was to determine the parameter combination that minimizes the Normalized Relative Compression (NRC) for the top-ranked sequences, thereby maximizing the model's ability to detect statistical similarities.

Figure 2: Heatmap of mean NRC values for different k and a values

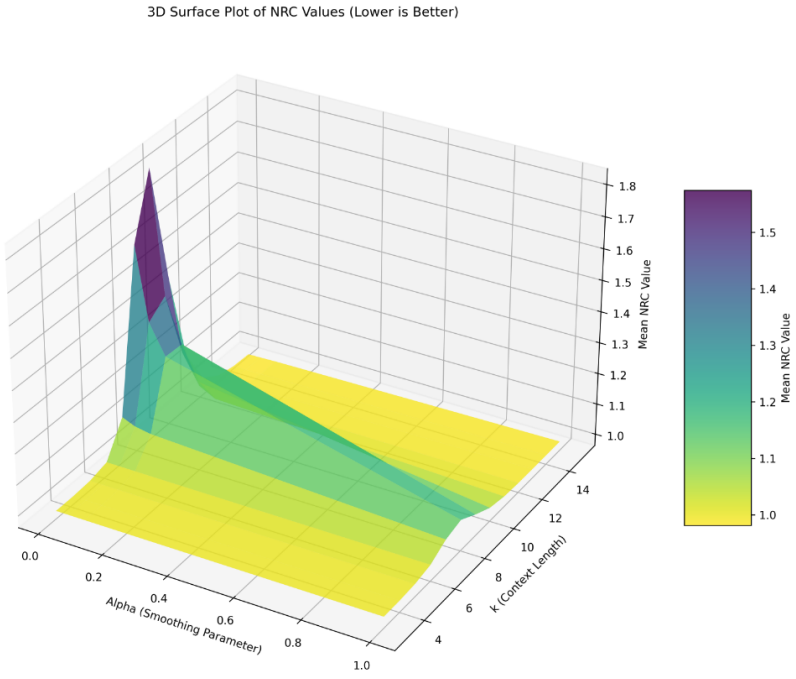


Figure 3: 3D plot of mean NRC values for different k and a values

The results were visualized as a heatmap - Figure 2 - and a 3D plot - Figure 3 - where each cell represents the mean NRC for specific k and α combinations. The optimal parameters were found to be $k = 15$ and $\alpha = 0.01$.

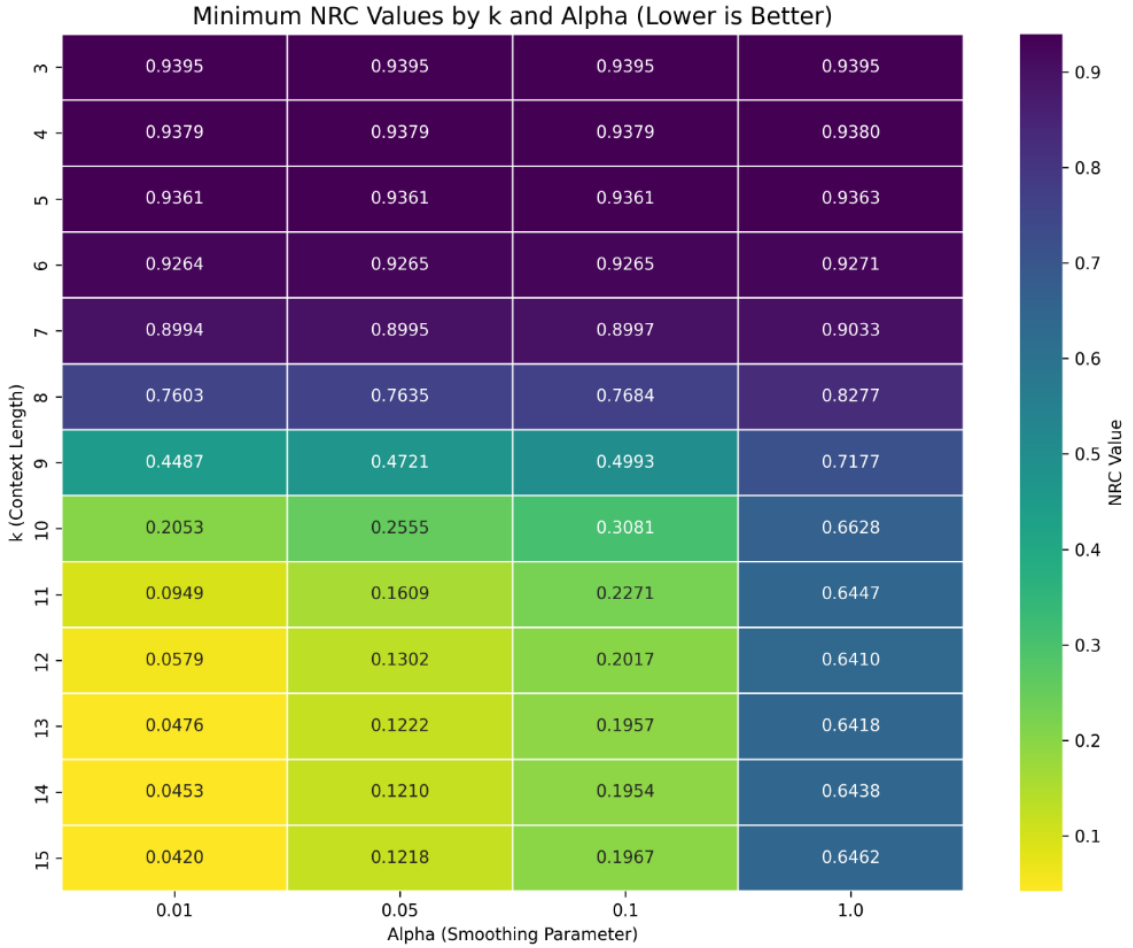


Figure 4: Heatmap of minimum NRC values for different k and a values

Figure 4 shows a heatmap of the minimum NRC that represents the NRC of the top-ranked sequence in the reference database, providing a direct measure of the best-case similarity between the sample and the database.

The best performance at $k = 15$, $\alpha = 0.01$ (minimum NRC 0.0420) aligns with the previous results, where the same parameters yielded the lowest mean NRC.

This consistency indicates that $k = 15$ and $\alpha = 0.01$ represent the optimal combination for both global performance (mean NRC) and best-case similarity detection (minimum NRC).

2.3.3 NRC Comparison Matrix - Sequence Similarity

To explore the relationships between the top 20 sequences identified by the `MetaClass` program, we constructed a sequence similarity matrix using the Normalized Relative Compression (NRC) values computed pairwise between these sequences. This matrix provides a comprehensive view of how similar the top-ranked sequences are to each other, revealing potential redundancies and clusters of related sequences.

The matrix was visualized as a heatmap, where each cell (i, j) represents the NRC of sequence j when compressed using a model trained on sequence i.

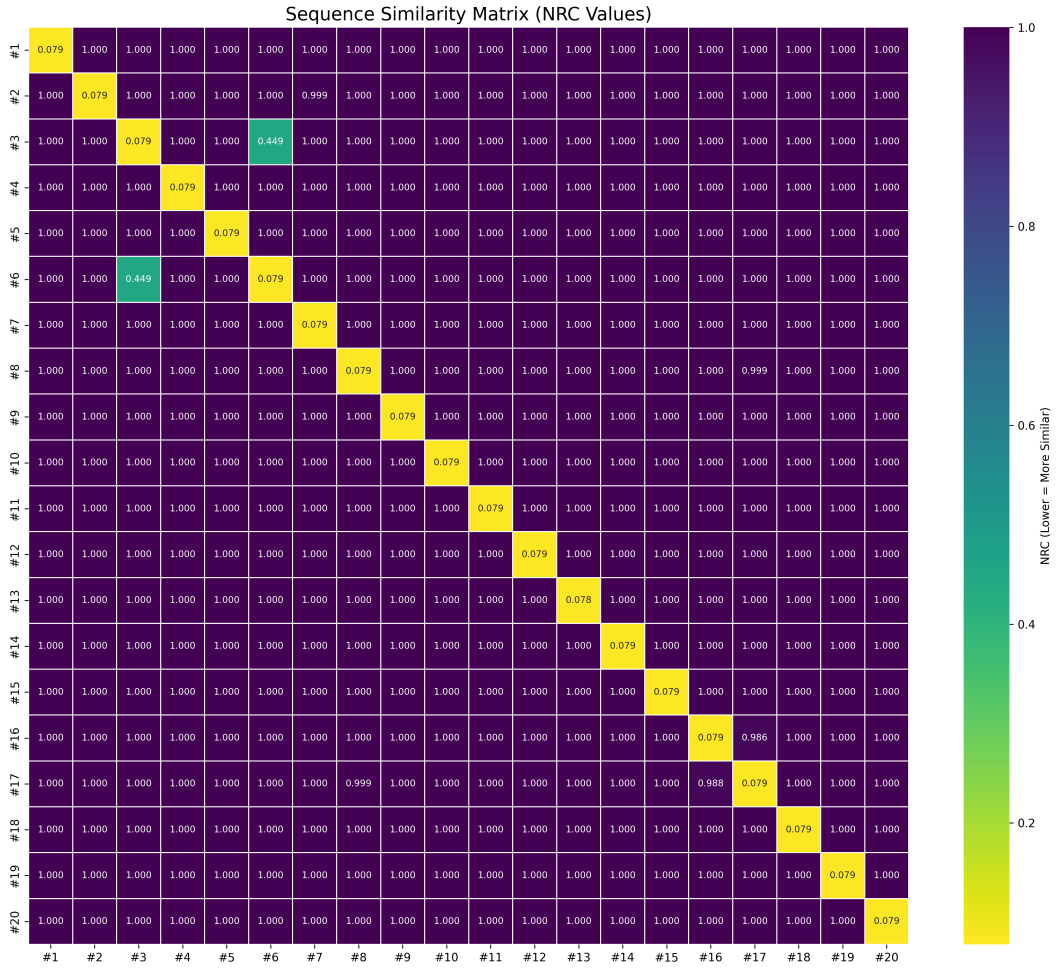


Figure 5: Sequence Similarity Matrix of Top 20 similar sequences based on NRC values

The sequence similarity matrix is shown in Figure 10. The diagonal of the matrix (where $i = j$) shows a cluster of low NRC values of around 0.079 for all sequences. This value represents the self-similarity of each sequence when compressed using a model trained on itself.

The non-zero value (instead of 0) arises due to the smoothing parameter $\alpha = 0.01$ and the normalization in the NRC formula.

Sequences #3 (Human Coronavirus NL63, complete genome, `NC_005831.2`, NRC 0.046122) and #6 (Human Coronavirus NL63, complete genome, `gi|49169782|ref|NC_005831.2|`, NRC 0.446922) form another cluster, with an NRC of 0.449 between them, confirming their similarity as duplicates or variants of the same virus.

Most off-diagonal entries outside the identified clusters have NRC values of 1.000, indicating maximum dissimilarity.

2.3.4 Complexity Profiles

To gain deeper insights into the structural properties of the sequences, we generated complexity profiles for selected entries among the top-ranked results. By visualizing these profiles, it becomes possible to detect highly repetitive or unique segments, which may indicate conserved biological functions, repetitive motifs, or artifacts such as contamination.

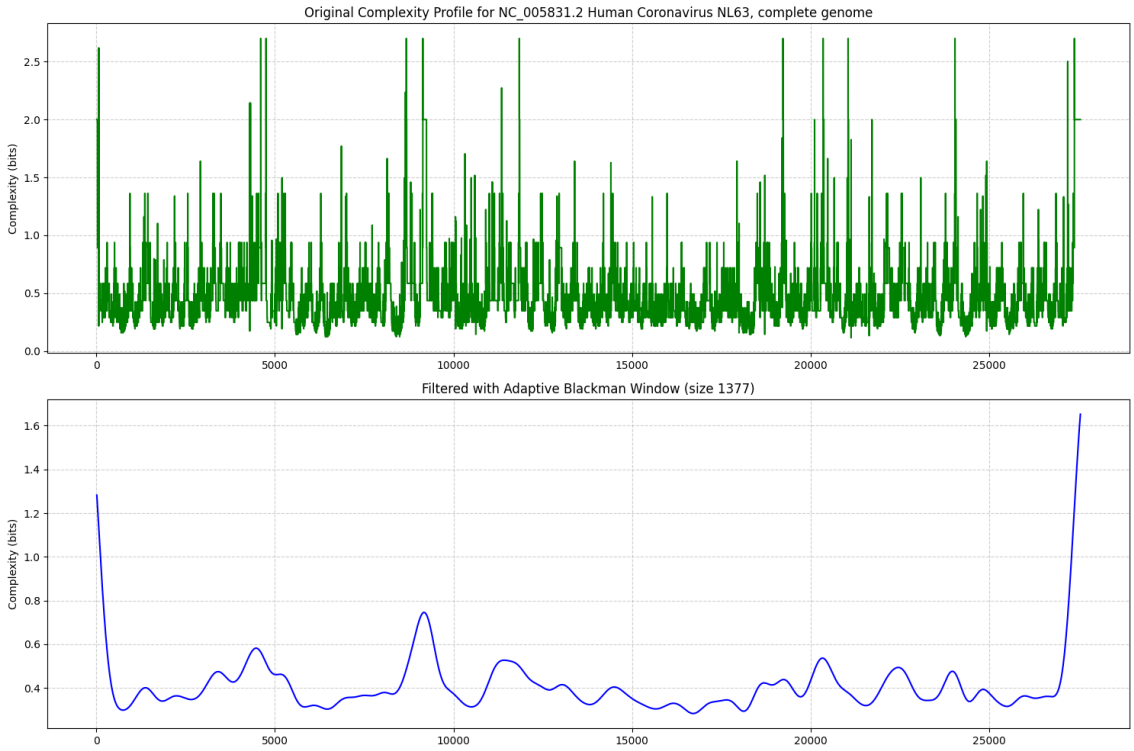


Figure 6: Complexity profiles for NC_005831.2_Human_Coronavirus_NL63(original and softened with blackman window)

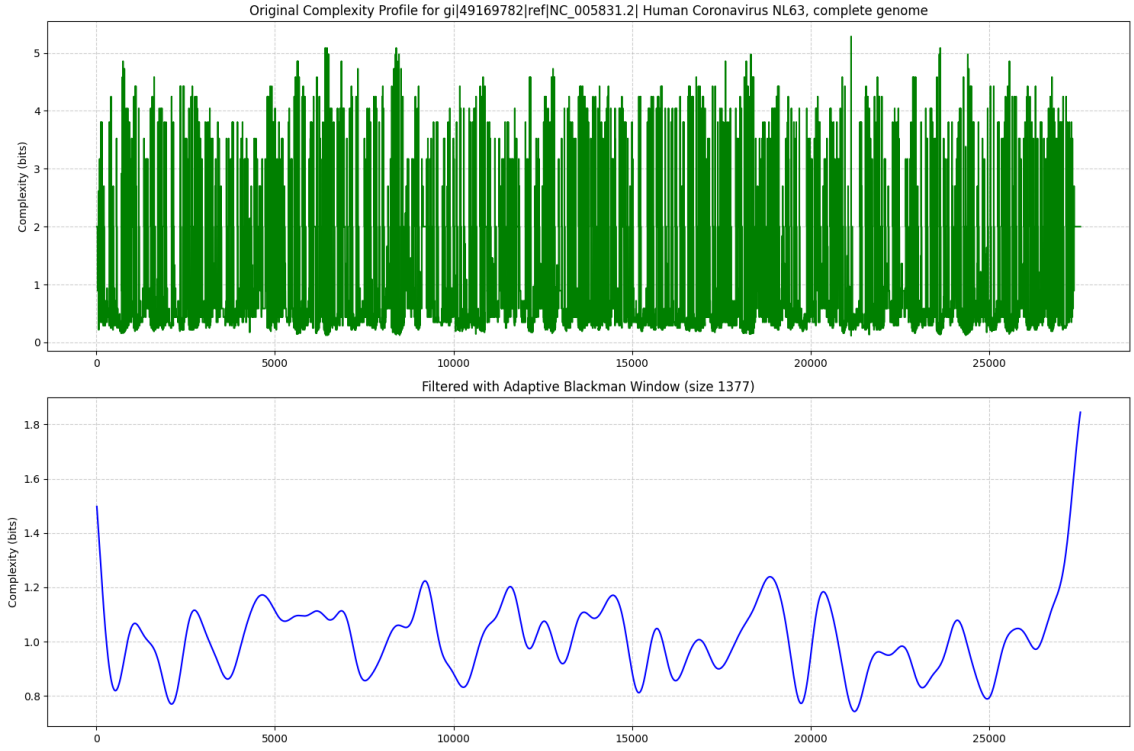


Figure 7: Complexity profiles for gi—49169782—ref—NC_005831.2—_Human_Coronavirus_NL63,_complete_genome (softened with window of size 2000)

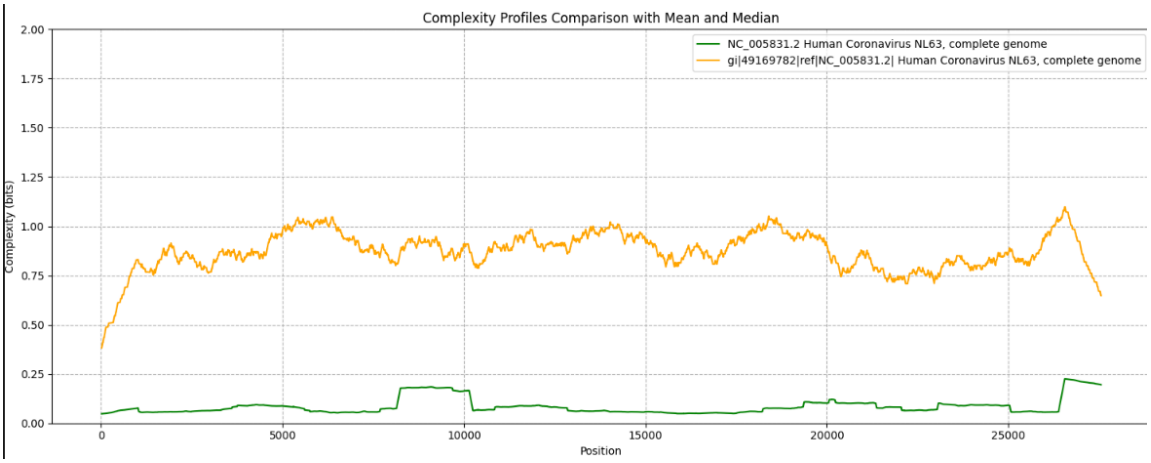


Figure 8: Comparison of Complexity profiles for the two corovavirus sequences

As shown in Figure 6, Figure 7 and Figure 8, both complexity profiles exhibit similar curves, most notably the curve before position 10,000. This similarity suggests that the sequences may share a common structure or origin. However, the differences observed in the form of distinct spikes could be indicative of mutations or contamination events, which alter the local compressibility and, consequently, the shape of the profile.

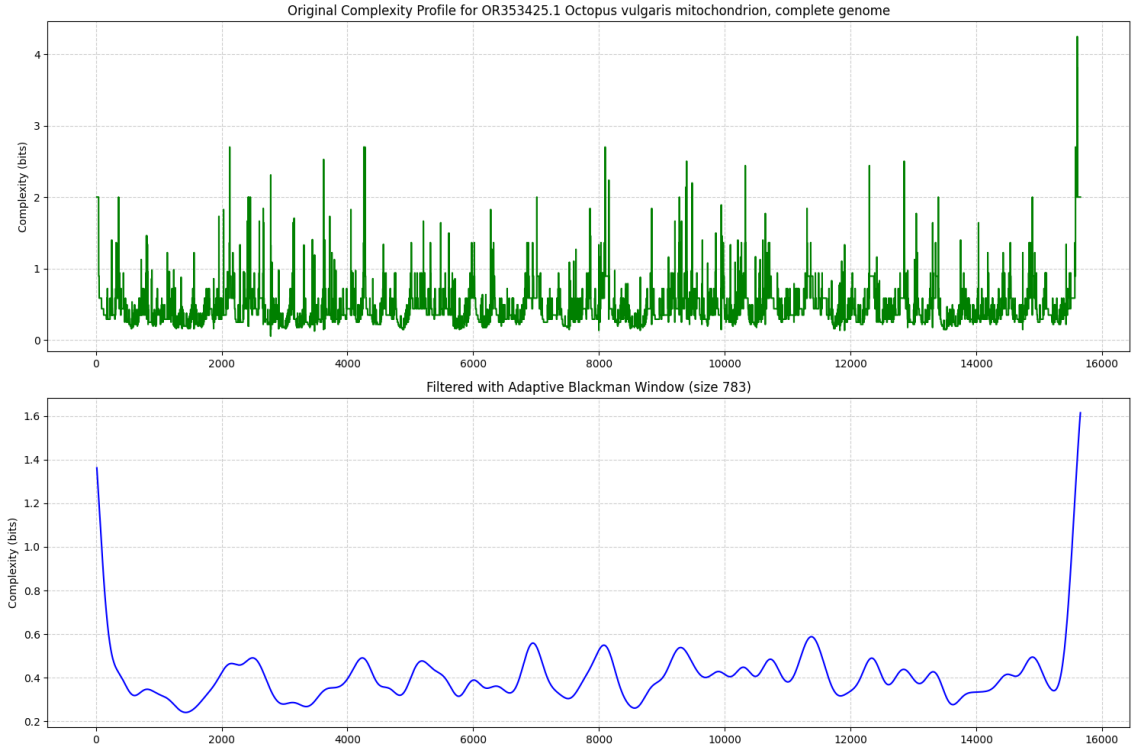


Figure 9: Complexity profiles for OR353425.1_Octopus_vulgaris_mitochondrion,_complete_genome(original and softened with blackman window)

Figure 9 presents the complexity profile of the top-ranked sequence in the results. This profile allows us to observe the internal structure of the sequence in terms of local information content.

The sharp variations and spikes in the raw profile may result from conserved elements, mobile genetic elements, or even contamination or sequencing artifacts.

2.3.5 Synthetic DNA Sequence Generation for Testing

To evaluate and validate the behavior of the `MetaClass` application under controlled and repeatable conditions, we created a synthetic reference database consisting of randomized DNA sequences. This was achieved using the **GTO toolkit** [**?**], a suite of bioinformatics tools that includes utilities for sequence generation, transformation, and mutation.

Specifically, we used the `gto_genomic_gen_random_dna` tool, which generates pseudo-random DNA sequences based on a user-defined seed and length. To automate the creation of a reference database containing multiple sequences, we wrote a Bash script, shown below:

Listing 2: Script to generate synthetic DNA sequences

```
#!/bin/bash

if [ $# -ne 1 ]; then
  echo "Usage: $0 <number_of_iterations>"
  exit 1
fi

iterations=$1
output_file="generated_db.txt"
> $output_file

for i in $(seq 1 $iterations); do
  random_seed=$RANDOM
  random_length=$(( RANDOM % 15001 + 5000 ))
  echo "@$i" >> $output_file
  ./gto_genomic_gen_random_dna -s $random_seed -n $random_length >> $output_file
  echo "" >> $output_file
  echo "Generated sequence $i with length $random_length and seed $random_seed"
done

echo "Generated $iterations DNA sequences in $output_file"
```

Sequence Properties

- Each DNA sequence is between **5,000 and 20,000 base pairs** long, generated with a randomly selected length per iteration.
- Each sequence is labeled using a FASTA-style header (`@<ID>`) to facilitate identification within the database.
- Sequences are appended to a single text file named `generated_db.txt`.

This synthetic dataset enables robust testing of the compression models by simulating various genomic patterns and sequence lengths. It also serves as a baseline for performance profiling and validation of the NRC similarity metric in the absence of real-world annotated metagenomic data.

2.3.6 Construction of the Metagenomic Sample (`meta.txt`)

To simulate a realistic metagenomic sample containing fragments from various organisms, we manually selected a subset of sequences from the synthetic database generated earlier. The selected sequences were:

`1, 29, 51, 65, 67, 130, 142, 165, 179, 187`

These sequences were extracted from the reference database using simple command-line filtering tools (e.g., `awk`) based on their FASTA-style headers (e.g., `@29`, `@165`, etc.). Their content was concatenated to form a single file named `meta.txt`, which serves as the metagenomic sample input for the `MetaClass` application.

This synthetic sample simulates the scenario where a mixture of DNA fragments from various sources is analyzed for similarity against a known reference database.

2.3.7 Simulating Mutations on Reference Sequences

To further test the sensitivity of the NRC-based similarity metric to small genomic changes, we introduced synthetic mutations to three sequences from the original database: `@1`, `@29`, and `@65`. This was accomplished using the `gto_genomic_dna_mutate` tool from the GTO toolkit, which allows controlled modification of DNA sequences through substitutions, insertions, and deletions.

The following command was used to mutate sequence 1:

Listing 3: Mutation of sequence 1

```
./gto_genomic_dna_mutate -s 42 -m 0.015 -i 0.005 -d 0.005 -a < seq1.txt > seq1_mut.txt
```

Explanation of Parameters:

- `-s 42`: Specifies the seed for the random number generator to ensure reproducibility.
- `-m 0.015`: Introduces a mutation rate of 1.5%, randomly substituting bases within the sequence.
- `-i 0.005`: Applies a 0.5% insertion rate, adding random bases into the sequence.
- `-d 0.005`: Applies a 0.5% deletion rate, removing random bases from the sequence.
- `-a`: Enables usage of the extended DNA alphabet including the symbol `N`.

Sequences `@29` and `@65` were mutated using the same insertion and deletion parameters but retained the default mutation rate of 1% (`-m 0.01`) and different seeds to ensure unique variations.

After generating the mutated sequences, they were appended to the reference database file `db.txt` using unique identifiers to avoid confusion. The new headers used were:

- `@1-mutated`
- `@29-mutated`
- `@65-mutated`

This setup allowed us to validate whether the `MetaClass` system could detect high similarity between original and mutated sequences.

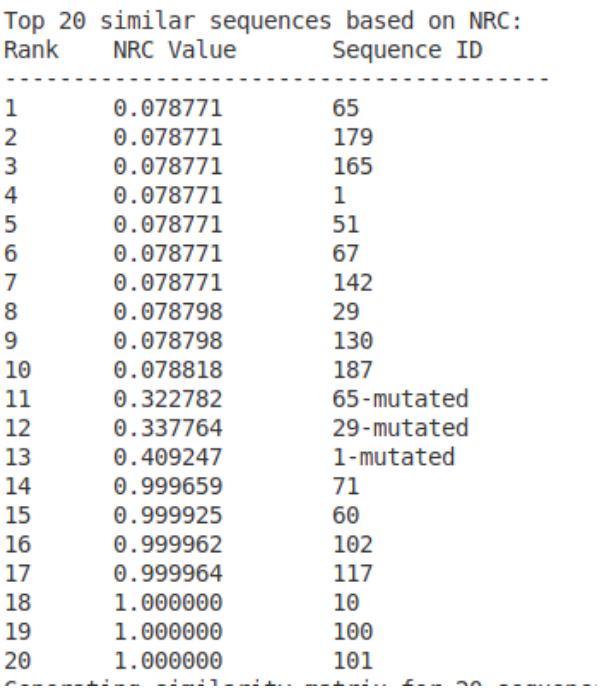
2.3.8 Results and Analysis of Sequence Similarity

To evaluate the effectiveness of the `MetaClass` application in identifying sequences similar to a metagenomic sample, we executed the following command:

Listing 4: Running MetaClass with NRC matrix output and complexity profiles

```
./MetaClass -cp -d db.txt -s meta.txt -k 15 -a 0.01 -t 20 -c similarity_matrix.csv
```

The results presented below correspond to the top 20 sequences in the database ranked by their NRC (Normalized Relative Compression) values with respect to the metagenomic sample (`meta.txt`):



```
Top 20 similar sequences based on NRC:
Rank    NRC Value       Sequence ID
--------------------------------------
1       0.078771        65
2       0.078771        179
3       0.078771        165
4       0.078771        1
5       0.078771        51
6       0.078771        67
7       0.078771        142
8       0.078798        29
9       0.078798        130
10      0.078818        187
11      0.322782        65-mutated
12      0.337764        29-mutated
13      0.409247        1-mutated
14      0.999659        71
15      0.999925        60
16      0.999962        102
17      0.999964        117
18      1.000000        10
19      1.000000        100
20      1.000000        101
```

Figure 10: Top 20 similar sequences based on NRC for the synthetic sample and db. Lower values indicate higher similarity.

Observations:

- The **first 10 sequences** in the ranking are exactly those used to build the metagenomic sample (`1, 29, 51, 65, 67, 130, 142, 165, 179, 187`). Their consistently low NRC values (approximately **0.078**) confirm that the compression model trained on the sample encodes them with high efficiency, which is expected for known content.
- The **11th**, **12th**, and **13th** ranked sequences are `65-mutated`, `29-mutated`, and `1-mutated`, respectively. Despite being synthetically mutated versions of sequences in the metagenomic sample, they still appear near the top with NRC values below 0.35. This demonstrates that the system can detect and rank near-identical or highly similar sequences even after structural modifications.
- The remaining sequences (`71, 60, 102, ...`) show NRC values close to **1.0**, indicating that they are dissimilar to the metagenomic sample. This reinforces the reliability of the NRC metric in excluding unrelated DNA sequences.

2.3.9 Pairwise NRC Matrix Analysis

To better understand how the most similar sequences relate not only to the metagenomic sample but also to each other, we computed a pairwise NRC matrix among the top 20 ranked sequences. Each cell (i, j) in this matrix represents the NRC value obtained by encoding sequence j using a model trained on sequence i. This analysis reveals intra-group relationships and allows us to identify clusters, redundancies, or asymmetries in similarity.

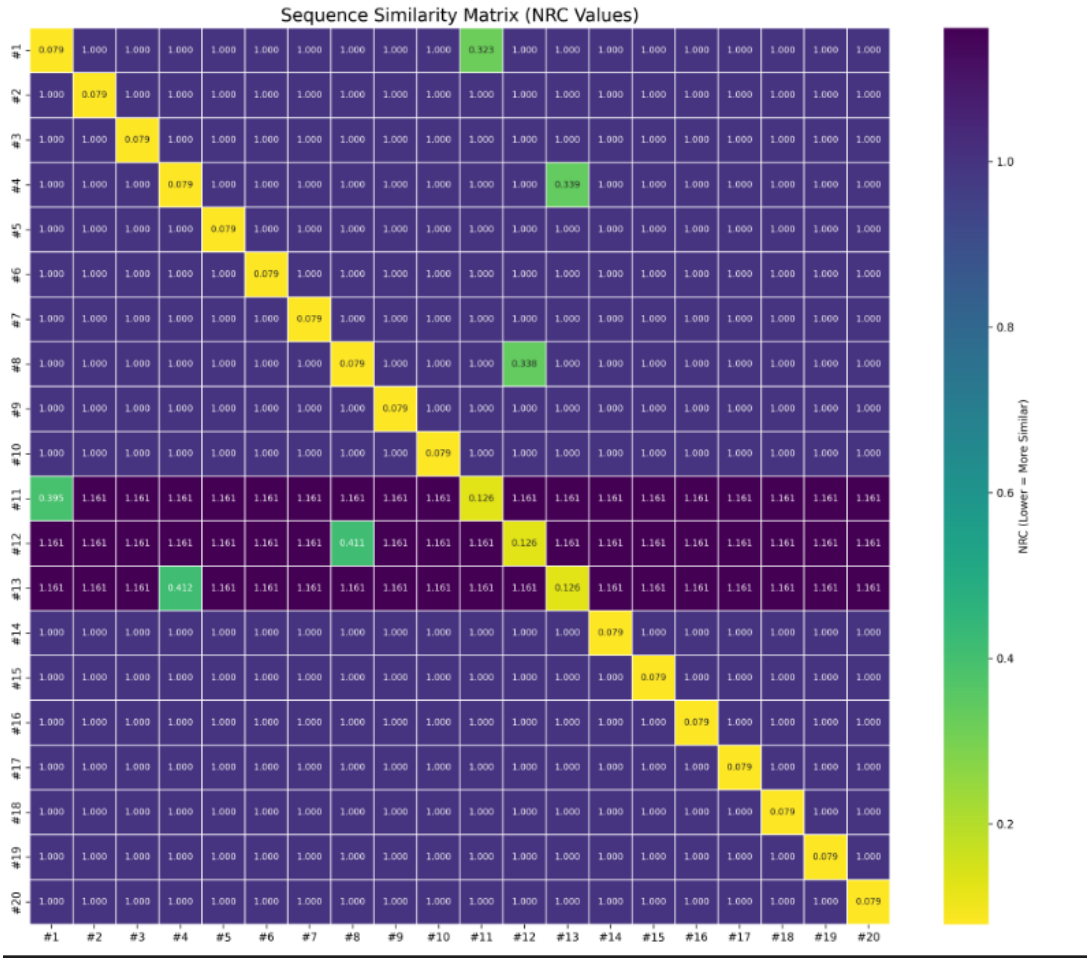


Figure 11: NRC-based Similarity Matrix among the Top 20 sequences

Matrix Observations:

- The **main diagonal** presents very low NRC values (≈ 0.078), which is expected as each sequence is encoded by a model trained on itself. This confirms the validity of the compression approach and the consistency of self-modeling.
- The **original sequences** from the metagenomic sample (`1, 29, 51, 65, 67, 130, 142, 165, 179, 187`) yield high NRC values (≈ 1.000) when encoded by models trained on other original sequences. This suggests statistical independence among them, confirming they were generated randomly and do not share compressible patterns.
- The **mutated sequences** (`1-mutated`, `29-mutated`, and `65-mutated`) all show:
  - **Lower NRC values** when compressed using the model trained on their original counterparts:
    - `65-mutated` encoded by `65`: **0.394562**
    - `1-mutated` encoded by `1`: **0.489530**
    - `29-mutated` encoded by `29`: **0.410908**
  - **Symmetric similarity**: when the original sequence is encoded using the mutated model, the NRC also remains low:
    - `65` encoded by `65-mutated`: **0.322782**
    - `1` encoded by `1-mutated`: **0.409247**
    - `29` encoded by `29-mutated`: **0.337736**

    This demonstrates that despite minor mutations, the statistical structure is largely preserved, allowing for mutual compressibility.

- The presence of symbols outside the standard ACGT alphabet (such as `N`, introduced by the `-a` flag during mutation) does not prevent the model from recognizing similarity in practice — although it may still lead to elevated NRC values in other contexts.
- Sequences unrelated to the metagenomic sample (`71, 60, 102, 117, 10, 100, 101`) consistently show NRC values near **1.000** when compared to any other sequence, confirming the model's capacity to reject non-matching entries.
- Despite being unrelated, each of those background sequences still yields a low NRC when modeled on itself (≈ 0.078), affirming consistent self-compression behavior.

Interpretation of NRC Values higher than 1.0: The occurrence of NRC values above 1.0, particularly between mutated and original sequences, stems from the inclusion of ambiguous bases (e.g., `N`) during the mutation process. Since the Finite Context Model in our implementation assumes a fixed DNA alphabet (`A`, `C`, `G`, `T`), the introduction of the symbol `N` leads to low-probability events and inflated bit estimates. Therefore, not only do the NRC scores for mutated sequences appear higher, but their perceived similarity to the original sequences is underestimated. If the alphabet were adjusted to include all characters present in mutated sequences, the NRC values would likely decrease, reflecting a closer resemblance.

Summary: The pairwise NRC matrix provides strong evidence of the Finite Context Model's reliability for sequence comparison:

- It detects and distinguishes individual sequences effectively.
- It correctly clusters both original and mutated versions together, with bidirectional similarity.
- It consistently separates unrelated entries through high NRC scores.

This symmetric behavior in compression between original and mutated sequences reinforces the robustness of the NRC metric for use in comparative metagenomics and mutation-aware applications.

2.3.10 Complexity Profiles Comparison

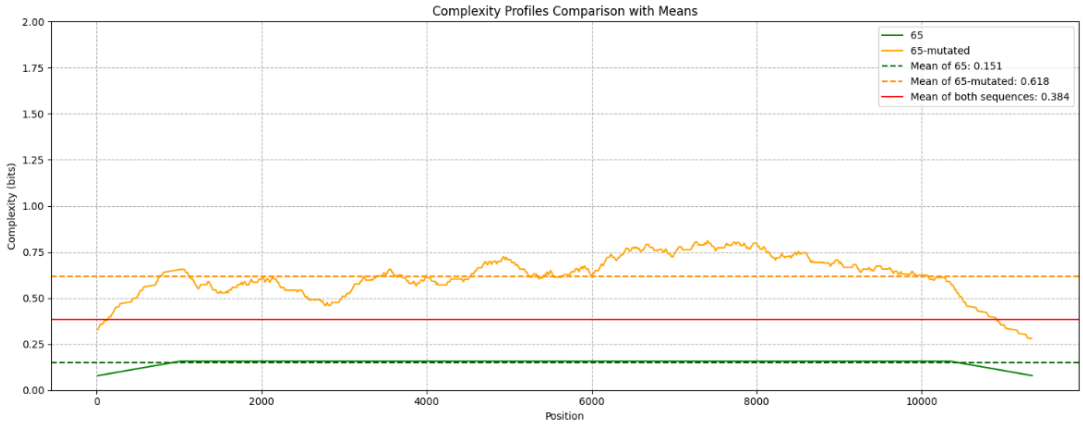


Figure 12: NComplexity Profiles Comparison of sequence 65 and 65-mutated

To further refine our ability to separate sequences that the metagenomic sample contains from those it does not, we derive an NRC threshold from the median complexity of the sequence 65 and 65-mutated complexity profiles (0.384 bits) and compare the NRC values of the top 20 sequences against this threshold. Sequences 65 and 65-mutated were chosen because of their similarity observed by the matrix in Figure 12. The hypothesis is that sequences with NRC values below this threshold are more likely to be contained in the sample (indicating high similarity), while those above the threshold are less likely to be contained.

2.3.11 False Positives Analysis:

threshold e semi refactor do FP e TP

To assess the system's precision, we first defined a hardcoded threshold of 0.6 for classifying True Negatives. Sequences with NRC values at or above this threshold are considered dissimilar to the metagenomic sample and are thus classified as True Negatives. Sequences below this threshold are evaluated further to determine whether they are True Positives or False Positives, using the previously defined NRC threshold of **0.384**, derived from the median complexity of sequences 65 and 65-mutated.

Sequences below the 0.384 threshold are considered similar enough to be classified as *positive*:

- **True Positives (TP)**: Sequences from the metagenomic sample with NRC < 0.384.
- **False Positives (FP)**: Sequences not in the metagenomic sample, but with NRC < 0.384.

Sequence ID	NRC Value	Classification
65	0.078771	True Positive
179	0.078771	True Positive
165	0.078771	True Positive
1	0.078771	True Positive
51	0.078771	True Positive
67	0.078771	True Positive
142	0.078771	True Positive
29	0.078798	True Positive
130	0.078798	True Positive
187	0.078818	True Positive
65-mutated	0.322782	True Positive
29-mutated	0.337764	True Positive
1-mutated	0.409247	**False Positive**
71	0.999659	True Negative
60	0.999925	True Negative
102	0.999962	True Negative
117	0.999964	True Negative
10	1.000000	True Negative
100	1.000000	True Negative
101	1.000000	True Negative

Table 3: Classification of Top 20 Results with NRC Threshold of 0.384

Notably, sequence `1-mutated` is classified as a False Positive because its NRC value (0.409247) lies above the 0.384 threshold. This can be attributed to its higher mutation rate (1.5% substitution rate compared to 1% for `65-mutated` and `29-mutated`), which introduces greater divergence from the original sequence `1` contained in the metagenomic sample. Consequently, `1-mutated` is sufficiently different to fall outside the True Positive threshold. In contrast, `65-mutated` (NRC 0.322782) and `29-mutated` (NRC 0.337764) are classified as True Positives due to their NRC values being below 0.384. Although these mutated sequences are not truly contained in the metagenomic sample, their high similarity to the original sequences `65` and `29` (owing to a lower mutation rate) results in their inclusion as True Positives.

These results demonstrate the effectiveness of the Finite Context Model in capturing similarities between DNA sequences based on compression. The NRC metric successfully identifies both exact and closely related (mutated) sequences, while distinguishing unrelated entries in the database. The behavior observed validates the model's reliability in real-world metagenomic comparison tasks.

Out of the 20 top-ranked sequences:

- True Positives (TP): 12
- False Positives (FP): 1
- True Negatives (TN): 7
- Total Sequences: 20

Accuracy is calculated as:

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{Total}} = \frac{12 + 7}{20} = \frac{19}{20} = 0.95$$

This indicates that the system correctly classified 95% of the sequences in the top 20 results.

3 Conclusion

The NRC-based similarity metric proved to be a robust and alignment-free approach, effectively identifying exact matches and near-identical mutated variants of sequences present in a synthetic metagenomic sample. Our synthetic dataset, constructed with randomized DNA sequences and carefully controlled mutations, enabled repeatable and interpretable evaluations. Experiments showed that the NRC scores reflect true biological similarity, and the system was able to classify sequences with high accuracy.

The use of a pairwise NRC matrix provided valuable insights into intra-group relationships, revealing both statistical independence among unrelated sequences and high mutual compressibility among mutated pairs.

By deriving an empirical NRC threshold from the complexity profiles of original and mutated sequences, we improved the interpretation of similarity scores and reduced the risk of misclassification. With this strategy, we achieved an accuracy of **95%** in the classification of top-ranked sequences into true positives, false positives, and true negatives.

Overall, the study demonstrates the potential of compression-based techniques for comparative genomics and mutation-aware analysis in metagenomic pipelines. The simplicity of the approach, its scalability, and the meaningfulness of its outputs make NRC and FCM highly promising tools in the broader landscape of bioinformatics.