Silesian University of Technology
Faculty of Automatic Control, Electronics
and Computer Science

PhD Thesis

Methods for similarity analysis of low complexity regions in protein sequences

Patryk Jarnot

Supervisor: dr hab. inż. Aleksandra Gruca, prof. PŚ.
Co-supervisor: dr hab. Marcin Grynberg

Gliwice 2023

Contents

1 Introduction

Low Complexity Regions (LCRs) are common protein fragments of little amino acid diversity. The definition is imprecise, and scientists agree that they are simple sequences composed of homopolymers, Short Tandem Repeats (STRs) and irregular fragments characterized by low amino acid diversity [1–3]. Their exact number depends on a method used for identification. Examples of different LCR types are shown in Figure 1.1. Although they look like simple sequences containing very little information, they are common, and thus we should consider them as important protein fragments. Numerous research studies have already demonstrated their functional and structural properties. For instance, they exhibit binding properties and exist in domains that fold amyloid structures [4–6]. For a long time, however, scientists believed that LCRs are biologically irrelevant fragments of proteins that evolved in a neutral way [7]. They mainly studied High Complexity Regions (HCRs), and even masked LCRs to improve homology searches [8]. This leads to a hypothesis that commonly available protein clustering and searching tools were designed towards HCRs and they are not reliable to analyze similarity between LCRs.

Researchers began studying LCRs when they discovered that these regions are the root cause of evolutionary irrelevant proteins sequence alignments hits in protein similarity search methods. That is why Wootton & Federhen introduced a SEG method in their article on low complexity region statistics [9]. The method has been further added to the BLAST method as a solution to frequent false positive hits. Promponas et al. assumed that only the most frequently occurring residues in LCRs lead to misalignments and developed the CAST method. This method detects compositionally biased residues in LCRs and masks them for protein similarity searching purposes [10]. Alba et al. adapted a SIMPLE method, initially created for DNA sequences, to proteins, and argued that this method could be used to study the evolution and function of LCRs [11]. Li & Kahveci published an approach to LCRs that them recognizes the repetitions in these regions as an important factor for better understanding their biological roles [12]. Harrison wrote in 2017 an article about

```
>sp|P14922|CYC8_YEAST General transcriptional corepressor CYC8
MNPGGEQTIM EQPAQQQQQQ QQQQQQQQQQ AAVPQQPLDP LTQSTAETWL SIASLAETLG
DGDRAAMAYD ATLQFNPSSA KALTSLAHLY RSRDMFQRAA ELYERALLVN PELSDVWATL
GHCYLMLDDL QRAYNAYQQA LYHLSNPNVP KLWHGIGILY DRYGSLDYAE EAFAKVLELD
PHFEKANEIY FRLGIIYKHQ GKWSQALECF RYILPQPPAP LQEWDIWFQL GSVLESMGEW
QGAKEAYEHV LAQNQHHAKV LQQLGCLYGM SNVQFYDPQK ALDYLLKSLE ADPSDATTWY
HLGRVHMIRT DYTAAYDAFQ QAVNRDSRNP IFWCSIGVLY YQISQYRDAL DAYTRAIRLN
PYISEVWYDL GTLYETCNNQ LSDALDAYKQ AARLDVNNVH IRERLEALTK QLENPGNINK
SNGAPTNASP APPPVILQPT LQPNDQGNPL NTRISAQSAN ATASMVQQQH PAQQTPINSS
ATMYSNGASP QLQAQAQAQA QAQAQAQAQA QAQAQAQAQA QAQAQAQAQA QAQAQAHAQA
QAQAQAQAQA QAQAQAQQQQ QQQQQQQQQQ QQQQQQQQQQ QQQQQQQLQP LPRQQLQQKG
VSVQMLNPQQ GQPYITQPTV IQAHQLQPFS TQAMEHPQSS QLPPQQQQLQ SVQHPQQLQG
QPQAQAPQPL IQHNVEQNVL PQKRYMEGAI HTLVDAAVSS STHTENNTKS PRQPTHAIPT
QAPATGITNA EPQVKKQKLN SPNSNINKLV NTATSIEENA KSEVSNQSPA VVESNTNNTS
QEEKPVKANS IPSVIGAQEP PQEASPAEEA TKAASVSPST KPLNTEPESS SVQPTVSSES
STTKANDQST AETIELSTAT VPAEASPVED EVRQHSKEEN GTTEASAPST EEAEPAASRD
AEKQQDETAA TTITVIKPTL ETMETVKEEA KMREEEQTSQ EKSPQENTLP RENVVRQVEE
DENYDD
```

Figure 1.1: Protein sequence with low complexity parts highlighted in red. It contain homopolymeric repeat of glutamine at the begining of the sequence. In the middle there is short tandem repeat of glutamine and alanine followed by homopolymeric repeat of glutamine. Irregular LCRs are at the end of the sequence. LCRs were identified using SEG method with default parameters.

fLPS, which is a method for identifying compositional biases in LCRs [13]. This method was created to facilitate functional reasoning of identified fragment of proteins. Recently, Cascarina et al. published LCD-Composer method that detects LCRs with regularly distributed common residues [14]. In parallel to scientific advances related to LCRs, scientists studied tandem repeats in proteins. Homopolymers and STRs, which can be clear or blurred, are subsets of LCRs. Other methods which are able to detect these fragments include XSTREAM and T-REKS [15, 16]. Looking at the evolution of LCR identification methods, it is clear that they were initially intended to mask LCRs and recently to discover their biological roles.

The most popular method for searching for similar protein sequences is BLAST. It uses Smith-Waterman algorithm to align the sequences and a heurictic that uses High Scoring Parts of the sequence to speed up computation [17, 18]. In the past, the SEG method has been used to mask LCRs to improve homology searches [9]. This method was then replaced with composition-based scoring matrix correction

that reduces the score of frequently occurring residues [19]. This compositional bias is mainly introduced by frequent residues in LCRs and therefore the score of these amino acids is simply decreased. In the meantime, BLAST has begun to use BLOSUM and PAM families of scoring matrices to improve the search for distant homologies [20]. Subsequently, new methods were developed to find more distant evolutionary relationships between proteins. These methods look for similar proteins that are used to create Multiple Sequence Alignment (MSA) and use them as a replacement for the scoring matrix. This group of methods includes PSI-BLAST, HHblits and HMMER [21–23]. These methods also mask LCRs by decreasing their alignment scores. For instance, in HHblits, the default score increases the significance of rare amino acids because they are harder to introduce by chance and are thus considered important for protein evolution [24]. Selected methods for similarity analysis of protein sequences are described in the ,,Protein similarity comparison methods" section. The mentioned methods and their improvements support the hypothesis about the direction of development and specialization of these methods toward HCRs ignoring or even excluding LCRs from similarity analyses.

1.0.1 Motivation and goals

In this thesis I set the following goals:

- Analyse existing methods for LCR identification and visualise their results.
- Analyse existing methods for similarity analysis of LCRs.
- Improve searching for similar LCRs.
- Improve clustering of similar LCRs.

Several methods for LCR identification exist. We know that all of these methods can mask LCRs for better homology searches, and they can discover different types of biologically relevant proteins. Some of them can detect methionine-rich prion-like domains [25]. Others can be used to analyze polar LCRs [26]. However, to the best of my knowledge, a detailed comparison of them is missing in the scientific literature, which could also be used to combine them to produce new results. **As part of this dissertation, I compare existing methods for LCR identification and introduce a consensus method that uses the analyzed tools.**

The biological roles of some LCRs have already been discovered but are not as well described as HCRs. Scientists can discover their functional and structural properties through time-consuming and expensive wet-lab experimental methods. We can speed up this process by making initial assumptions about properties of a protein fragment based on inferences about other similar fragments whose role is already known. If we find similar LCRs with a known function, we can reduce the number of research scenarios and reduce the cost of experimental methods. Therefore, in silico methods have the potential to work in cooperation with experimental methods. Currently, we can use LCR identification methods to retrieve LCRs from databases, but we lack methods for their efficient comparison. **The main goal of this dissertation is to check whether existing methods for similarity analysis of protein sequences can be used to compare LCRs, and to develop new methods for their analysis.**

1.0.2 Thesis structure

This work is divided into eight main parts which are: Introduction, Description of sequence analysis methods, Low complexity identification methods, Analysis of canonical methods for protein sequence comparison in the case of LCRs, LCR-BLAST - searching for low complexity regions, GBSC - clustering short tandem repeats, Applications and Summary and conclusions. ,,Introduction" leads the reader through the background information and settles objectives of this thesis. ,,Description of sequence analysis methods" describes used in this thesis methods for LCR identification and sequence similarity comparison methods. ,,Low complexity identification methods" chapter compares selected LCR identification methods and it demonstrates an approach to visually analyse these fragments. In the ,,Analysis of canonical methods for protein sequence comparison in the case of LCRs" chapter, I compare three state-of-the-art methods of protein sequence similarity analysis for fragments of low complexity. These methods are BLAST, HHblits and CD-HIT [17,22,27]. In the ,,LCR-BLAST - searching for low complexity regions" chapter, I present LCR-BLAST method which is a new modification to BLAST for searching for similar LCRs. GBSC - clustering short tandem repeats chapter describes a new method for identifying and clustering STRs by similar repetitive patterns. Presented methods were already applied for several LCR analyses. Therefore, in ,,Applications" chapter I describe them to show usefulness of presented methods.

1.0.3 Most important results

In this thesis, I present new methods of LCR identification, visualization and comparison. New methods for LCR identification include the consensus method, which uses relationships among other methods, and GBSC, which identifies STRs. For LCR comparison, researchers can use LCR-BLAST, which is a new modification of the BLAST method, and for clustering, they can use the newly developed GBSC. These methods were evaluated by comparing them with other solutions and using them in case studies.

As a result, I published two articles in scientific journals as the first author. The first article entitled ,,PlaToLoCo: the first web meta-server for visualization and annotation of low complexity regions in proteins" is published in Nucleic Acid Research. This work combines selected methods for LCR identification into a consensus, and visualize the results [28]. The second article is ,,Insights from analyses of low complexity regions with canonical methods for protein sequence comparison" is published in Briefings in Bioinformatics [29]. This article provides HHblits and CD-HIT parameter sets methods for more efficient LCR analysis. It also warns the scientific community that even if we use optimized parameter sets, we still need to be aware of false positive hits caused by core design assumptions towards HCRs in these methods. I also published a conference paper at the International Conference on Man-Machine Interactions (ICMMI 2019) as the first author. The article entitled ,,LCR-BLAST—a new modification of BLAST to search for similar low complexity regions in protein sequences" introduces a new modification to the BLAST method for LCRs [30]. I propose there an optimal set of parameters for these fragments and an additional metric as an alternative to the E-value which sort the results independently of the LCR lengths. I am also co-author of three other articles that used the methods for similarity analysis of protein sequences presented in this thesis. These articles are: ,,Tandem repeats lead to sequence assembly errors and impose multi-level challenges for genome and protein databases" by Tørresen et al., ,,Common low complexity regions for SARS-CoV-2 and human proteomes as a potential multidirectional risk factor in vaccine development" by Gruca et al. and ,,Quantitative conformational analysis of functionally important electrostatic interactions in the intrinsically disordered region of the delta subunit of bacterial RNA polymerase" by Kubáň et al. These articles were published in Nucleic Acid Research, BMC Bioinformatics and Journal of American Chemical

Society respectively [31–33]. The GBSC method, which I designed and developed in the scope of this work, is included in the OPUS grant number 2020/39/B/ST6/03447 in which I am co-investigator. Additionaly, I used my experience from analyses of already existing methods for protein similarity comparison to design a new methods, which are part of PRELUDIUM grant number 2021/41/N/ST6/01919 of which I am the principal investigator.

2 Description of sequence analysis methods

2.1 Identification methods

In this section, I describe existing identification methods for different kinds of LCRs. CAST and fLPS are methods for identifying fragments with a strong bias to one or more residues. SEG and LCD-Composer detect fragments consisting of up to a few residue types. SIMPLE searches for STRs, while XSTREAM and T-REKS search more broadly for Tandem Repeats (TRs).

2.1.1 fLPS

fLPS is a tool which has been successfully used for a wide range of protein sequence analyses of non-standard amino acid composition fragments [13]. Authors propose parameter sets for long fragments of compositionally biased and short fragments of low complexity. It uses a scanning window to find highly biased domains below a given probability threshold. At the begining the method uses a long window length and then tries to shorten it keeping the same number of biased residues. The algorithm scans for low probability sequences using a short window. When multiple sequences overlap then only these with high fraction of biased residue types remain. Also, these with higher P-value are rejected. Subsequences of single residue type which overlap with fragments of different residue type are merged. Finally, merged fragments are then shrunk and extended to find these with the best P-value. Input sequences must be in FASTA format. The output format may be changed using the method's parameter. fLPS, depending on used parameters, may be comparable to the CAST or SEG method. The method does not provide information about repeats in the identified subsequences. It is fast and able to analyze large databases.

fLPS was used e.g. for analysis of prion-like protein and intrinsically disordered regions [34].

2.1.2 CAST

CAST is a commonly used method designed to identify and mask compositionally biased LCRs [10]. When used for masking it only hides biased residue types only. Therefore, it masks only these residues which may cause false positive alignments. At the same time, it leaves residues that can reveal homology relevance. For each sequence in the database, the algorithm creates 20 homopolymeric sequences from all 20 canonical amino acid types. It uses these homopolymeric sequences to perform a single forward pass in the parsing sequence. For each residue, it computes non-negative cumulative score by adding the previous score to the score of the actual residue pair. The actual pair score is calculated using a scoring matrix. Identified LCRs range from the first positive cumulative score to its maximal value. As input, CAST processes FASTA formatted sequences. As a result, it produces the same sequences with biased residues changed to X and their range with the type of residue it is biased to. Therefore, LCRs identified by this tool is comparable to these known from fLPS. It is able to handle large databases of protein sequences in a reliable time. The method has been used in many biological analyses [35].

2.1.3 SEG

SEG is the most popular method for LCR identification [9]. It is embedded in BLAST and can be turned on for LCR masking to improve homology searches between proteins. By design, it determines whether a protein fragment is low complexity using Shannon entropy. It scans a sequence using a window of a given length (W parameter) and checks whether the first threshold (*locut*) is met. If it is then the fragment is extended to the left and to the right until it reaches the second threshold (*hicut*). Input to the algorithm is FASTA formatted dataset. Output may be either masked sequences or positions of identified LCRs. This method searches for low entropy fragments of protein sequences, thus it may be comparable to LCR-Composer. The method has been already applied to many biological studies, but some research warns about its limitations to complexity measure only [36, 37].

2.1.4 LCD-Composer

LCD-Composer is a new LCR identification method [14]. It focuses entirely on biologically relevant domains rather than masking non-homologous domains. It assumes that residue distribution and composition are key factors in the biological roles of LCRs. It uses a scanning window to find fragments that meet the composition threshold and linear dispersion. The composition threshold is a user-specified percentage of a certain type of residue in the window. The user must specify residue types to be analyzed. Linear dispersion determines how the selected residue type is distributed in the window. High linear dispersion values mean that analyzing residues are regularly spread in the window. Overlapping identified fragments are merged and residues from the head and tail of the sequence that are out of thresholds are removed. The remaining residues form the identified LCR. It accepts sequences in FASTA format as input. The output displays detailed information about the composition of identified fragments. To obtain information about all residues, the algorithm must be run for each residue separately. Together with the description, the authors provided their implementation in python.

2.1.5 SIMPLE

SIMPLE is a method first published in 1986 by Tautz et al. for DNA sequences, further improved by Hancock & Armstrong in 1994 [38] [39]. In 2002, Alba et al. published the third version of SIMPLE and used it to analyze protein sequences [11]. This method focuses on the importance of STRs rather than on masking them. It works by scanning the sequences using a specified window. For each window, the algorithm checks how many times the central residue occurs in it. The number of occurrences is multiplied by user-specified parameters and stored as a simplicity score. The final score is calculated as a ratio of expected probability calculated from the simplicity score of randomly shuffled sequences and observed probability from the simplicity score of the query sequence. The central residue can be extended to short motifs to analyze STRs. As input, the tool takes FASTA formatted sequence. Output are ranges of STR locations. The method works on a fixed window length thus it may not be specic enough when reporting ranges, but it is sensitive for STR detection [9]. It was already used to study e.g. IDRs and transcription factor proteins [40].

2.1.6 T-REKS

T-REKS is the choice for tandem repeats that are strongly degenerated [16]. It can be applied for both DNA and protein sequences. It is designed to analyze the biological roles of tandem repeats. The algorithm filters out homorepeats considering them as well-studied. For protein sequences, it searches for short strings of length 2. The distances between them are grouped according to similar values where, the similarity threshold is given as a parameter. These groups are then clustered using K-means algorithm to handle the case where a single sequence contains multiple TRs of the same type. The length of the repeats is determined by finding a similar distance between repeating units of the same length. This action handles the case where a repeating unit is consists of multiple copies of the same short string. The maximum distance between repetitions is a parameter entered by the user. Finally, an MSA is created from each tandem repeat unit. If the similarity score is below a certain threshold, the algorithm removes the irrelevant flanking unit and re-evaluates the MSA. For MSA, it uses a ,,center-star" algorithm but also accepts external tools. As input, the tool accepts parameters with FASTA formatted sequences. As a result, it shows the repeating unit, consensus sequence, MSA and a range of repeating units in the original sequence. The method is comparable to XSTREAM. It has several parameters that must be specified and depend on the length of the sequence and the characteristics of the repetitive unit. Compared to other tools for TR identification, it is able to identify STRs. The method has already been used for many biological analyses. Among others, it was used to analyze TRs in plants [41].

2.1.7 XSTREAM

XSTREAM is a popular method for tandem repeats identification [15]. It is designed to also detect tandem repeats that are highly degenerated. Therefore it identifies neutrally-evolved TRs. The method is also designed to identify TRs of all repeating unit lengths. The algorithm is complex and uses repeat seeds to find TRs. Each seed has to pass several steps to test repeats and transform them. These steps are: preprocessing, detection, characterization and postprocessing. As preprocessing it detects seeds that are short repeating subsequences. Neighboring seeds are extended and compared using fractions of identical residues. TR domains, which are made up of two units are then expanded to join more similar units which occur in tandem. Resulting domain is then maximised joining residues which create repetitive units

and lay at both ends of the domain. Identified domain location is masked to prevent identification of the same domain in the future. Identified domains are then characterized by dividing them into repetitive units. These repetitive units are aligned using a multiple sequence alignment algorithm. From generated MSA, the method calculates the consensus sequence. Units at domain ends are compared to consensus and trimmed if the similarity to consensus is below user-defined threshold. Finally, neighboring TRs are compared using consensus sequence and merged if they are similar. Input to the method is a FASTA formatted sequence. Output is presented in HTML. It shows identified domains and similarity analysis of repetitive units. Since the method identifies degenerated repeats it is comparable to T-REKS. The method has complex architecture but is able to compute results in a reasonable time. It was successively used for the analysis of different kinds of TRs [42].

2.1.8 GBA

GBA is a graph-based method for LCR identification [12]. The method focuses on tandem, interspread and cryptic repeats. It introduces a modified Shannon entropy metric that relies on residue similarity. The similarity between residues is indicated by BLOSUM62 scoring matrix. The first step in identifying LCRs is to construct a directed, unweighted and acyclic graph. Nodes in this graph represent similar residue pairs that occur in a sliding window. Edges connect residue types that occur in the same repeating units. To determine repetitive patterns, GBA uses three distance thresholds. The algorithm finds the longest path in the constructed graph. It uses a modified Dijkstra algorithm to find the longest path. The longest identified paths represent the core of identified LCRs. This core is further extended using the modified Shannon entropy metric. Finally, the post-processing steps select the fragments consisting of LCRs. Selection is based on a statistical analysis of repeats from UniprotKB/Swiss-Prot. The method processes FASTA formatted sequences and reports LCR locations. The approach to LCRs has evolved from SEG, therefore the results of both methods should be comparable when analyzing LCRs. GBA uses canonical residue types, which complicates the use of an extended alphabet. On the other hand, both complexity and residual relationships are taken into account during LCR analyses. The method has already been used in practice to analyze the function of protein domains [43].

2.2 Protein similarity comparison methods

In this section, I describe methods for similarity analysis of protein sequences. We distinguish two groups of them. The first group searches for similar protein sequence fragments, while the second clusters similar sequences.

2.2.1 BLAST

BLAST is the most popular method for similar protein and nucleotide searches. It is designed to find local similarities between sequences. At its core, it performs Smith-Waterman algorithm, but this time-consuming operation is executed only on sequences with a high probability of similarity. To achieve this, the query and database sequences are divided into k-mers. K-mers that are similar to both query and database sequences are marked as High Scoring Words. High Scoring Words are used to find potential matches that consist of similar subsequences. These words are compared to fragments of the query sequence and are extended until they encounter a negative score. Fragments obtained from this operation are called High Scoring Parts (HSP). Finally, Smith-Waterman is performed on merged HSPs, and alignments with E-value above a given threshold are reported. The heuristic associated with HSP significantly speeds up the calculations, making the method able to handle huge protein and nucleotide databases. On the other hand, it makes the method suboptimal, and therefore better results may exist. BLAST is the successor of the FASTA method and is widely used by other algorithms where protein similarity is required e.g. protein sequence clustering [17, 44].

2.2.2 HHblits

HHblits is a broadly used tool for sensitive protein similarity searches. It is based on profile-profile comparison and is a response to common race to create a tool that finds sequences characterized by distinct similarity but which still share similar biological properties. Profile-profile comparison is a time-consuming process thus HHblits, like BLAST, uses prefiltering steps to filter out profiles that are most probably out of the similarity threshold. Before searching, the database of Hidden Markov Model (HMM) profiles must be created from FASTA file. In the pipeline for database of HMM profiles creation, MMseqs method clusters database of interest by sequence similarity. Then it creates MSA from each generated cluster using Clustal Omega [45]. Finally,

each MSA is converted to the profile of HMM. For searching, HHblits firstly converts the input sequence to a profile of HMM for HMM-HMM comparison. Then it scans the database and compares HMM profiles which were selected by prefiltering steps. HMM profiles that are above a given E-value threshold update the query profile of HMM and the next iteration begins. After the last iteration, the method reports the best hits that are above a given threshold. Even if this method need improvements in derivative analysis like secondary structure prediction it is frequently compared with PSI-BLAST as a faster and more sensitive [22].

2.2.3 CD-HIT

CD-HIT is a method designed for clustering large protein sequence databases in a short amount of time. It is based on local alignment between sequences. The method significantly decreases calculation time by limiting the number of performed alignments. It calculates sequence alignments between incoming sequences and sequences representing clusters. First, the method sorts the input database descending by sequence length. It selects the first (longest) sequence and creates a new cluster from it, marking this sequence as the cluster's representative. The second sequence is compared to the representative sequence of the first cluster. If the sequence similarity is above a certain threshold then the sequence is joined to this cluster, if not, then a new cluster is created with that sequence as a representative. All subsequent incoming sequences are treated the same way. The algorithm first checks whether the sequence is similar to any of the representative sequences. If similarity has been detected, then the sequence joins the cluster whose representative is similar to the sequence. If not, then the sequence becomes representative of the newly created cluster. As with other methods, the algorithm checks whether two sequences are likely to be similar before performing a final alignment. CD-HIT uses short word filtering for this purpose. Finally, the method reports clusters of similar sequences with similarity value to its representative. Other methods for protein sequence clustering include Uclust, MMseqs and MCL. CD-HIT was successively used to remove redundancy from protein databases, protein family classification, and for protein function analysis [27].

2.2.4 MMseqs2

MMseqs2 is a method for fast, sensitive searching, and clustering. Protein-protein comparison consists of three stages which are short word match, vectorized ungapped alignment and gapped alignment. Two sequences are considered to be similar if they pass all three stages. These stages are ordered from fastest and less accurate to slow but precise. Therefore, short word match is able to filter out pairs that are obviously dissimilar. It makes the method able to handle large datasets in a short time compared to other tools. Alignment stages provide detailed information about the similarity between proteins and filter out non-obvious cases that have passed previous stages. For clustering, the method reduces the alphabet. From sequences, it retrieves 20 k-mers characterized by a high probability to share significant homology information with other sequences. Input dataset is clustered by identical k-mers. This step is performed using a reduced alphabet, thus the number of significant hits is increased. In such created groups, the method marks the longest sequences as clusters' centroids. It compares each centroid sequence to other sequences in the cluster using the three-stage comparison. If the similarity is significant, the sequence is joined to the cluster. Finally, the method reports similar sequences to the query with corresponding statistics or provides a list of similar clusters [46]. The method has been already used by projects like UniProtKB and AlphaFold [47, 48].

2.2.5 MCL

MCL is a method designed to cluster objects described by a similarity metrics. Therefore, it is able to cluster protein sequences by similarity as we can use, for example, E-values to describe the distance between sequences. Enright et al. show that it is able to cluster large protein datasets into families. MCL can parse BLAST output and use it to create the initial network. The network is stored in a square matrix which is then processed by expansion and inflation operations until no changes are made. The algorithm is based on the assumption that by choosing a random node and moving through the transition to the neighboring node we will most likely move to another node in the same cluster. In other words, we assume that nodes in the same cluster are connected with many transitions, while inter-cluster connections are sparse and weak (low transition weight). In the algorithm, random walks in the network are called expansion. Inflation is an operation that boosts the probabilities of walks located in the cluster. The output of the method is pairs of similar sequences

that form clusters. The method clusters similar protein sequences into families, thus it is comparable to CD-HIT and MMseqs. The algorithm uses randomness which makes it non-deterministic, but is able to run in parallel and handle large datasets [49]. The method has already been applied to many studies that have investigated the similarity between proteins [50, 51].

3 Low complexity identification methods

3.1 Introduction

Scientists have already developed several methods to identify LCRs for different purposes. Some methods have been developed to mask LCRs for better homology searches, while others have been developed to discover their biological roles. These methods rely on complexity metric, repetitive patterns and composition biases. It is known that repeats may be relevant to protein structure. For instance, collagen is often a residual triplet where one residue in the pattern is glycine [52]. Prion proteins may contain five runs of P(H/Q)GG(-/G/S/T)WGQ pattern [53] Composition bias can also lead to the function of a protein fragment. For example, an argine-rich motif is key in a protein for binding to RNA [54]. Some efforts have already been made to link LCRs with disordered regions. However, it has been shown that combining a complexity metric along with the composition biases to one or more residue types can lead to a particular structure with a greater probability. [55–57] In 2020, Mier et al. published an article in which they attempted to characterize different types of LCRs based on their composition, periodicity and structure [58]. They also described SEG, CAST and SIMPLE methods separately, using a set of well-defined 21 protein sequences which represent a wide range of LCR roles and types. However, we have more methods of LCR identification, the results of which more or less overlap with each other. For instance, we have two methods to identify compositionally biased fragments. These methods are CAST and fLPS, and they are popular for biological analyses. Consider a scenario where a researchers have a compositionally biased fragment of a protein and want to identify more such fragments in a particular dataset. Then which of the two methods should they use? Scientific literature lacks of comparison of these two methods. Therefore, in this chapter, I explore a wide

range of LCR identification methods. I will show what type of protein fragments each method is able to recognize and how they overlap.

3.2 Methods

In this section, I present the comparison of LCR identification methods. First, I explain workflow of the comparison, then overlap between methods and finally I describe possible use of the consensus method. I use the following methods for the comparison: SEG, CAST, SIMPLE, GBA, XSTREAM, T-REKS, fLPS, LCD-Composer and GBSC [9–16]. An article about the last method is in preparation, but the method is described in the chapter ,,GBSC - clustering short tandem repeats".

3.2.1 Workflow

The workflow of the analysis is presented in Figure 3.1. As an input dataset I used canonical sequences from the UniProtKB/Swiss-Prot version 2022_05. I used it to extract different types of LCRs from sequences using selected methods. For SIMPLE, GBA, XSTREAM, fLPS and GBSC methods I adjusted the default parameters for the analysis. In the case of SIMPLE, I used the parameter set recommended in the PlaToLoCo service, which are 1 for each peptide length score, 11 as a window length, 100 for the number of random sequences, 3 random method and 0.9 as stringency [28]. The GBA parameters are the same as the authors used in their analysis when publishing an article about it ($t_1 : 15, t_2 : 3, t_3 : 5$) [12]. For XSTREAM I changed the minimum period to 1 which by default is set to 3, and thus excludes STRs. Parameters of the fLPS method are adjusted for LCRs as recommended by the authors, which are as follows: minimum tract length (m) is 15, maximum tract length (M) is 500 and binomial p-value (t) is 0.001 [13]. GBSC is a method developed in the scope of this dissertation, thus I used two parameter sets called GBSC-strict and GBSC-relaxed. The method with *strict* parameters uses the following values: 4 (w), 10 (l), 2 (m) and 6 (x). For relaxed parameter set, I increased l to 20, m to 3, and I set the n to ,,true". For the rest of the methods, I used the default parameter sets. Selected methods have been developed independently, hence their input and output formats are non-standardized. Therefore, I unified the output of all methods to FASTA formatted sequence fragments with additional information in the headers. I then examined the results of the methods by plotting their length distribution, amino

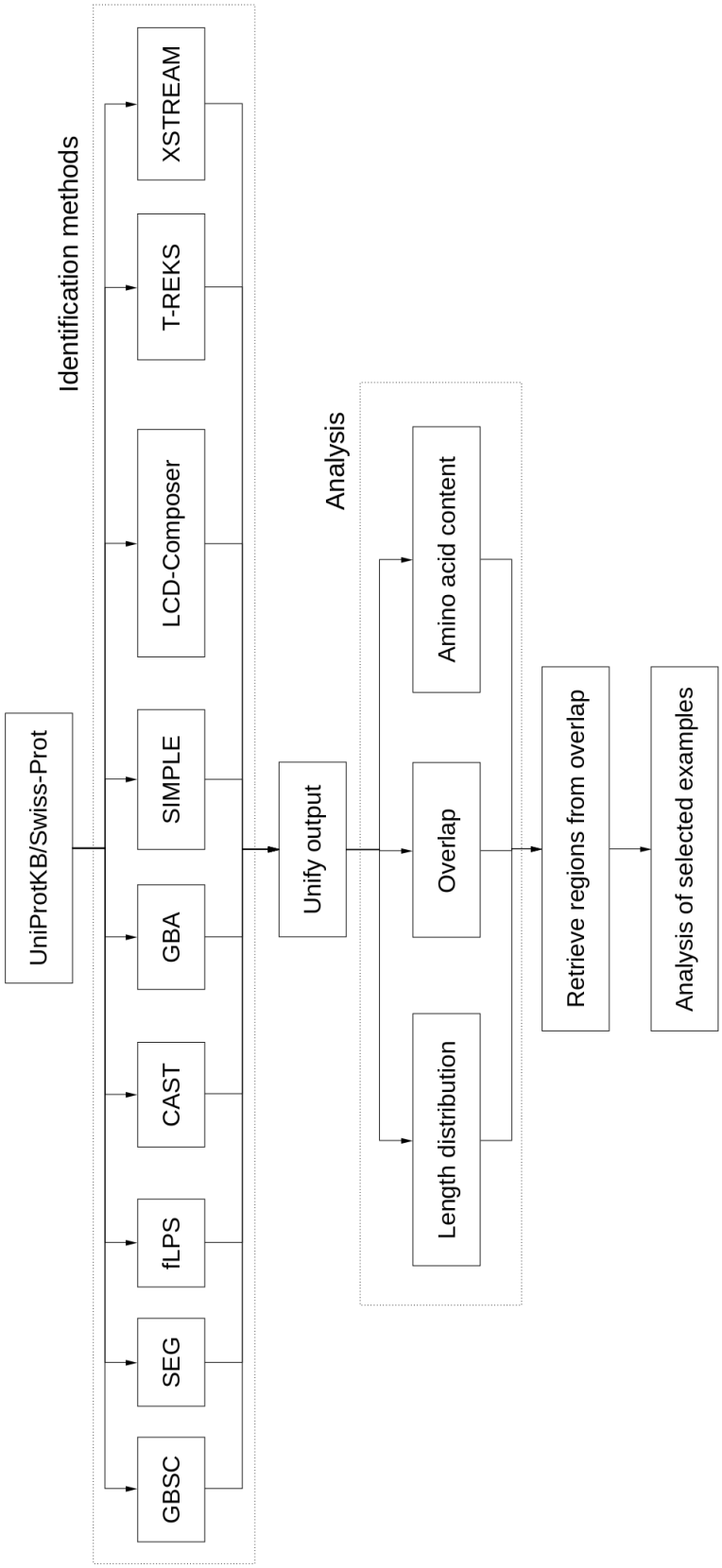

Figure 3.1: The workflow of the analysis of LCR identification methods. The selected methods were used to identify LCRs in the UniProtKB/Swiss-Prot database. The results were then analyzed quantitatively and qualitatively.

acid content and overlap. The amino acid content is also calculated for the input dataset for comparison. Finally, I used the overlap to select and analyse interesting cases in details.

3.2.2 Overlap analyses

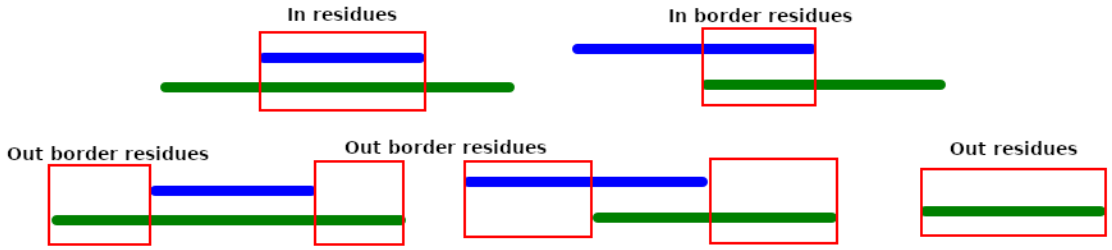

Figure 3.2: Residue location types in relation to overlap are: *in residues*, *in border residues*, *out border residues* and *out residues*.

The selected methods are designed to identify different types of LCRs. SEG and LCD-Composer identify fragments of several residue types, CAST and fLPS are used for composition biases while the rest identifies tandem repeats. To demonstrate how these methods relate to each other, I created the overlap of their results. It shows how many residues were recognized as LCRs by the two methods and how many were not. To make the overlap more informative, I distinguish between residues in domains that are fully covered by domains from another method, partial overlap, and the case where the domain is identified by only one of the two methods. This helps to determine whether two methods found similar or different domains, and whether one method is more specific than another. Figure 3.2 shows all four location types of residues in detail. I used all these types to pick interesting cases to describe each method and relationships between them.

3.2.3 Consensus

Each of the LCR identification methods separately provides a value for biological analyses. In this study, I show that we can combine them to produce new results. For instance, the intersection of several tandem repeat identification methods can report domains consisting of a clear pattern of tandem repeats, while the union of all methods can yield a wide range of tandem repeats from clear repetitive patterns to

all sorts of fuzzy patterns. In this method, researchers can choose the methods for consensus and its type, where possible consensus types are union and intersection. I use exploration and detailed analysis of the overlap to characterize the relationship between pairs of methods that could potentially be combined.

3.3 Results

3.3.1 Data exploration

Length distribution

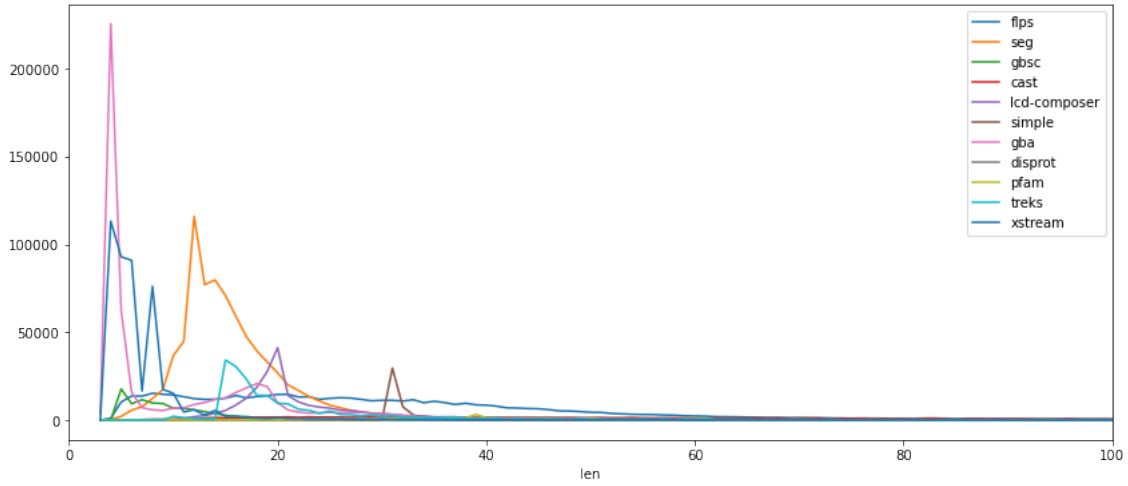

Figure 3.3: The number of detected domains by their lengths. Compositionally biased domains are well distributed over the chart while other types of LCRs are focused around a single point where they have identified the highest number of domains.

As presented in Figure 3.3, LCR domain lengths depend on the chosen method, and consequently on LCR types. These methods identify composition biases, short tandem repeats and fragments composed of several types of residues. Intuitively, perfect tandem repeats of short repetitive units should be shorter than fragments that are only compositionally biased to a certain residue, or at least because composition biases can cover a wide range of complexity. To investigate how the results of the methods vary in length, I generated a length distribution plot presented in Figure 3.3. There we see methods whose line is flat and whose results are accumulated near a single point. Methods with flat length distribution line are CAST and fLPS, which

are designed to detect composition biases. These methods identified the largest number of amino acids. Our results confirm previous conclusions that compositionally biased domains are superset of other types of LCR domains [59]. Even if CAST and fLPS detect fewer short domains than other methods, multiple short domains can be contained within one long compositionally biased domain. On the other hand, methods with accumulated length distribution near a single point rely on a sliding window of fixed size. These methods often extend the identified region until predefined condition is met. Depending on the method, this may be the upper complexity threshold (SEG) or the end of a repeatable pattern (GBSC). To analyze the selected methods, I grouped them according to the type of domains they are designed to identify. GBSC identifies STRs which are often short runs of a simple repetitive pattern, thus the method mainly identifies short domains. SEG has a more relaxed approach, and also allows for irregular LCRs. Therefore, the accumulation of its length distribution is shifted to the right. Another reason is that GBSC is able to distinguish between two different types of adjacent STRs. An example of this is presented in Figure 1.1, the second LCR. SEG identified it as single LCR, but GBSC will split it into two different STRs, which are STR of glutamine and alanine, and homopolymer of glutamine. Therefore, it is counted in SEG as a single long fragment while GBSC counts it as two shorter fragmetns. LCD-Composer lengths as expected partially overlap with SEG lengths and to partially be on the left side of SEG lengths since this method also may distinguish adjacent LCR types. The distribution GBA method is interesting since the function has two maximum extremas. The first is low, and equal to 4, while the second is around 18 residues. SIMPLE method identifies repetitive fragments which are slightly longer or equal to window size, and that is the reason why its length distribution is shifted far to the right in comparison to ther repeat identification methods. We can also suppose here that if SIMPLE identifies almost fixed-length domains, it will either skip some results or specify domain boundaries outside of their actual positions. GBSC, XSTREAM, T-REKS and SIMPLE methods can be used to identify repeats. From the chart we can see that XSTREAM fully covers GBSC results, but T-REKS and SIMPLE identified longer repeats, therefore they can be considered as method to identify different types of repeats. In detail, the differetces between these methods will be revealed in the ,,Detailed analysis" section of this chapter.

Amino acid frequencies

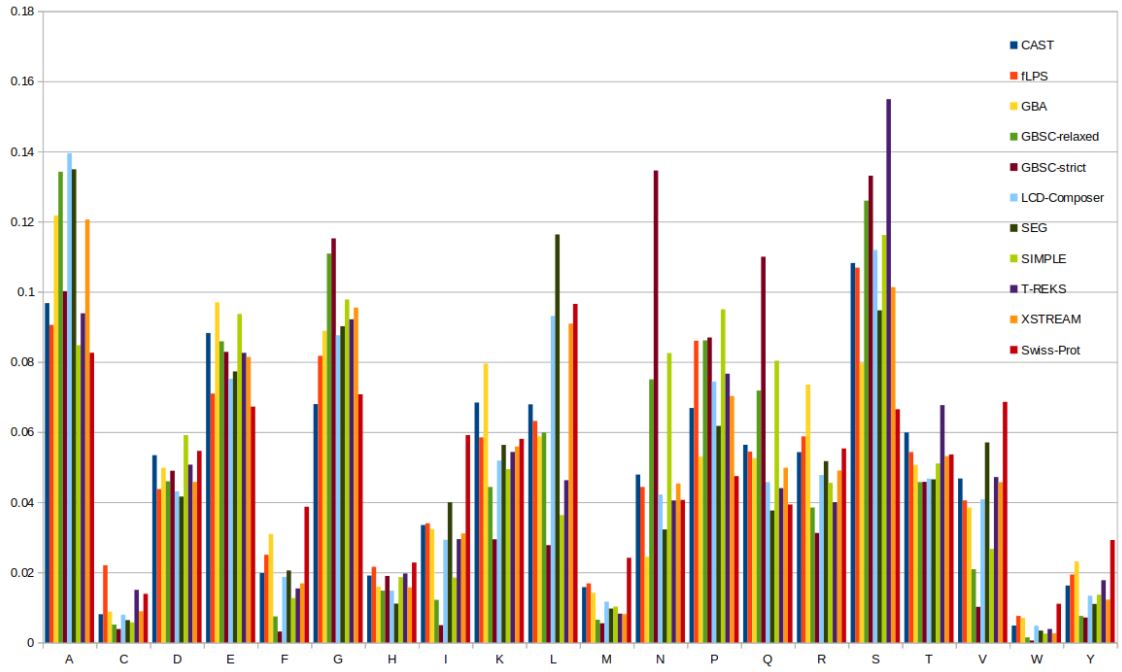

Figure 3.4: Frequencies of each residue type for each method. Domains identified by selected methods have different amino acid frequencies than the UniProtKB/Swiss-Prot, and the difference is higher for methods which identifies highly biased domains. Random frequency is 0.05.

Individual amino acids are more likely to occur in regions of low complexity than others. The average differences between LCR identification methods and the UniProtKB/Swiss-Prot database frequencies are as follow: fLPS (0.013), CAST (0.014), SEG (0.014), XSTREAM (0.015), GBA (0.016), LCD-Composer (0.016), T-REKS (0.019), SIMPLE (0.024), GBSC-relaxed (0.028). fLPS frequencies are the closest to these in the database while GBSC frequencies are the most outliers. The above methods are sorted by deviation, but also from those that allow the most diverse composition of residues in the domains to these prefering homogeneity and order. For instance, SEG and LCD-Composer are considered as methods for similar low complexity domain identification, but LCD-Composer also requires domains to be above dispersion parameter, which consequently increases their regularity. The exceptions here are XSTREAM and T-REKS, which indeed identify tandem repeats, but they also accept long patterns in domains.

Some amino acids are more likely to form short tandem repeats than others. For comparison, residues more common in the UniProtKB/Swiss-Prot database than in the GBSC results by a chance are: W (7.7 times), F (5.2), I (4.9), Y (3.9), M (3.7) and V (3.3). These residues are present in the UniProtKB/Swiss-Prot in the following percentages compared to all residues: 1.1% (W), 3.9% (F), 5.9%(I), 2.9%(Y), 2.4 %(M) and 6.9%(V). This contrasts with the highest GBSC frequencies versus UniProtKB/Swiss-Prot: S (1.89 times), N (1.85), Q (1.83), P (1.82), A (1.63) and G (1.57). We can see that the difference is significant and is also visible in the residue popularity. In the database, these residues have the following occurrence chance: 6.6% (S), 4.1% (N), 3.9% (Q), 4.7% (P), 8.3% (A) and 7.1% (G). Interestingly, the rare residues in the database are even rarer in the GBSC results, suggesting that these residues are not prone to repetition. This is especially evident in tryptophan and tyrosine. On the other hand, residues that are overrepresented in repeats are also common in average sequences. However this is not the rule. Valine has a higher occurrence rate than serine in the database, but is rare in STRs. If we look at the SR protein family, which is, among others, important for binding to RNA then we notice interesting relationships in frequencies of SR repeats. Phylogenetic tree analyses suggest that SR repeats are important in the evolution of the protein family [60, 61]. Therefore, we would expect from these residues to be frequent in the GBSC results. Even if this is true for serine, argine is much less frequent in the GBSC results than in the UniProtKB/Swiss-Prot.

Asparagine and glutamine are residues that have similar frequencies in some methods and databases, while different in others. Methods with similar frequencies include CAST, fLPS, GBA, LCD-Composer, SEG, T-REKS and XSTREAM. GBSC and SIMPLE, which search for STR, obtained higher values for these amino acids. Additionally, if we change the GBSC parameters to identify more pure repeats, the difference becomes even more visible. It seems that these residues tends to occur in simple sequences. In the GBSC results, 79% of glutamine is present in the poly-Q domain, 81% of asparagine in poly-N domains, while, for instance, only 54% of glycine is in the poly-G domains. Interestingly, glutamine homopolymers are already well studied, and is known to be the root cause of some disorders, thus it is rather expected to be less common in nature [62–64]. Glycine, on the other hand, is an important amino acid commonly found in collagen domains composed of repeating triplets [65]. Similar amino acid preferences were also observed in LCD-Composer by Cascarina et al. [14].

	CAST	fLPS	GBA	GBSC	LCD-Composer	SEG	SIMPLE	T-REKS	XSTREAM
CAST	18,129,024								
fLPS	7,396,409	13,812,877							
GBA	1,263,552	1,436,352	4,825,550						
GBSC	996,654	1,193,676	340,529	1,246,639					
LCD-Composer	3,565,831	4,228,890	806,489	905,772	5,372,664				
SEG	5,066,023	5,829,852	1,390,621	1,165,203	3,746,029	12,052,107			
SIMPLE	627,441	558,582	155,620	295,053	499,907	526,396	731,551		
T-REKS	1,815,399	1,775,806	375,255	642,569	1,214,186	1,656,244	338,623	2,995,775	
XSTREAM	1,919,929	2,071,087	638,689	919,761	1,280,495	2,171,070	354,740	1,408,019	4,000,128

Table 3.1: Overlap between LCR identification methods. Numbers determine how many residues from detected domains overlap between two methods.

Overlap between methods

CAST, fLPS and SEG identified significantly more residues than GBA, GBSC, LCD-Composer, SIMPLE, T-REKS and XSTREAM. This is justified by the fact that the first group of methods has a loose approach to LCR identification. They check whether one or more residues occur in a given protein fragment more often than in the average sequence (CAST, fLPS) or whether the complexity of the fragment is below a certain threshold. On the other hand, the second group of methods is finding patterns in protein sequences. LCD-Composer ensures that rare residues are well distributed in the detected domain, while other methods detect tandem repeats.

CAST identified the highest number of residues of all the selected methods. This method aligns all types of homopolymers to the sequence and checks whether it contains fragments with high alignment score to identify compositionally biased domains [10]. As a result, It found over 18.1 milion residues. Interestingly, the overlap with fLPS, which identified more than 13.8 million residues, is only nearly 7.4 million, and CAST covers only about 53.5% of fLPS residues. However, these two methods are considered to detect the same LCR type, thus I also investigate them in the ,,Detailed analysis" section. CAST covers fewer residues of SEG that is about 42.0%. It is reasonable since CAST identifies compositionally biased LCRs, while SEG identifies LCR based on their complexity. CAST covers 66.4% of the results reported by LCD-Composer. This value is higher than in the case of fLPS, but I must point out that LCD-Composer identified less residues than fLPS, which makes it easier to cover more results of LCD-Composer. The overlap between CAST and GBA was expected to be lower than between CAST and SEG or fLPS, but it is extremaly low and requires further investigation. T-REKS and XSTREAM have a similar number of residues overlapping with CAST. However, these two methods also identified a different number of residues in repeats, and T-REKS domains are covered in 60.6% by CAST regions while XSTREAM domains in only 48.0%. From Figure 3.3 we can see that XSTREAM identified short repeats, while T-REKS rather longer, which is closer to CAST behavior. Probably for the same reason, CAST largely overlaps with SIMPLE. Moreover, the definition of LCR by these two methods is similar: a given fragment of a protein sequence should be sufficiently enriched with a certain type of residue to detect it as LCR. However the SIMPLE method identified significantly lower number of residues than CAST since it detects biases in a window of fixed length. SIMPLE results are covered in about 85.8% by CAST. Results of

GBSC, which identifies repeats, are covered in 79.9% by CAST. In summary, the method is sensitive to longer compositionally biases fragments.

The second method by the number of residues identified is fLPS. According to the authors, the set of parameters used for the analysis should identify short fragments of compositionally bias [13]. The method uses scanning window of variable length to find highly biased domains below a given probability threshold. Even though it identified fewer residues than CAST by about 23.8%, it has higher overlap with the rest of the methods excluding SIMPLE and T-REKS. It has the greatest overlap with LCD-Composer, thus we can define a large part of the fLPS results as consisting of well-spaced residual types in the window. The overlap between the two methods is about 78.7%, which is about 12.3% more than between CAST and LCD-Composer. This suggests that these methods may complement each other if the domains identified by one method are insufficient. GBA and SEG results poorly overlap as it was in the case of CAST method. Interestingly, fLPS has less overlapped residues with SIMPLE (76.4%) compared to CAST and more with GBSC (95.8%). Similar situation, but less visible is with T-REKS and XSTREAM. Clues about the relationship between the two cases can be read from the length distributions of the detected domains. First of all, it is worth recalling that GBSC and SIMPLE are methods for STR, while T-REKS and XSTREAM are for TR. In Figure 3.3 we see that fLPS detects domains shorter than CAST, GBSC shorter than SIMPLE, and finally XSTREAM shorter than T-REKS. The above statements suggest that fLPS, GBSC and XSTREAM should be used for shorter domains, while CAST, SIMPLE and T-REKS for longer. Alternatively, fLPS can be used with CAST to identify a wide range of domain lengths. For the same purposes, GBSC can be used with SIMPLE and XSTREAM with T-REKS.

SEG is the only method that fully relies on complexity measurement when detecting domains. It uses Shanon entropy with two thresholds to find the LCR seed and expands it. This method with fLPS identified 20,035,132 residues, of which 31.1% belong to SEG only, 39.8% were found by fLPS only, and common results account for 29.0% of results. Additionally, both methods have a large overlap with repeats identified by GBSC, which are 93.5% and 95.8% for SEG and fLPS, respectively. Therefore, both methods identify STR, most probably irregular LCR consisting of several residue types and domains specific to both methods, which I will detail in the next section. The overlap between SEG and LCD-Composer is lower than between LCD-Composer and fLPS, but it is still high, providing additional insights into the

intersection of SEG and fLPS. Intriguingly, the intersection between SEG and GBA is small despite the fact that GBA uses a modified SEG metric. The SEG results cover only 28.8% of the GBA results. I already mentioned that fLPS includes fewer results from T-REKS than CAST, but more results from XSTREAM. In the case of SEG, this trend continues and the method overlaps with even fewer T-REKS results than fLPS and even more XSTREAM results. Although the LCR approach of the SIMPLE method is more similar to fLPS than SEG, the last method retains high overlap with it.

LCD-Composer identifies protein sequence fragments based on the percentage of a given type of amino acid in the window and its dispersion. Its approach to LCR detection is similar to that of CAST, fLPS and SIMPLE. It has the greatest overlap with fLPS, accounting for 78.7% of LCD-Composer results. CAST requires a score above a certain threshold to recognize the domain as LCR. This leads to different minimum LCR lengths of different residues, where they are treated equally in LCD-Composer. Therefore, CAST can skip short LCRs composed of residues with low score assigned in BLOSUM matrix surrounded by HCRs. CAST identifies a greater number of amino acids than fLPS, but includes fewer residues detected by LCD-Composer. I would expect LCD-Composer to have a high overlap with SIMPLE, as both methods examine the abundance of a certain residue type in a given window, but the first method only covers 68.3% of the residues identified by the second method. It has a slightly greater overlap with GBSC, resulting in 72.7% coverage. Due to the dispersion parameter, it omits LCRs composed of two or more short adjacent LCRs that are included in the SEG results. For instance, it will omit the ,,AAAAAAPPPP" LCR that occur in the Retrotransposon-derived protein (Q86TG7). To find only such cases, both methods can be used to exclude the LCD-Composer results from the SEG results. Lower coverage, which is less than 50%, the method has with GBA, T-REKS and XSTREAM, which is later investigated in detailed analyses. However, it is worth noting that GBA coverage is extremely low at around 16.7%.

In theory, GBA searches for repetitive domains and further extends them using modified Shannon entropy. It has the lowest overlap with other selected methods. The highest overlap it has with fLPS, but it is only 29.8%. The difference to other methods is huge, and therefore I cannot provide more insights without a detailed analysis.

XSTREAM and T-REKS identified comparable domains that similarly overlap with other methods. XSTREAM is a complex method that identifies repeat seeds and extends domains left and right according to residual similarity. T-REKS, like XSTREAM, identifies repeat seeds, but determines the final domains by comparing the distances between them using k-means algorithm. These methods aim to find TRs in protein sequences. Since the pattern of repeats can be either short or long, the overlap with LCR identification methods should be moderate. They overlap with other methods when detecting short tandem repeat, but diverge when detecting long tandem repeats. Quantitatively, the methods overlap comparably with CAST, fLPS, LCD-Composer and SEG, but differ from SIMPLE and GBSC, which are also designed for repetitions. T-REKS contains 41.2% and 30.1% fewer residues identified by SIMPLE and T-REKS, respectively. In addition, the overlap between these methods is small at approximately 47.0% of the T-REKS results. Detailed analysis is required to reveal the differences between the methods.

SIMPLE generated the fewest results. Its overlap with other methods is as follows: CAST - 85.8%, fLPS - 76.4%, SEG - 72.0%, LCD-Composer - 68.3%, XSTREAM - 48.5%, T-REKS - 46.3%, GBSC - 40.3% and GBA - 21.3%. Excluding GBA, these methods are sorted by the number of identified LCR residues, thus overlap analysis does not provide meaningful conclusions about this method. Interestingly, this method regularly overlaps with the repeat identification methods. However, the expectation is that this method should overlap more with the repeat identification methods. The characteristics of the results of this method require detailed analysis.

GBSC largely overlaps with all methods except GBA, SIMPLE and T-REKS. High overlap, that is even greater than with CAST which detected the most residues, GBSC method has with fLPS and SEG which are 95.8% and 93.5% respectively. Moderate overlap it has with CAST - 79.9%, XSTREAM - 73.8% and LCD-Composer - 72.7%. GBSC overlaps with 5 methods that cover its results above 70% It has a slight overlap with GBA and surprisingly with the repeat identification methods which are T-REKS and SIMPLE. This may be due to the fact that T-REKS and SIMPLE focus on longer domain identification, as shown in Figure 3.3. XSTREAM and T-REKS are methods for tandem repeat detection in protein sequences. Therefore, moderate and low overlap with GBSC should not be surprising. The high overlap with fLPS and SEG can lead to useful combination of these methods for consensus investigation. For example, one may identify domains using SEG and GBSC, and remove GBSC results from SEG to analyze non-repetitive, low complexity sequences. On the other

hand, when combined with T-REKS, it can identify both short and long tandem repeats.

This exploratory analysis of domains length distribution, amino acid frequency and overlap between results lead to conclusions about how each method works and how they can be combined for better results. Unfortunately, it cannot answer some important questions. For instance, ,,why is GBA overlap with other methods so poor?" or ,,why SIMPLE method does not show a striking similarity to the repeat identification methods?". Therefore, in the next section, I provide a detailed analysis of selected examples to unravel the mysteries.

3.3.2 Detailed analysis

In this section, I analyze non-obvious combinations of methods for which exploratory analysis is insufficient to fully describe them.

(A) - CAST

```
>sp|Q8R5F7|204|294|S
EDNTDLANSSHRDGPAANECLLPAVDESSLETEAWNVDDILPEASCTDSSVTTESDTSLA
EGSVSCFDESLGHNSNMGRDSGTMGSDSDESV
>sp|Q3L1C9|626|680|K
NSKKTSKKKAAELMLEELRKLPPLATPAFPRPKSKIQMNKKKNRNLIKSELQQQKA
```

(B) - fLPS

```
>sp|D2Y2E2|8|81|C
CLVWMMAMMELVSCECWSQADCSDGHCCAGSSFSKNCRPYGGDGEQCEPRNKYEVYSTGC
PCEENLMCSVINRC
>sp|Q2ILE0|127|130|P
PPPP
```

Figure 3.5: CAST versus fLPS comparison. **(A)** demonstrate two sequences identified by CAST which are missing in fLPS results. **(B)** on the other hand, shows example sequences identified by fLPS and missed by CAST. The first sequence in this panel is compositionally biased to cysteine while the second is short poly-P fragment.

CAST and fLPS are designed to detect protein fragments with a compositional bias. Taking into account that these methods are designed to identify the similar type of LCRs, there is little overlap between them. The Figure 3.5 show example

LCRs identified by either CAST or fLPS. Sequences identified only by CAST are generally out of fLPS threshold. In the first example of the panel **(A)**, serine is an overrepresented residue, but it is not the only reason why CAST tagged this fragment as LCR. It uses a scoring matrix to denote similarity between residues. Since this fragment is rich in serine and similar residues, which are threonine and analine, the alignment to the poly-S sequence is above the default threshold, and thus the method tagged this fragment as compositionally biased to serine. In the second example, we have a similar situation with lysine. This residue is overrepresented in the fragment, but there are also similar residues that increase the fragment score. On the other hand, the first example in the panel **(B)** shows a sequence where the cysteine is overrepresented, but no other residues are similar. Therefore, fLPS was able to recognize this fragment due to the number of cysteines, but CAST assigned this fragment a score below the default threshold. The second sequence from the same figure is a short poly-P fragment. CAST skipped this fragment because its score is 27, which is below the default threshold of 40. The only such short homopolymer that CAST is able to recognize is poly-W, as four runs of this residue give an alignment score of 44.

(A) - SEG

```
>sp|Q814H0|48|81|
EAQKRKEEKDAAAELENAKELKETLEKLTVELKAK
>sp|P51912|52|80|
LLVLLTVAAVVAGVGLGLGVSAAGGADALG
```

(B) - fLPS

```
>sp|Q9NRJ4|1223|1296|P
PAVVLQPLYPPSLSYCTLPPMYPGSSTCSSLQLPPVALHPWSSYSACPPMQNPQGTLPPK
PHLVVEKPLVSPPPA
>sp|Q6NZL8|767|806|CH
CEAKVHCSPGHHYNTTTHRCIRCPVGTYQPEFGQNHCISCP
```

Figure 3.6: Example protein fragments which show differences between SEG and fLPS. **(A)** presents protein fragments from fLPS results which cannot be found in SEG results, while panel **(A)**, gives two examples which were identified by SEG method, but were skiped by fLPS.

SEG is a classic approach to LCRs detection that is based on Shannon entropy. This terminology is still close to fLPS method, which identified slightly more residues in domains of low complexity. Figure 3.6 **(A)** shows the SEG results that are not present in the fLPS results. Both fragments that do not belong to the fLPS results are highly enriched in four residues. The first domain is rich in alanine, glutamic acid, lysine and leucine. In the second, the dominant residues are: alanine, glycine, leucine and valine. I overviewed the results and noticed that most of the sequences are of this kind. This is rational since fragments enriched equally in four residues are less biased. Sometimes degenerate homopolymers also appear in the results, but this may just be an artifact introduced in the latest version of the implementation, since in the previous version the method correctly identifies these fragments. On the other hand, Figure 3.6 **(B)** shows sample protein sequence fragments detected by fLPS that were missed by SEG. Both sequences are biased, but also contain a large variation of the rest residues. The first fragment is abundant in proline, but also contains residues such as valine, leucine, alanine, glutamine, tyrosine, methionine, glycine, serine, threonine, cysteine, histidine, tryptophan, glutamic acid and lysine. That is a huge number of different residues and does not fit SEG definition of LCRs. The second sequence is biased to cysteine and histidine, which are rare residues. As the first sequence, the second is rich in other types of residues. In short, the results contain many irregular domains that are equally biased to several residues, or are biased to some residue types, but also contain random types.

T-REKS and XSTREAM are methods for identifying short and long tandem repeats. However, there is small overlap between these methods, less than 50% of the results. In addition, both methods differ in the length of the identified domains, which can be seen in Figure 3.3. Therefore, I selected examples to show the differences between them. In Figure 3.7 **(A)** I present T-REKS results that were not reported by XSTREAM. The first look at the sequences may confuse someone as to whether there are any repetitions in the sequences. But if we look at these sequences from a method design perspective, we see evenly spaced seed repeats. The first sequence contains three repeats of LV residues, where the number of insertions between them is approximately equal. One may also notice the repetition of VW residues in the first two runs. The second fragment consists of ,,PLXXXAV" pattern, where X is a random residue. The review of the results also confirmed that T-REKS excludes homopolymers from the analysis, as claimed by the authors [16]. On the other hand, XSTREAM, whose results are shown in the Figute 3.7 **(B)**, generally identified short

(A) - T-REKS

```
>sp|P33972|171|189|
LVTVVWLVYPVWWLVGSEG
>sp|P69216|124|137|
PLNAFAVPLLNTAV
```

(B) - XSTREAM

```
>sp|B7IE26|415|426|
VLEKVLEDVLFE
>sp|Q8WWL7|805|869|
QEEPSIEKEAVLKEPTIDTEAHFKEPLALQEEPSTEKEAVLKEPSVDTEAHFKETLALQE
KPSIE
```

Figure 3.7: Comparison of T-REKS with XSTREAM. **(A)** shows two protein sequence fragments with repeats of LV and PL seeds respectively identified by T-REKS and missed by XSTREAM. **(B)** presents sequences skipped by T-REKS, but identified by XSTREAM. The first sequence is STR while the second is long tandem repeat.

fragments, but also identified some long repeats. An example of a short repeat is shown in the first sequence, but it is not the shortest repeat the method can identify. It also identifies simple patterns like ,,AAAA" and ,,ALALA". XSTREAM is a complex method that in later stages extends the detected domain left and right to find neighbouring part of the pattern as well. In the second sequence, we can observe this feature as it shows two repetitions and partly a third.

In Figure 3.3, which shows the length distribution of the identified domains, we can see that the SIMPLE domains strongly depend on the window size, and almost all results are focused in a single point. This results in hits that are partially out of the STR regions. For example, the Q05049 protein presented in Figure 3.8 **(A)** contains a poly-T region of 21 residues. This fragment was correctly identified by GBSC as seen in the first sequence. The second sequence is the domain identified by SIMPLE where we can also see the appended trailing residues due to the fixed window size. This additional fragment is irregular and contains only two serines out of ten residues, which are the only ones similar to a threonine. Figure 3.8 **(B)** shows repeats identified by GBSC only. The first is poly-N region. A review of the results shows that SIMPLE method skiped many homopolymers. The second is the

(A) - GBSC (1st sequence) and SIMPLE (2nd sequence)

```
>sp|Q05049|501|522|
TTTPTTTTTTTTTKATTTTT
>sp|Q05049|501|532|
TTTPTTTTTTTTTKATTTTTSGECKMEPSK
```

(B) - GBSC

```
>sp|Q54ZX8|151|167|NN-NN
NNNNNNNNNNNNNNNNN
>sp|Q2FDK5|2042|2078|SD-DS_ST-TS
STSDSTSISDSESLSTSDSDSTSTSTSDSTSGSTSTS
```

(C) - SIMPLE

```
>sp|Q23933|184|215|
PAPAPAATPLAPAPAADPPAAPVPDAAQPAI
>sp|Q06587|206|237|
GAGGSSVGTGGGGTGGVGGGAGSEDSGDRGG
```

Figure 3.8: Selected exaples showing differences between GBSC and SIMPLE methods. **(A)** shows the same fragment of Integumentary mucin C.1 protein identified by GBSC and SIMPLE respectively. **(B)** presents sequences identified by GBSC, which are missing in SIMPLE results. The first sequence is poly-N fragment. The second is short tandem repeat rich in SDT residues. And panel **(C)** shows sequences found by SIMPLE, but not by GBSC. Both sequences are degenerous repeats.

ST/SD repeat domain. This domain is 37 residues long, thus we cannot state that it is out of the SIMPLE window size. Both protein fragments undoubtedly are STRs. Figure 3.8 **(C)** on the other hand shows repeats identified only by SIMPLE method. Both sequences consist of degenerate repeats. The first sequence has several PA repeats, while the second is a highly mutated homopolymer of glycine. The distance between the repetitions makes the fragments undetectable for the GBSC parameter set used. In conclusion, SIMPLE is a method with reasonable sensitivity for STRs, but unsatisfactory specificity for their exact location, however it skips some obvious STRs.

(A) - GBSC (1st sequence) and SEG (2nd sequence)

```
>sp|E9PYH6|441|460|GG-GG
GGGGGGSGGGGGGGGGGGGG
>sp|E9PYH6|427|463|
PPASEAPPPEPPEPGGGGGGSGGGGGGGGGGGGGAPSP
```

(B) - GBSC

```
>sp|Q9GLP1|1195|1271|LS-SP_SP-PD_TL-LS_TT-TL
TLSPDLSQTTLSPDLSHTTLSPDLGHTTLSPDLSHTTLSPDLSQTTLSPDLSHTTLSPDL
GHTTLSPDLSHTTLSPD
>sp|Q8CJ78|806|848|LQ-QP_QE-EV_QL-LQ_VQ-QL
QEVSGVQLQPAQEVATVQLQPAQEVTTVQLQPAQEVTTVQLQP
```

(C) - SEG

```
>sp|Q8N697|92|131|
GGWLADARLGRARAILLSLALYLLGMLAFPLLAAPATRAAL
>sp|Q9JLA3|27|41|
LLIALALLCLFSLAEA
```

Figure 3.9: Selected examples for comparison of GBSC with SEG methods. **(A)** shows sequences identified by GBSC and SEG respectively, **(B)** fragments detected by GBSC only, and **(C)** by SEG only.

SEG results include almost all GBSC results. However, it is worth seeing examples of which regions have been identified only by GBSC and which have been omitted. In Figure 3.9 I provide examples to characterize the overlap of these methods. In the panel **(B)**, I have included two sample repeats that have been identified by the GBSC but are out of SEG complexity threshold. The pattern of the first sequence can be written using the following regular expression: TTLSPDL[SG][HQ]. The second sequence pattern is QEVXXVQLQP where X is a random residue. These patterns consist of 9 and 10 residues, which is out of SEG thresholds. The panel **(A)** shows two variants of a single protein fragment that is recognized differently by the two methods. The first fragment is identified by GBSC which identified only homopolymer of glycine with a single serine insertion. This domain, however, is surrounded by proline-rich regions that were detected by SEG and shown in the second sequence of the same panel. The last panel demonstrates sequences from

SEG results only. Both sequences consist of several types of residues, but they are arranged irregularly and lack repetitions.

(A) - GBA

```
>sp|P27799|334|362|
GFVVFSILGFMAQEQGLPISEVAESGPGLA
>sp|A2R9P6|69|99|
ALAALDAMISRMQGLKRKMENLHQEERKIQEQ
```

(B) - GBA, SEG and GBSC

```
>sp|P14043|62|92|EP-PE_PE-EP
PEPEPEPEPEPEPEPEPEPEPEPEPEPEPEP
>sp|Q54U87|33|66|QQ-QQ
QQQQLQQQQQQQQQQQQQQQQQQQQQQQQQ
```

(C) - SEG and GBSC

```
>sp|P23746|275|320|EE-EK_EK-KE_KE-EE_KE-EK
KEEEEKEKEKEKEKEEKEKEEKEKEKEEKEKEEKEKEEKEEKEEEK
>sp|P14196|302|361|NN-NN_NN-NT_NT-TN_TN-NN
NNNNNNNSNNNNNSSNNNNNNNNNSTNNNTNNNNNNNTNNNTNNNNNNINNNNNNTNNNNNN
```

Figure 3.10: Selected examples from GBA, SEG and GBSC which aims to help with GBA method characterisation. Panel **(A)** shows regions out of SEG and GBSC scope. Panel **(B)** presents results from GBA and GBSC intersection. **(C)** contains sequences which are low compexity and contain repetitions, but are not included in GBA results.

In the exploratory analysis, I found that it is difficult to characterize GBA results and a more detailed analysis is needed. Therefore, I tried to find its features and compare it with other methods. I supposed the comparison would help find the actual type of LCR that the method is capable of detecting. I compared it with SEG and GBSC methods since these methods finds well-established domains. SEG is based on Shannon entropy while GBSC finds STR in which it allows some mutations. Figure 3.10 demonstrate exapmle sequences from these methods. The sequences in **(A)** are GBA results that are missed in the SEG results. The complexity of these domains is quite high as for LCRs. The composition of the first sequence is comparable to UniProtKB/Swiss-Prot. The only over-represented residue is glycine, which occurs

in the sequence with a probability of about 0.17, while in UniProtKB/Swiss-Prot it is about 0.07. However, this difference is insufficient to call this sequence ,,low complexity". The panel **(B)** in Figure 3.10 contains the sequences reported by SEG, GBA and GBSC. These sequences consist of clear repetitive patterns. The panel **(C)**, on the other hand, shows the sequences recognized by SEG and GBSC, but ignored by GBA. I deliberately chose sequences that have repeats with mutations to show that it is still not possible to characterize GBA results. This is because GBA detects clear repeats and sequences that are questionable LCRs, but does not recognize repeats with several mutations. In conclusion, the exploratory analysis together with the study of selected examples does not provide enough information to describe the types of LCR that GBA identifies.

(A) - GBSC

```
>sp|A1IGV8|170|181|NP-PD
NPDNPDNPDNPD
>sp|P32103|109|134|AK-_KK-_SA-
SAKKASARKSSAKKPAKGGKKKSAKK
```

(B) - T-REKS

```
>sp|Q6YZW0|543|588|
QPQLVVQAQLQQPQVILQAQLQQPQVVVQAQLQQTQPSVQSHTVLQ
>sp|P29458|867|906|
FREQSSTRLDASDFEACLGAFREQSSTRLDASDFEACLGA
```

Figure 3.11: Example sequences which demonstrate differences between GBSC and T-REKS. Panel **(A)** provides sequences identified by GBSC, but not by T-REKS. Panel **(B)** shows reversed case in which T-REKS identified domains missed by GBSC.

Both GBSC and T-REKS are designed for short tandem repeats, but T-REKS is also capable of identifying long repeats. Examination of the results revealed that T-REKS omits the detection of homopolymers, which was also claimed by the authors [16]. This may partly explain the difference in the overlap between the two methods. In Figure 3.11 I present selected fragments to show the differences between these two methods with examples. Panel **(A)** contains sequences identified only by GBSC. The first sequence is a clear STR and it is interesting why T-REKS

missed it, but there are just few such cases in the results. The second sequence, however, is common in the results. It consists of a blurred SAKK pattern. Sequences identified only by T-REKS are shown in the panel **(B)**. In the first one may see several repetitions of QPQ, LQ and more generally Q The second sequence shows an example of a long tandem repeat of two patterns This detailed investigation of selected examples provides information about why the overlap between the two methods is moderate. The intersection of GBSC and T-REKS consists mostly of clear STRs excluding homopolymers.

(A) - GBSC

```
>sp|Q3IMY2|232|256|EE-
EEVIEEETEAEAVEEVAEEEIIDEE
>sp|Q5UQ16|323|338|AK-
AKAKPVAKTAKKTPAK
```

(B) - XSTREAM

```
>sp|Q26307|222|229|
QRKKQRKK
>sp|Q5GC94|50|229|
KRSAEEQNLAEDLVKRGSNKGFNFMVDMIQALSNGKRSAEEQDLAEDLVTRGSNKGFNFM
VDMINALSNGKRSAEEQDLAEDLVKRGSNKGFNFMVDMINALSNGKRSAEEQDLAEDLVT
RRSNKGFNFMVDMIQALSKGKRSAEDQDLAEDLVTRGSNKGFNFMVDMIQALSKGKRSAE
```

Figure 3.12: Example sequences which demonstrate differences between GBSC and XSTREAM. **(A)** shows sequences identified by GBSC only, while **(B)** seuqnces detected only by XSTREAM.

In order to fully characterize the identification part of GBSC method, which is introduced in the scope of this thesis, I also compare it with XSTREAM method, which is designed for TR detection in general. Therefore, in Figure 3.12 I show example domains identified by **(A)** GBSC and **(B)** XSTREAM. For both cases, results are presented that are missing in the results of the other method. In the panel **(A)**, the first sequence is a poly-E fragment containing several mutations regularly spaced throughout the fragment. The second sequence contains 5 repeats of AK residues. In conclusion, XSTREAM lacks degenerate repeats in its results. XSTREAM, on the other hand, identified many short repeats of just two runs. For

example, such a repeat is ,,HPHPH" in the kappa-casein protein (P50425) at positions 119-123. Figure 3.12 **(B)** contains another example where the ,,QRKK" pattern occurs twice. The second sequence is long tandem repeat. The length of the pattern in this sequence is 70 residues. Since both methods identify clear and low degenerate repeats, they can be used together to analyze consensus results. Consensus can be the sum to find a wide range of TR types or intersection to find clear STRs.

3.3.3 Visualisation

I visualized the LCR results of identification methods on the PlaToLoCo web service for meta-analyses. The visualization is presented in the Figure 3.13. Researchers can explore LCRs using PlaToLoCo by comparing the merged identified fragments of each method between proteins and view individual records for detailed exploration. The overview is shown in Figure 3.13 **(A)**. They can see all identified fragments along with Shannon entropy, Phobius signals, transmembranes and Pfam domains. All information is presented in a single Feature-Viewer (Figure 3.13 **(B)**), which makes it easy to compare the fragment of interest with many information in one view [66]. Since LCRs are biased to one or more residue types, one may want to see the amino acid frequency for a given protein or a fragment of it. The chart for its analysis is presented in Figure 3.13 **(C)**. It is integrated with the Feature-Viewer. By default, the chart shows the residual frequencies for the entire protein. Selecting one of the detected fragments or a custom interval in the Feature-Viewer recalculates the chart for the selection. The identification methods described in this chapter and how they overlap can be combined into a consensus method. It is visualised in Figure 3.13 **(D)** and allows to select the consensus type and methods. The available consensus types are union and intersection. The detected fragment, Pfam and PDB details are shown in Figure 3.13 panels **(E)** and **(F)**. The service where I implemented the presented visualization methods is available at https://platoloco.aei.polsl.pl.

3.4 Discussion

In this chapter, I compared existing state-of-the-art LCR identification methods, which are SEG, CAST, SIMPLE, GBA, XSTREAM, T-REKS, fLPS and LCD-Composer [9–16]. I also add GBSC to the analysis, which is described in the GBSC - clustering short tandem repeats chapter. The main expected outcome of the

Figure 3.13: LCR identification methods visualization. **(A)** list of input proteins summary with identified LCRs, **(B)** sequence details consisting of identified fragments, sequence entropy, Pfam and Phobius domains, **(C)** amino acid chart for the sequence or selected fragment, **(D)** selected methods consensus, **(E)** Pfam & PDB details, and **(F)** details about identified fragments. Source: Jarnot et al., Nucleic Acid Research, 2020.

comparison is to provide an overview of each method and to guide researchers when to use which method. In addition, it provides insight on how to combine these methods for even more ways to identify LCRs. Another outcome of this work are LCR visualization approaches to facilitate their exploratory analyses.

Comparison of LCR identification methods brought us closer to their definition. Each method provides its own definition of LCRs. Mier et al. have already made some effort to describe these fragments using an LC diagram. They also attempted to characterize LCR features using 21 different proteins containing low complexity domains and analyze them relative to disordered proteins [58]. In the same year, Cascarina et al. showed that only a complexity measure along with residue types the fragment is compositionally biased to, can provide more accurate data to conclude about domain structures [57]. This, together with the results of this chapter, leads to the conclusion that combining SEG with either CAST or fLPS may improve identification of such domains. Sequence complexity is often combined with other sequence attributes, including residual composition, hydrophobicity, charge and other [67]. Therefore, mixing two or even more methods and visualizing their results can provide more information on the biological significance of protein domains.

All methods are important for discovering the biological significance of LCRs. T-REKS, XSTREAM and GBSC can be used to detect sequences containing short tandem repeats. These domains often coexist with specific protein structures and functions. For instance, they can form a coiled coil structure or be responsible for binding through RGG boxes [68,69]. CAST and fLPS are methods for detecting bias in LCRs. This type of LCRs also plays a crucial roles in proteins. For example, the lysine-rich region is required to regulate proliferation and senescence [70]. Therefore, we cannot choose one method as the gold standard for LCR identification. It is believed that understanding how these methods work and knowing the differences between them makes it easier to choose the right method for a particular problem.

Some of the results collected by identification methods are unexpected. The main reason for including GBA in the analysis is that this method and GBSC are based on graphs. The graphs they produce are different, but I hypothesized that using a similar tool to design the algorithm for a similar problem might lead to comparable results. However, the overlap between these methods is small, and this applies not only to the overlap between GBA and GBSC but also to overlap between GBA and all other methods. Furthermore, detailed analysis did not reveal the characteristics of the protein fragments identified by the method. The method described in the article

is interesting and seems relevant to LCRs [12]. Another surprising result is the low intersection between CAST and fLPS, as with methods that detect protein fragments overrepresented by one or more residues. However, this finding shows the importance of a consensus method that can be used to find either strong compositional biases in the sequence or relaxed. The same conclusions can also be drawn from the repeat identification methods. A similar approach that use consensus of methods is TRAL. This is a Python package that combines several methods to annotate short tandem repeats in proteins [71]. However, HHrepID method, which is one of its components, is no longer available [72]. T-REKS and XSTREAM, which have been analyzed in the scope of this chapter are also part of the TRAL method. An example of the consensus method output developed in the scope of this chapter is presented in Figure 3.13 **(E)**

LCR visualization can improve the exploration of protein sequences and their potential roles. It has already been shown that the proposed LCR visualization method can, for instance, help in proteomic analysis of heat-stable proteins or phase separation [73, 74]. There are also other approaches to facilitate LCR analysis. LCR-eXXXplorer uses pre-calculated results from SEG and CAST to show them with UniProtKB annotations. However, in this service, researchers cannot submit their own sequences [75]. The recently published method LCD-Composer also has its own service where scientists can submit their sequences and download the results. But no additional visualization has been added to this tool [76].

To summarize, in this chapter I presented exploratory analysis of selected methods for LCR identification. I compared the results, and for not obvious cases I provide a detailed analysis of selected examples. I also discussed possible combinations of methods for the consensus method and finally presented a method for visual analysis of low complexity domains in protein sequences.

4 Analysis of canonical methods for protein sequence comparison in the case of LCRs

4.1 Introduction

Scientists are able to infer the functions and structures of proteins based on similar sequences with already known properties. It can speed up lab experiments by reducing the number of scenarios and can decrease their cost. Several methods for protein sequences comparison already exist. These methods are clustering and searching for similar proteins. State-of-the-art methods have two approaches to search for similar protein sequences [17, 77]. The first is based on local pairwise alignment, for instance Smith-Waterman algorithm [18]. In the second approach, the methods iteratively search for similar sequences, create MSA, which they use in the next iteration to search for more sequences. Both approaches aim to find homologous proteins. For a long time, scientists were mainly interested in HCR of sequences, considering LCRs to be junk sequences that evolved neutrally [7]. However, today we know that LCRs may play a key role in protein functions. The similarity comparison methods were designed based on statistical models primarily for HCRs, therefore the hypothesis is that they are not applicable to LCRs. In this chapter, I analyze three state-of-the-art protein similarity comparison methods in the case of LCRs to prove the hypothesis. These methods are BLAST, HHblits and CD-HIT [17, 22, 27].

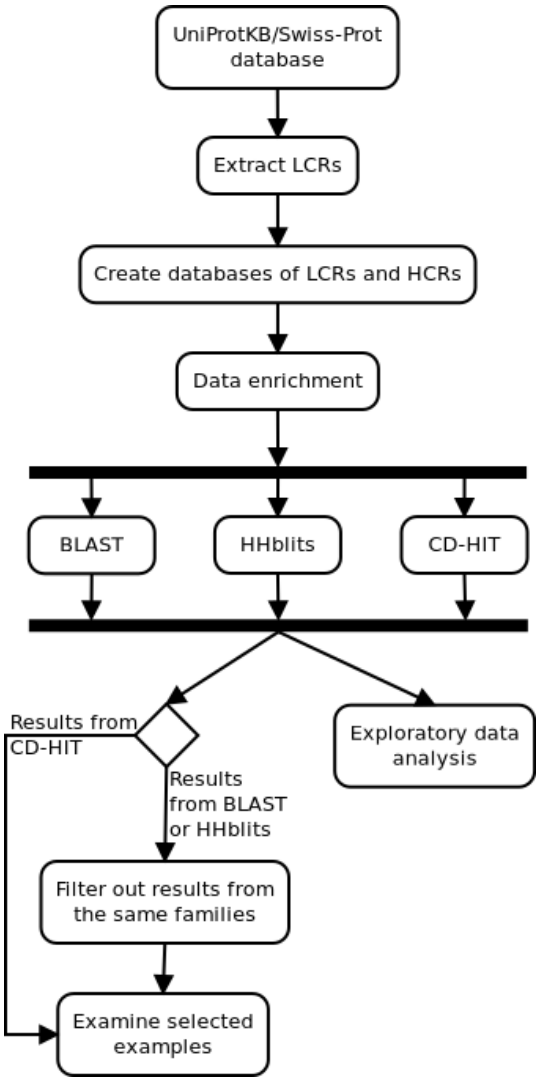

Figure 4.1: This diagram shows the workflow of this analysis. Firstly, I divide input dataset into HCR and LCR. Then, I execute all three methods on them. Finally, I explore results and examine selected examples. Source: Jarnot et al., Briefings in Bioinformatics, 2022.

4.2 Methods

4.2.1 Workflow

For the analysis of selected methods of protein sequence similarity, I used the UniProtKB/Swiss-Prot database. I extracted LCRs from the database sequences and generated two distinct datasets of LCRs and HCRs. I also enriched the FASTA-formatted sequences with information about protein families. From the datasets created this way, I generated specific databases for the methods which require it. Next, I executed all the methods to search for HCRs and LCRs. Finally, I explored results and examined selected examples to show the advantages and disadvantages of each method for LCRs similarity analysis. The workflow of the analysis is shown in Figure 4.1.

4.2.2 Data preparation

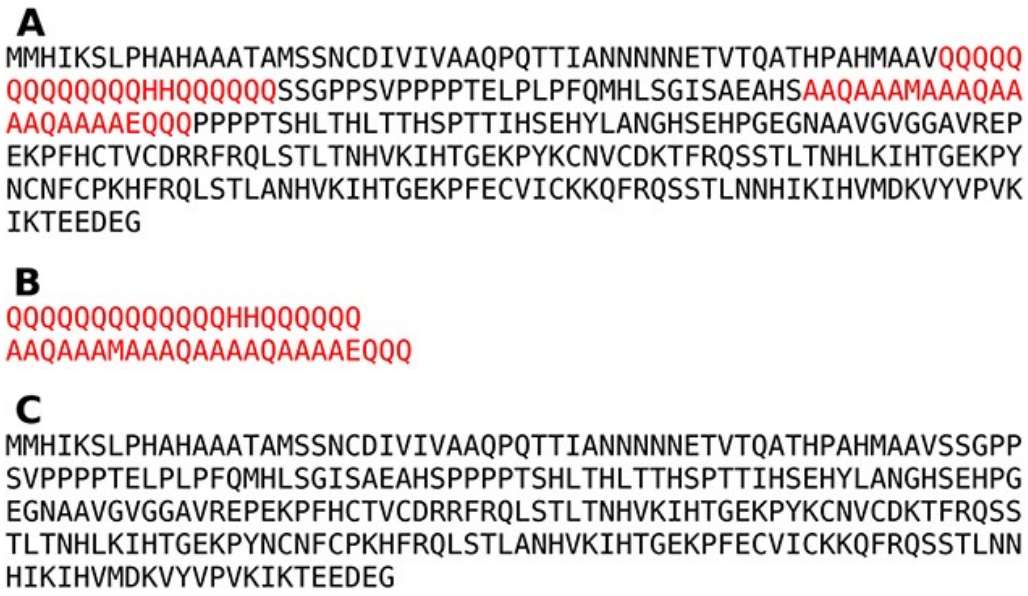

Figure 4.2: The division of example sequence into LCR and HCR parts. Source: Jarnot et al., Briefings in Bioinformatics, 2022.

I used the Swiss-Prot subset of the UniProtKB version 2020_04 as the source for the protein sequences [78]. This dataset contains high-quality and manualy curated data which I use to examine proteins in detail. Importantly, the database also ensures that protein sequences lack redundancy. In this research, I execute selected

methods for LCRs and HCRs. I use the LCR results to check whether the methods are suitable for LCRs, and I used the HCR results as a reference. Therefore, I have divided this database into HCR and LCR datasets. For this purpose, I extracted the LCRs from the sequences using the SEG method and created the LCR dataset. I then merged the remaining parts of each sequence into a single HCR and created the second dataset. As a consequence, I divided a single sequence into multiple LCRs and one HCR. An example of sequence division is presented in Figure 4.2

4.2.3 LCR extraction

Several methods have already been developed to identify LCRs in protein sequences. However, none of these methods have been established as the gold standard. For this work, I used the SEG method as it is the most popular tool, which was also used in BLAST for LCR masking [9]. It uses a sliding window to find protein fragments with Shannon entropy below a given *locut* threshold. When a fragment is identified as a low complexity, then it is exteded left and right joining residues that cause the window's entropy to be still below the hicut threshold. By design, the algorithm should be able to extend the identified low complexity fragment, thus the *locut* value should be less than the hicut. This method covers a wide variety of LCRs that are homopolymers, short tandem repeats and irregular regions containing just a few residue types. For this work, I used so-called strict parameter set (*locut*: 1.5, *hicut*: 1.8, *window*: 15), which has already been used for several biological analyses [2, 79].

4.2.4 Family assignment

Proteins are grouped into families that share similar properties and common evolutionary origins. Even though some proteins lack family assignment, the knowledge about how protein families depend on HCR and LCR can provide new insights about methods for searching for similar proteins. Theoretically, LCRs often play secondary roles in proteins, and the impact of HCRs on a protein family should be greater than that of LCRs. To check this using protein similarity search methods, I assigned families to sequences in the LCR and HCR datasets. From analyzes where this information is required, I have excluded protein sequences without family assignment. I used the families to compare the created datasets, and to check how many similar sequences found by the selected methods belong to different protein families in both datasets. We suppose that LCRs within families will have similar properities, thus

to select non-obvious cases, I used pairs of similar sequences belonging to different families for detailed analysis of selected examples.

4.2.5 Selection of similarity search methods

To check whether commonly used methods for protein similarity analyses can handle LCRs, I tested three state-of-the-art methods for three different types of similarity analysis. These methods are BLAST, HHblits and CD-HIT [17, 22, 27]. BLAST is the most popular tool for protein similarity searching based on local alignment. HHblits is the choice when more distant evolutionary relationships between proteins are needed. CD-HIT has been successfully used to cluster sequences into families and remove redundancies from protein databases. Using this analysis, conclusions can be drawn about a wide range of protein similarity analysis tools as they have similar statistical models. For example, using HHblits results we can infer about HMMER while BLAST results can be used to infer the FASTA method [44, 80].

4.2.6 Input dataset preparation

Before running the selected methods, I pre-processed the input datasets as required by each method. I downloaded the input dataset in FASTA format, which is one of the standard protein sequence formats. This format stores sequences in plain text, which simplifies manual investigation of sequences, but is inefficient in terms of disk space usage. Indexing in this format is not supported, thus most similarity search methods require input dataset in their own format, which can be easily generated from FASTA.

BLAST calculates High Scoring Parts of sequences, and thus requires converting FASTA formatted dataset to its specific format which is indexed and significantly increases performance.

HHblits performs an HMM-HMM comparison to report similarity between sequences. Since it compares HMM profiles, which are a condensed form of MSA, it would require high computing power to compute the profiles on the fly. Therefore, the database of HMM profiles is built before searching. The dedicated way to build the database is to use the available pipeline which uses MMseqs to create clusters of similar sequences which are then converted to HMM profiles [81]. It is worth mentioning that creating MSA for LCRs has an issue with aligning the LCR in the right position. The issue is shown in Figure 4.3 and is known as shift error [82].

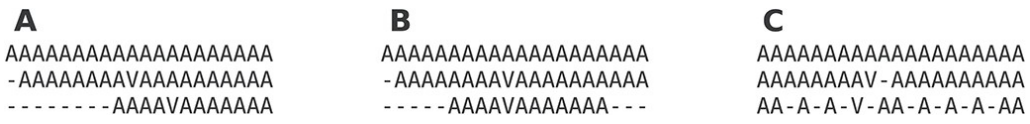

Figure 4.3: MSAs created by **(A)** MUSCLE, **(B)** Kalign and **(C)** Clustal Omega. MSA of LCRs can be created in several ways. LCRs comes from proteins of the following UniProtKB AC: Q9V727, D3ZKD3 and Q5BGE2, respectively. These MSAs are used to create profiles of HMMs for HHblits. Source: Jarnot et al., Briefings in Bioinformatics, 2022.

For LCRs, the pipeline needs to adjust the parameters. MMseqs is a method for protein sequence clustering and is indirectly used in this research to build the database for HHblits. Like other methods, it was designed to analyze similarity between proteins, focusing on HCRs. Therefore, I changed the *mask* and *comp-bias-corr* parameters. The first parameter changes the masking strategy and is responsible for masking the LCRs. The second, corrects locally biased composition of residues. As a result, the pipeline generates three databases for three different identity thresholds, which are 10%, 20% and 30%. I chose Uniboost 30, which is created with the highest identity threshold. The rationale for this choice is that LCRs are not yet well-characterised, thus closer similarity should be investigated first.

4.2.7 Parameter adjustment

All methods are designed to analyze the similarity between HCRs. Therefore, in the case of LCRs, I adjusted the parameters for all methods, while for HCRs I left the default parameters. Since they are intended to reduce the significance of the LCRs, I changed them to get more results with reasonable quality. To adjust them, I read their description to select the best candidates and confirmed the selected parameters by looking at the results with and without changes. Frequently used statistic to assess the evolutionary significance between protein sequences is the E-value. Intuitively, it tells how many times one may expect the same or better alignment simply by chance. I set its value to 0.0001 in BLAST and HHblits, as this value has been used in many biological analyses [83, 84].

Additionally, I changed the following BLAST parameters: *max_target_seqs*, *task* and *comp_based_stats*. By default, BLAST reports 250 best hits, which is sufficient

in most cases. In this work, I searched for similarities to all sequences, thus I turned off this filter by setting its value to the highest possible which is 1,073,741,798. This should include results for proteins that have many similar sequences. One of the BLAST parameters that has a large impact on the results is *task*. It accepts the following values: *blastp*, *blastp-fast* and *blastp-short*. This is a meta parameter that affects other parameters by setting them to the optimal values for a given task. *blastp* is the default value, *blastp-fast* changes parameters to speed up calculation and *blastp-short* optimizes parameters for short sequences. For LCRs, I used the last task since these regions are often short, as can be seen from the length distribution figure 3.3. *blastp-short* changes the default scoring matrix to PAM30 [85]. It also affects the parameters related to gaps by setting their open cost to 9 and extension cost to 1. To skip potentially dissimilar sequences, BLAST uses pre-filtering, which is disabled in *blastp-short* task. This option also changes E-value, which is however explicitly overwritten in this study to 0.0001. The next parameter is *comp_based_stats*. LCRs are known to cause false positive homology hits due to their striking similarity between non-homologous proteins. Therefore, in the previous versions of BLAST, it used SEG to mask the LCRs. Currently, by default it recalculates the scoring matrix by increasing frequently occurring values and decreasing values of residues that are rare in the query sequence. In conclusion, this greatly diminishes the importance of LCRs that I investigate in this study, thus I just disabled the feature. The analysis of these parameters is presented in the ,,LCR-BLAST - searching for low complexity regions" chapter.

In the case of HHblits, when analyzing the LCR dataset, I changed the following parameters: *noprefilt*, *sc*, *norealign*, *diff* and *id*. All these parameters affect the number of results and/or their quality. HMM-HMM comparison is a computationally complex process, and to speed up the search, HHblits filters HMM profile pairs that are considered dissimilar. Unlike HCRs, for LCRs this pre-filter rejects many true positives, thus I disabled it with the *noprefilt* parameter. The method provides various equations for calculating the score, which can be changed using the *sc* parameter. I analyzed all the possible values and evaluated their quality and quantity of results in order to select the appropriate equation. As a result, I chose the equation described by the Formula 4.1.

$$S = \log_2 \sum \frac{t_{ja} * q_{ia}}{p_a} \tag{4.1}$$

Where $t_j(a)$ is the template score at j position, $q_i(a)$ is the query score at position i, and $p(a)$ is the background frequency [22]. HHblits also uses MAC algorithm to handle repeats in proteins [72]. We classify repeats in proteins as tandem and non-tandem or short and long, where short repeats are an integral part of the LCRs. The MAC algorithm uses TMscore, which according to the authors of the algorithm, incorrectly evaluates repeat units shorter than 15 and is intended for highly divergent protein repeats [72]. Disabling this algorithm for LCRs with the *norealign* parameter increases the number of results reported by the method. The *id* and *diff* parameters control query MSA growth by filtering out very similar sequences. Since LCRs are often very similar to each other, I turned off these filters.

In contrast to previous methods, CD-HIT lacks of parameters to control fragments of low complexity. The only thing I set for LCR analysis is the minimum length of accepted sequences. By default, l is set to 10. I changed this value to the minimum, which is 4 residues. Another parameter that may be useful for LCR analysis is s, which controls the maximum length difference between incoming and representative sequences. In this work, I omit this parameter since it greatly reduces the number of results and introduces other cases where two sequences differing in the length of only one residue belong to different clusters. I described the analysis of all changed parameters and their impact on the results in the ,,Results section.

4.2.8 Methods execution

After preprocessing the data and adjusting the parameters, I ran the selected methods. Selected methods can be used to analyze various aspects of similarity between protein sequences. BLAST and HHblits are search algorithms, while CD-HIT is a clustering algorithm. To make the results comparable, I unified the output of all methods. It consists of similar pairs of sequences. To achieve this, for BLAST and HHblits, I ran methods on each sequence in the LCR and HCR datasets. I then created pairs of similar sequences by combining the query sequence with all the sequences reported by the methods as similar. For CD-HIT, I created similar sequence pairs by combining all sequences belonging to the same cluster.

4.2.9 Results analysis

To explore the results, I calculated the number of similar pairs identified by each method, the overlap between their results, and I plotted the relationships between

alignment features. I analyzed the LCR and HCR datasets for two cases. The first case applies to all generated results, and the second excludes sequence pairs where both sequences belong to the same family. This visual and statistical exploration of the results provides an overview that can be used to draw some conclusions and formulate new hypotheses. Because data exploration only provides an overview of the results, I additionaly analyzed selected examples in detail to draw new conclusions and confirm those already drawn. For the analysis of selected examples, I chose these that come from different families for BLAST and HHblits to show more interesting cases. For CD-HIT, I used all the results since the insights are generic and family independent. I also analyzed the results, paying attention to the features of methods that work very well on the HCR dataset, but prevent effective LCR similarity analysis. At the end of next section I show how I selected parameters used for LCR analysis of HHblits and CD-HIT. BLAST parameters I describe in detail in the next chapter.

4.3 Results

I investigated the results of each method using an exploratory analysis approach and looking at selected examples. As an exploratory analysis, I summarized the number of results, created a Venn diagram, compared the dependence of E-value on alignment length, number of alignments and sequence identity using line plots. As part of selected sample studies, I showed which kind of LCRs are handled by the methods correctly using the proposed set of parameters. I also provide ill-matched examples to show the features of each method that need improvement. I examined all cases from a compositional point of view to see whether the LCRs are properly matched according to their definitions, and for selected examples provided their biological roles.

4.3.1 Results in numbers

The proportion of results collected from each method varies between HCRs and LCRs as can be seen in Table 4.1. HHblits is a tool designed to sensitively search for similar sequences using HMM-HMM profile comparison. This tool identified the largest number of similar pairs in the HCR dataset, representing more than 75% of the total number of similar pairs identified by all methods. Next is BLAST, which identified about 16% of all results. The method uses Smith-Waterman algorithm to

	HCR	LCR	HCRs without same families in pairs	LCRs without same families in pairs
BLAST	3,205,592 (15.83%)	11,507,921 (71.58%)	46,413 (0.65%)	4,550,663 (67.56%)
HHblits	15,296,119 (75.55%)	4,331,254 (26.94%)	7,096,205 (99.32%)	2,105,748 (31.26%)
CD-HIT	1,745,171 (8.62%)	237,782 (1.48%)	2,477 (0.03%)	79,705 (1.18%)
Total	20,246,882 (100%)	16,076,957 (100%)	7,145,095 (100%)	6,736,116 (100%)

Table 4.1: Total number of similar sequence pairs found by all three methods. HHblits found 4.77x more alignments than BLAST for HCRs. For LCRs it is different and HHblits found only 0.38x of BLAST results size. In CD-HIT removing pairs with sequences from the same family remove almost all results from HCRs, but for LCRs it removes only 66%. Source: Jarnot et al., Briefings in Bioinformatics, 2022.

report alignments, thus it reports closer sequence similarities than HHblits. CD-HIT cluster sequences with the highest probability of redundancy, thus it identified the fewest number of sequence similarities. It identified almost 9% of all results. For LCR results, the proportion is different. HHblits idnetified 27%, BLAST 71.5% and CD-HIT only 1.5%. BLAST found many more similarities than other methods. This is a huge difference between HCR and LCR results. For CD-HIT, the small number of LCR results compared to HCR results is due to the algorithm that assigns sequences to clusters. It creates a separate cluster for the sequence, even if it can be assigned to an already existing cluster containing similar sequences. This behavior is discussed in more detail in the section devoted to the analysis of selected CD-HIT examples. A significant difference between the size of both dataset results can be observed when we remove pairs of similar sequences whose proteins belong to the same family. We can explain the HCR results by referring to the design assumptions of each tool. BLAST is a tool designed to find homologous sequences, therefore it reduces the ratio of similar sequence pairs from 16% to 0.5% when removing sequence pairs where both sequences belong to the same family. CD-HIT is used to remove redundancy in protein databases and group proteins into families. Taking these assumptions into account leads to the conclusion that the decrease from 8.6% of the results to slightly above zero after removing pairs belonging to the same family is also reasonable. HHblits is designed to search for distant homologies. Unfortunately, the risk of finding non-homologous sequences increases with sensitivity. These false positives explain why HHblits results contain almost all similar pairs belonging to different

families for HCR dataset. However, the above explanations are not applicable for LCRs. The ratio of the number of results before and after removal of pairs of similar sequences assigned to the same family is similar. Changes in the ratio of all methods is small. This suggests that protein families are irrelevant to LCR similarity.

4.3.2 Overlap analysis

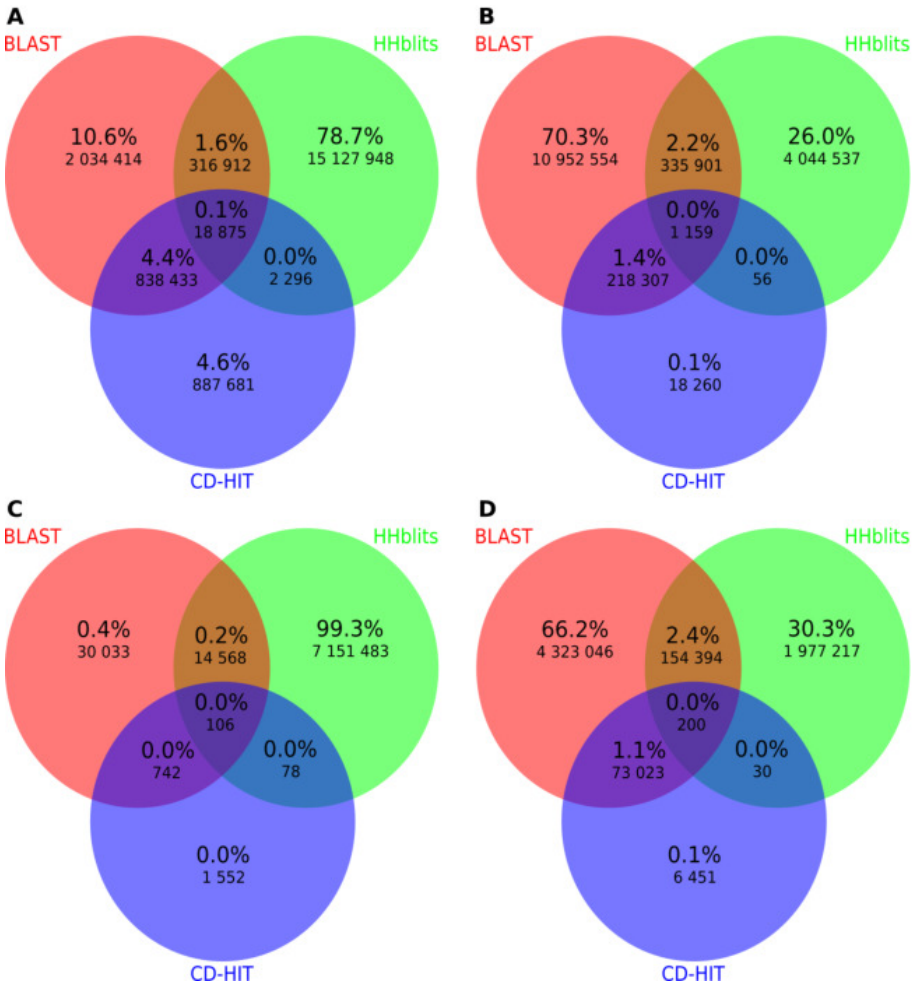

Figure 4.4: Venn diagrams which show overlap of methods. The intersection of all the methods is small. Common similarities can be seen only between BLAST and CD-HIT. **(A)** shows results for HCRs, **(B)** for LCRs, **(C)** for HCRs without pairs where sequences come from the same families, **(D)** for LCRs excluding sequence pairs from the same families. Source: Jarnot et al., Briefings in Bioinformatics, 2022.

HHblits found unique results, while BLAST and CD-HIT share some similar sequence pairs. Figure 4.4 presents Venn diagrams that help in analyzing the overlap of selected methods. Researchers often use different methods of protein similarity analysis to see whether the results overlap. For the purpose of increasing the chance of similar protein properties, they can check whether the two methods find the same proteins as similar, or if the results of one method are not satisfactory, they can use other methods to find more similarities. The intersection of all the selected methods is slight, suggesting that these methods are intended for different purposes. The overlap between HHblits and CD-HIT is extremely small. As a matter of fact, these two methods have two opposing goals. The first method searches for distant similarities and even filters out the most similar ones, while the second method detects redundancy in databases by focusing on very similar sequences. BLAST with HHblits also share a small number of results for both LCRs and HCRs. However, I would consider these methods in terms of complementary rather than different design purpose. Indeed, BLAST is capable of finding similar proteins, but HHblits finds more distant similarities, thus researchers looking for variety of similarities would use the results of both methods. The only noticeable overlap is seen between BLAST and CD-HIT. For the HCR dataset, almost half of the CD-HIT results are also included in the BLAST results. For LCRs, however, this proportion changes and almost all CD-HIT results are included in BLAST. In the case of HCRs, when removing sequences from the same family, all overlaps disappear, leaving only the sequences identified by HHblits. At the same time, the LCR overlap keeps similar proportions. This confirms the above claims.

4.3.3 E-value and alignment analysis

E-value behaves differently for HCRs and LCRs. In order to illustrate the differences, I will refer to Figures 4.5 and 4.6. We can see that the E-value in the HCR results varies significantly between BLAST and HHblits. In the BLAST results, we may encounter many sequences with high E-value and fewer but still significant numbers of lower E-value results. In the case of LCRs results are concentrated near the highest E-value because these domains are mostly short and E-value is length dependent. For $log(evalue)$ below -75, HCR results of BLAST cover about 66.39% while for LCRs it is only about 0.01%. From charts (b, c, e and f) we can read that LCR alignments are shorter, but on average they consist of more identical residues. The high deviation in

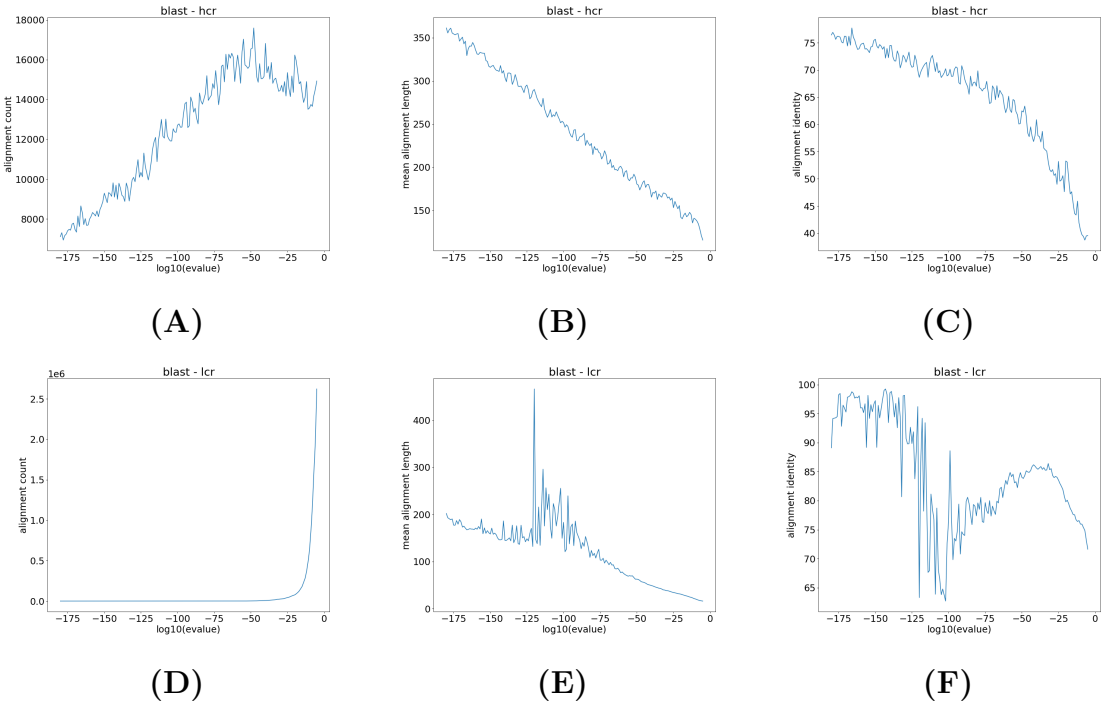

Figure 4.5: Relationship between E-value and alignment count **(A, D)**, mean length **(B, E)** and the number of identical residues **(C, F)** for **BLAST** for HCRs **(A, B, C)** and LCRs **(D, E, F)**. When BLAST E-value is out of float range, the method assigns 0 which is skipped in this figure.

the LCR plots below -75 $log(evalue)$ is due to low number of alignments. For HCRs, as the E-value decreases, the identity increases logarithmically. On the other hand, in the LCR results, in the alignments for $log(evalue)$ belog -30 is higher than in HCR, but it is starting to decrease. In the case of HHblits, presented in Figure 4.6, the results for both datasets, most of the results are assigned to the higher $log(evalue)$. For HCRs, most of the $log(evalue)$ of the results are above -100 while for LCRs they are mostly above -30. The alignment identity and length increase logarithmically as $log(evalue)$ decreases. In the case of LCRs, the identity behaves the same as in the case of HCRs. The chart of alignment length has significantly lower values for LCRs in comparison to HCRs, which is justified by their lengths. On the other hand, LCRs for the same E-value are more identical.

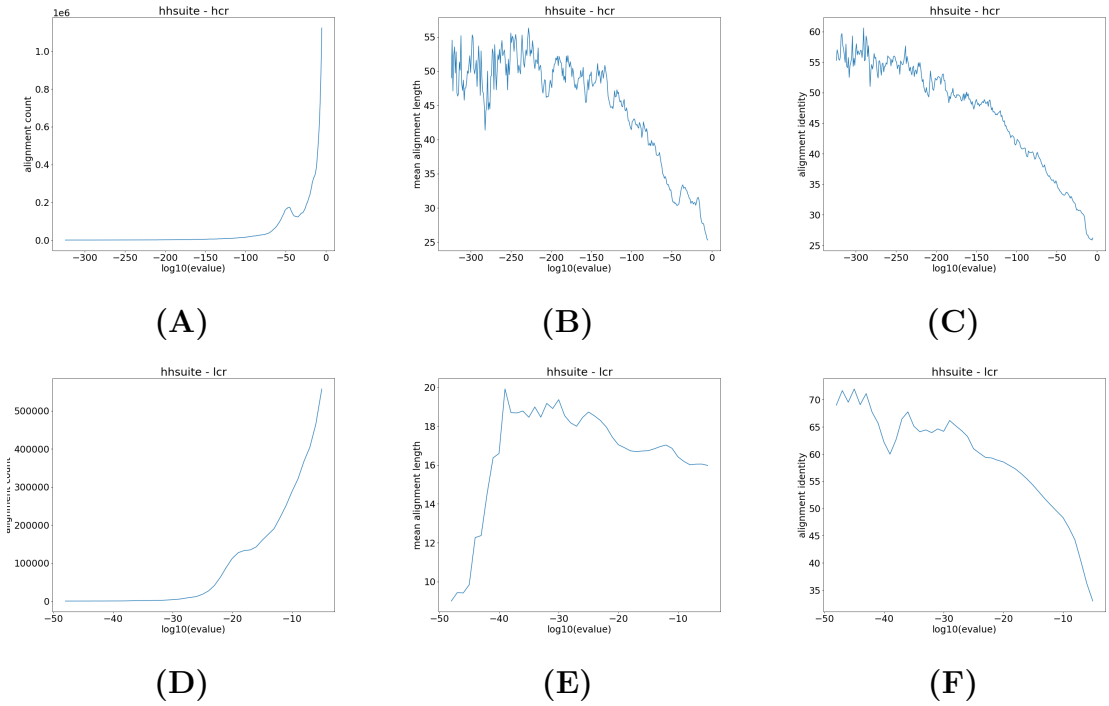

Figure 4.6: Relationship between E-value and alignment count **(A, D)**, mean length **(B, E)** and the number of identical residues **(C, F)** for **HHblits** for HCRs **(A, B, C)** and LCRs **(D, E, F)**.

4.3.4 BLAST example alignments

BLAST assigns an E-value to alignments that inconsistently evaluate differences in sequence lengths and their types. Local pairwise alignment provides a similar E-value when both sequences are of the same and different lengths. Figure 4.7 shows these issues in the examples. In the alignment **(A)** both sequences are poly-A and their lengths are the same. However, in **(B)** the aligned sequences are poly-H of the same length, but the E-value of both alignment vary significantly. The E-value of the first alignment is $3.8 * 10^{-6}$ while the E-value of the second alignment is $3.5 * 10^{-12}$. The difference is introduced by different scores in the BLOSUM scoring matrix. Even if this helps to find evolutionary related sequences, it cannot be used to demonstrate the relevance of different LCR types, for instance when clustering LCRs using MCL [49]. In the example **(C)** both sequences are also poly-H domains, but their lengths are different. The first sequence (G3V7L5) is shorter than the second (Q1XH10). However, the E-value is very similar to the previous case, being about $4.7 * 10^{-12}$. The biological similarity between proteins in the alignment **(A)** is

(A)

```
P31361  103  AAAAAAAAAAAA  114
             AAAAAAAAAAAA
Q2MJS2  151  AAAAAAAAAAAA  162
E-value: 3.79259e-06
```

(B)

```
G3V7L5  174  HHHHHHHHHHHH  185
             HHHHHHHHHHHH
P97838  222  HHHHHHHHHHHH  233
E-value: 3.49068e-12
```

(C)

```
G3V7L5  174  HHHHHHHHHHHH  185
             HHHHHHHHHHHH
Q1XH10  337  HHHHHHHHHHHHHHH  351
E-value: 4.70105e-12
```

Figure 4.7: Sequence alignments generated by BLAST. The figure shows relationships of sequence lengths and E-value..

that the first (P31361) is transcription factor protein while the second (Q2MJS2) may acts as a transcriptional regulator [86, 87]. The first sequence of **(B)** and **(C)** alignments (G3V7L5) is responsible for targeting signal to nuclear speckles [88] The protein of the second sequence of **(B)** alignment (P97838), among others, may act as an adapter protein linking ion channel to the subsynaptic cytoskeleton [89]. Even if the exact role of the poly-H fragment is unknown, these fragments are known as binding to ions which may be crucial to the function of the protein [90]. The second protein of the last alignment (Q1XH10) is not well annotated.

4.3.5 HHblits example alignments

HHblits reported similarity between distant sequences. Figure 4.8 shows example alignments collected from HHblits. The first example is 22 amino acids long and the

(A)

```
>consensus_sp|Q8ST25|5f229009737a9faef0170d8a
Probab=99.77  E-value=1.7e-24  Score=79.71  Aligned_cols=22
Q Q54FP8            1 NNKNNKNNNNNNNNKTNKNNNN   22 (22)
Q Consensus         1 ~~~nnsnnnnnn~n~nn~nnnn   22 (22)
                      ||||+||||||||||+|+||
T Consensus         1 ns~ns~nnnnn~snn~~~n~nn   22 (22)
T consensus_sp|Q    1 NSNNSNNNNNNNSNNNNSNSNN   22 (22)
```

(B)

```
>consensus_sp|Q7ZY40|5f229009737a9faef016a8ed
Probab=98.72  E-value=5.4e-13  Score=45.14  Aligned_cols=17
Q Q9EST5            1 DEDEEDEDEDEDEEEEE   17 (17)
Q Consensus         1 eEeeeeeeeeeeeeeee   17 (17)
                      |++||+||||||+||+|
T Consensus         1 ed~~e~eee~~ede~de   17 (22)
T consensus_sp|Q    1 EDDEEGEEEEEEDEEDE   17 (22)
```

(C)

```
>consensus_sp|A5E203|5f229009737a9faef016fdfa
Probab=98.74  E-value=4.1e-13  Score=48.24  Aligned_cols=15
Q P29029            8 SSGSTSSGSTSSDST   22 (22)
Q Consensus         8 SSsssSssssSssSs   22 (22)
                      ||.|+|||+||||||
T Consensus         2 sSds~ssS~~ss~~s   16 (19)
T consensus_sp|A    2 SSDSDSSSDSSSSSS   16 (19)
```

(D)

```
>consensus_sp|Q920D3|5f229009737a9faef016edb9
Probab=98.87  E-value=8.1e-14  Score=49.70  Aligned_cols=15
Q E9PZQ0            6 EPPKKTPPPPPPKKE   20 (21)
Q Consensus         6 ~pp~p~~P~Ppp~~~   20 (21)
                      ||||||||||.+|-|
T Consensus         2 ~pp~pppp~pg~pgq   16 (16)
T consensus_sp|Q    2 PPPPPPPPPPGLPGQ   16 (16)
```

Figure 4.8: Sequence alignments found in HHblits results. **(A)** and **(B)** are common with BLAST. **(C)** and **(D)** were identified by HHblits only.

rest are 15 amino acids long. Alignments **(A)** and **(B)** were also reported by BLAST, while **(C)** and **(D)** were reported only by HHblits. These reported by both methods have higher similarity and identity than alignments reported by HHblits alone. The identity of **(A)** and **(B)** is approximately 59% and 60%, respectively. This is higher than in alignments **(C)** and **(D)**, which are 53% and 40%, respectively. The similarity also follows this pattern. It is 77% and 93% for the first two alignments, and 67% and 47% for the last two. Similarity in mutations is indicated by a positive score of Viterbi algorithm [22]. The above observations may lead to the conclusion that HHblits reports more distant similarities, some of which are common to the BLAST results. However, HHblits evaluated these pairs of sequences differently, consistent with Hidden Markov Model profiles. Therefore, the profile used to construct the **(D)** alignment reported better similarity accoding to E-value and score than the **(B)** alignment.

4.3.6 CD-HIT example clusters

In the case of CD-HIT, I analyzed clusters using all sequences without excluding these belonging to the same families. The CD-HIT method requires a different approach to similarity analysis than BLAST and HHblits since it is a protein sequence clustering method. It is easy to exclude pairs of similar sequences belonging to the same family, but it is more difficult when considering a group of similar sequences. In addition, the number of CD-HIT results is relatively smaller compared to other methods. Therefore, I decided to analyze the results of this method without filtering out sequences belonging to the same family.

The way CD-HIT evaluates similarity and length between sequences cause assignment of similar sequences to different clusters and different sequences to the same cluster. In Figure 4.9 I have shown examples that help to demonstrate the issues. In the first example, the representative sequence consists of sub-LCRs, which may also exist separately. These sub-LCRs are mainly poly-A and poly-G sequences. Therefore, when CD-HIT will search the cluster for a poly-A or poly-G LCRs, it is highly possible that these sequences will join this cluster as shown in **(A)**. Such representative sequences consisting of adjacent LCRs are common since they are longer than atomic LCRs. This wrong assignment is caused by an alignment method that only compare LCR to a part of the representative sequence consisting of sub-LCRs. The example **(B)** presents two similar sequences of the same length

(A)

```
>Cluster 301
0   71aa, >sp|E9Q4N7|5f229009737a9faef016a8a3... *
GAGGGGGGGGGGGGSGGGGGGGGAGGAGGAAAAAAGAGAVAAAAAAAAAAAAAAGGGGGG
GYGSSSSGYGV
1   15aa, >sp|P35453|5f229009737a9faef016a941... at 93.33%
AAAAAAAAAAAAAAA
7   17aa, >sp|O77215|5f229009737a9faef016ad58... at 94.12%
GGGGGGGGGTGGGGGGG
```

(B)

```
>Cluster 264
0   75aa, >sp|Q1ZXH2|5f229009737a9faef016aabe... *
QQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQCQPQQQQTQQQQQQQQQQQQQQQQQQQQ
QQQQQQTQQQQQQPQ
>Cluster 271
1   75aa, >sp|Q54DK4|5f229009737a9faef0170790... *
QLNQQQQHLQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQNQQQNQQQNQ
QQQQQQQQQQQQQQN
```

(C)

```
100 63aa, >sp|Q54ZP8|5f229009737a9faef016c96d... at 90.48%
NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNTNT
NNN
188 232aa, >sp|Q54L50|5f229009737a9faef016e962... *
NNNGNNLNNSNNNNNNNNSNSSSNNNNNNSSSNNSSSNNNSNNNNINNNNNNGNNNNNMG
NNNNNMGNNNMGNNNMGNNNMNNNNMNNNNINNNNSNNNNNNNMNNNNNSNNNSNNMNNN
NNNSNSNNNNNNNNNNNNNNNNMNNNNMNNNNNNNNNNMNNNNNNNNNNNNNNNNMN
NNNNMNMNNNMNNNNMNNNNNNNMNNNNMNNNNMNNNNMNNNNVNNNNMINN
123 18aa, >sp|Q55DD4|5f229009737a9faef016d095... at 100.00%
NNNNNNNNNNNNNNNNNN
```

Figure 4.9: Two cases which show wrong assignment to clusters by the CD-HIT method. **(A)** The cluster contain sequences of different LCR types.**(B)** Highly similar sequences belong to different clusters. **(C)** The cluster contain sequences of which the length differ.

that form separate clusters. The statement about similarity between sequences is supported by biological properties because both sequences create the same domain which is coiled-coil in probable serine/threonine-protein kinase fhkB (Q1ZXH2) and alpha-protein kinase 1 (Q54DK4) proteins. Since the sequences are similar, they form the same structure and both sequences are kinases that most likely should join the same cluster. This example shows that sequences of similar length have lower chance to create the same cluster due to several random mutations. On the other hand, they are more likely to be joined by a short poly-Q fragment if it will not be joined by a longer poly-Q fragment beforehand. This case is unexpected, because a significant difference in the length of the sequences may realize different functions [91]. The difference in length is shown in the example **(C)** where three poly-N LCRs with significantly different lengths were assigned to the same cluster.

4.3.7 HHblits parameters analysis

For the LCR analysis, I adjusted the HHblits parameters, while for the HCRs I left them unchanged, since the method is designed to detect the similarity between HCRs. In this section, I propose a set of parameters that can be used for LCR analyses. I also provide a number of results and example alignments illustrating their quality. To adjust them, I first read the method manual and selected parameters that potentially affect LCR similarity. Then I ran the method with the full set of parameters and sets, of which I removed a single one. This approach helps to assess whether a parameter has a positive or negative impact on LCRs.

The *diff* parameter introduces MSA diversity in the results. This feature improves the search for more distant similarities in HCR results. However, LCRs are still poorly investigated and I disabled the parameter to get more similar results instead. Another parameter that limits the most similar alignments is *id*. It filters out the specified percentage of alignments with the best score from the analysis to discard the most obvious cases. However, in the case of LCRs, it is not known how distant similarities may share biological properties, thus I set its value to 100. Changing *diff* and *id* to their proposed values increase quality of reported alignments. Figure 4.10 presents examples of alignments that were removed when I applied these parameters. Both alignments matched weakly similar sequences. Because the similarity is unclear and we do not yet know whether this kind of similarity also leads to functional or structural similarities. The first example relies on the similarity

#	Parameter set	Missing parameter in comparison to #1	Number of similar pairs
1*	-id 100 -diff 0 -norealign -sc 0 -noprefilt		4 331 254
2	-diff 0 -norealign -sc 0 -noprefilt	-id 100	4 357 676
3	-id 100 -norealign -sc 0 -noprefilt	-diff 0	4 354 196
4	-id 100 -diff 0 -sc 0 -noprefilt	-norealign	623 205
5	-id 100 -diff 0 -norealign -noprefilt	-sc 0	4 002
6	-id 100 -diff 0 -norealign -sc 0	-noprefilt	2 309 843

Table 4.2: Number of alignments reported by HHblits for given parameter sets. * indicates used parameter set. Source: Jarnot et al., Briefings in Bioinformatics, 2022.

between asparagine and serine. The similarity between these two residues was scored with value 1 according to BLOSUM62. The second example relies on the similarity between glutamic acid and lysine, which also has a similarity value of 1 according to BLOSUM62. Regarding the Table 4.2, these parameters do not affect the number of reported alignments.

```
NNNNNNNNNNNSNNNSS
+|+.+|+++++|+|.||
SSSGGSGNSSGSSSRSS
```

(A)

```
EELEEEEELEEEEEELGED
.|+|+.||+|++|++++.||
KEKEKKKDKEKKEKKRKRED
```

(B)

Figure 4.10: Example alignments which were removed from results after changing -*diff* 0 **(A)** and -*id* 100 **(B)** parameters. Source: Jarnot et al., Briefings in Bioinformatics, 2022.

Parameters that increase the number of similar pairs are *norealign*, *sc* and *noprefilt*. The first disables MAC algorithm, which is used in addition to Viterbi algorithm for HMM-HMM profiles comparison [22] [72]. Disabling the MAC algorithm increases the number of alignments by approximately 695%. This is a huge number and rationale behind it is that, according to authors, the MAC algorithm wrongly handles STRs which are a large portion of LCRs. In the set of rejected alignments by the algorithm, we can see such alignments as in Figure 4.11 **(A)**. In this alignment, we can notice some matches of glutamic and aspartic acids. In addition, we see substitutions from

aspartic acid to glutamic acid, which are scored with a value of 2 by BLOSUM62 and are considered highly similar. The *sc* parameter determines how the score is calculated. Available approaches consider rare amino acids as these which are harder to reproduce by chance and therefore the assigned score obtained from their matches should be higher [24]. However, LCRs are naturally composed of a few amino acid types and rarely occuring residues in these domains are rather considered as random mutations. Therefore, lowering the score obtained from frequently occurring residues underestimates the LCR alignment. For this reason, I changed the score calculation to a basic approach based on background frequencies. This change significantly increased the number of reported alignments by 108,227%. Figure 4.11 **(B)** is an example alignment where both sequences are similar. The last changed parameter is *noprefilt*. Pre-filtering steps have been added to HHblits because HMM-HMM comparison is a highly computationally complex and fast method that is able to estimate whether two HMM profiles are likely to be similar. If the probability of two profiles being similar is low, the match is ignored and the similarity is rejected. As it is shown in the Table 4.2 and in Figure 4.11 **(C)** the number of results increases while keeping their high quality.

```
ELEDDRDDDDDDD        SSSSSSLSSSSSNSM      AAVAAAPVAADAAPAA
||||++||||+||        ||||||+||||||||      |+||||||.|||||+|
EEEEEEEDEEKDD        SLSSSSIKSGSSSSS      AEKAKAAALAAAAADA
     (A)                  (B)                  (C)
```

Figure 4.11: Example alignments which were added to the results after changing *norealign* **(A)**, *sc* **(B)** and *noprefilt* **(C)** parameters. Source: Jarnot et al., Briefings in Bioinformatics, 2022.

4.3.8 CD-HIT parameters analysis

I adjusted CD-HIT parameters to increase the number of results when clustering LCRs. In this method, most of the parameters are irrelevant to sequence complexity, and finally I changed only one parameter. As in the case of HHblits, I provide the number of similar pairs generated by CD-HIT when changing parameters and analyzing the quality of selected examples.

#	Parameters	Number of similar pairs
1*	-l 4	237 782
2	default	206 367
3	-l 4 -s 0.7	32 039

Table 4.3: The number of similar pairs generated from CD-HIT clusters. * indicates used parameter set. Source: Jarnot et al., Briefings in Bioinformatics, 2022.

This method has two parameters that can be changed when clustering LCRs. These parameters are l and s. The first specifies the minimum sequence length for clustering. LCRs are short sequences, as it can be seen in Figure 3.3. Disabling this filtering is not supported, and therefore I just set it to the minimum value. Setting this parameter joined short sequences to already existing clusters. The s parameter limits the maximum difference between sequences and cluster representative for comparison. For example, if the value is set to 0.7, the maximum difference between the representative sequence and other sequences is 70%. Even though this solves the issue of large length differences in clusters, which can lead to different biological functions, it also assigns two highly similar sequences to different clusters and therefore significantly decrease number of similar pairs (Table 4.3). An example of highly similar sequences that were assigned to different clusters is presented in the Table 4.4. The LCR identified in the sequence from anaphase-promoting complex subunit 4 protein (Q54NI1) and in probable serine/threonine-protein kinase tsuA protein (Q54NI1) are very similar, differing only in the length of one residue, but were joined to different clusters.

4.4 Discussion

The analysis shows that BLAST, HHblits and CD-HIT are methods designed to compare HCRs, and they need improvements to properly handle similar LCRs. By default, BLAST and HHblits underestimate the role of LCRs, significantly decreasing the score assigned to frequently occurring residues. Disabling LCR masking significantly increases the number of results collected by these methods. However, other parameters and algorithms are also optimized for HCRs including local alignment and general purpose scoring matrices. I have already proposed parameters that can be used for LCR analysis. In this section, I will discuss the

Cluster	Sequence	UniprotACC
1	QQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQ	Q559R1
	QQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQ	Q498D1
	QAQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQ	Q4P209
	QQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQ	Q54NI1
	QQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQ	P0CB66
	HQQQQQHQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQ	Q551F8
2	QQKQQQQQQQQQQQQQQQQQQQQQQQQQQQHQQQQQHQQQQQHQQQQQHQQQQQQ	Q6CPW4
	QQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQ	P58462
	QQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQ	Q55FT4

Table 4.4: Example clusters from the CD-HIT method which are highly similar to each other.

characteristics that cannot be changed with parameters and that are core of the methods.

General-purpose scoring matrices, which were generated from wide range of MSAs, may cause misleading comparison of alignments of different LCR types and are not suitable for LCR analyses. Examples of such matrices available in the BLAST tool are BLOSUM and PAM. They were generated from MSA of HCRs, thus they mainly indicate whether substitutions of one amino acid for another occur frequently in evolutionarily related sequences. These should not be applied to LCRs since these fragments have different structural and functional preferences than HCRs [55] [92]. For instance, LCRs are often associated with disordered regions, which can be seen in H-rich and P-rich regions. On the other hand, they can also form secondary structures such as alpha-helices in the A-rich and L-rich

regions [57]. The use of specialized scoring matrices, which are optimized for a particular class of domains, is recommended whenever possible. For example, when analyzing transmembrane proteins, the PHAT matrix developed by Ng et al. can be used because transmembrane proteins are known to contain LCR [93] [94].

HMM profiles are a condensed form of MSA, which are difficult to create for LCRs. MSAs are used to perform phylogenetic analyzes based on the evolutionary relationship between proteins. However, LCRs often evolve neutrally, and the same LCR types can be found in different proteins and play different roles [95] The alanine-rich fragment found in two *streptococcus gordonii M5 antigen I/II* proteins (SspA and SspB) interact differently with salivary glycoproteins [96]. In addition, LCRs are often impossible to align without knowledge of their evolution since we lack the ancestral sequences [97]. Figure 4.3 presents a case with multiple possibilities to align seuqences, especially the first sequence to the others. Such alignment errors, in the literature are called shift errors [82]. Unfortunately, MSA creation is an open problem and is even present in HCRs [98]. Hubley et al. used blastx to show that the consensus between automatically generated and manually curated MSAs differs [99]. For their work, they used selected sequences from UniProtKB/Swiss-Prot. These evidences confirm that MSA should be manually curated whenever possible, and external study is needed to assess usefulness of MSA methods for LCRs.

The E-value helps to determine whether two sequences are evolutionarily related, therefore it is irrelevant to LCR comparison. This metric, by definition, indicate how often one may see a similar hit by chance. A lower value means that similar and better matches are less likely to occur or, to put it differently, higher probability that the hit is evolutionary related. In the case of LCRs, the number of similar LCRs of one type differs significantly from the number of LCRs of another type. Furthermore, as shown in Figure 3.4, LCRs have different frequencies compared to the database. These findings suggest to mainly focus on other metrices including score and identity, when comparing LCRs.

Local pairwise alignment and HMM-HMM profile comparison used in BLAST and HHblits produce perfect alignments when the hit sequence is longer than the query and both are of the same LCR type. As seen in Figure 4.7 **(b, c)**, increasing the length of the hit sequence only slightly changes the E-value introduced by the target length in the equation 5.1. Local alignment accurately searches for similar domains between two highly complex sequences. Nevertheless, the LCRs are already local sequence fragments and thus it should be considered to compare them globally. This

conclusion is supported by studies showing that LCRs can gain their functions if they reach a certain length [100]. For example, homopolymer of glutamine is the root cause of neurological disease when its length exceeds the threshold [91].

CD-HIT clusters are based on representative sequences, and therefore, the method forms clusters where the LCR sequences are different. The method parses the database sequences, starting with the longest, and compares them to the longest sequence in clusters. If the algorithm detects a similarity, it appends the sequence to the cluster or otherwise creates a new cluster. To determine whether the sequences are similar or not, CD-HIT aligns a shorter sequence to local fragment of a longer sequence. In this approach, sequences in clusters may not be similar to each other. This scenario is presented in Figure 4.9 **(A)** where the homopolymer of alanine and glycine are in the same cluster. Another issue with comparing similarities in CD-HIT may be that it joins locally similar sequences that differ significantly in length. This can be corrected with the s parameter, as shown in the CD-HIT parameter analysis section. Unfortunately, the use of this parameter is a dilemma because it greatly reduces the number of similar sequences and appends two sequences that differ by only one residue to different clusters. An example is presented in the Table 4.4.

This chapter shows that existing methods are not efficient enough to search for similar LCRs. I ran the analysis for LCRs and HCRs. For LCRs, I adjusted the parameters to increase the number of results and their quality. I explored the results by creating the Venn diagram (Figure 4.4) of the results of selected methods, and by analyzing the relationships between the characteristics of the methods. I showed problematic examples of selected alignments and clusters. Next, I discussed issues related to the features of methods that cannot be improved with their parameters. The discussion led to the conclusion that these LCR comparison methods should be improved or new ones should be developed for this purpose. Even if this study covers analysis of only three canonical tools, the other tools are often based on solutions used in selected methods.

5 LCR-BLAST - searching for low complexity regions

5.1 Introduction

Proteins may follow the rule of transitivity. In other words, two similar sequences from two different proteins may share similar properties. If we have two similar proteins, one of which is well described, then we can use it to hypotesize about function of the second protein or its fragments. It is helpful to have many similar sequences if we want to discover the biological role of a protein without functional annotations. Then we can significantly reduce the cost and time of wet lab experimental methods to determine its real role. In this way, the search for similar protein sequences can help discover new functions and structures. BLAST is the most popular tool for finding similar protein sequences. For a long time, scientists focused mainly on high complexity parts of protein sequences. We know from the previous chapter that BLAST is designed to detect similarities in HCRs. In this chapter, I propose LCR-BLAST, which is able to search for similar LCRs more efficiently.

5.2 Methods

In this study, I analyze the parameters introduced in the previous chapter and add new suggestions to improve BLAST. Each parameter is discussed with a hypothetical influence on LCR results. For convenience, I provide the following notation for each set of parameters:

- DEF-BLAST – default parameters
- SHORT-BLAST – switched *task* parameter to *blastp-short*
- COMP-BLAST – switched off compositional based scoring matrix recalculation

- SHORT-COMP-BLAST – switched *task* parameter to *blastp-short* and switched off compositional based scoring matrix recalculation
- LCR-BLAST – switched *task* parameter to *blastp-short* and switched off compositional based scoring matrix recalculation. Additionally, simplified scoring matrix and introduced mean score.

5.2.1 Short sequences

The average protein length in the UniProtKB/Swiss-Prot is 361 residues. Low complexity regions, however, are short parts of protein sequences and the average LCR, according to SEG, is 24 residues long. The *task* is a meta-parameter that adjusts other parameters to be optimal for different types of searches. Possible values for *task* are: *blastp*, *blastp-fast* and *blastp-short*. *blastp* represents the default parameters, *blastp-fast* is a set of parameters optimized for searching large datasets, and *blastp-short* optimizes for short sequences. The last value, which I used in LCR-BLAST, changes the gap open and extension penalties to 9 and 1 respectively. It also changes the scoring matrix to PAM30, E-value threshold, word size to 2 and clear filter options.

5.2.2 Compositional based statistics

Composition-based scoring matrix recalculation was introduced to BLAST as a response to the many false positive hits caused by LCRs. This parameter by default recalculates the scoring matrix based on amino acid frequencies in the query sequence. It decreases the score of frequently occurring residues and increases the score of rare residues considering them biologically important [19]. Since LCRs consist of several types of residues, their significance is reduced. To effectively analyze LCRs, I disabled this parameter in COMP-BLAST, SHORT-COMP-BLAST and LCR-BLAST.

5.2.3 Mean score

BLAST calculates E-value as the main statistic for comparing protein sequences. The following equation describes how to calculate it:

$$E = (n * m)/(2^{S'}) \tag{5.1}$$

where n is the total number of residues in the database and m is the length of the query sequence. S' is a bit score that shows the similarity between two protein or nucleotide sequences based on the selected scoring matrix. The main principle of E-value is to estimate whether two sequences share a common ancestor. Intuitively answers the question: How many times one may expect by chance an alignment with similar or better score? Therefore, the smaller the value, the more likely the two sequences share a common ancestor. However, a particular type of LCRs can often be found in proteins being important to their function [101]. Therefore, the bit score is more informative for fragments of low complexity than the E-value. The bit score is the normalized version of the raw score resulted from alignment methods and can be described by the following formula.

$$S' = (\lambda * S – ln(K))/ln(2) \tag{5.2}$$

In this equation, λ and K are used to make the normalized score independent of the parameters used to create the scoring matrix and gap penalties. Finally, the bit score can be compared between alignments created using different parameters. Short LCRs often play an important role in proteins [102]. But when analysing similar sequences, the bit score reports hits in the following order. First, it reports long identical sequences to the query. It then reports sequences with few mismatches, and then it decreases the alignment length as the number of mutations in the long alignments increases. A researcher who analyses low complexity domains may be interested in shorter perfect matches instead of highly mutated long matches. Therefore, I propose mean score which makes the alignment score independent from its length, and divides the bit score by its length. The equation for this metric is as follows:

$$M = S'/L \tag{5.3}$$

where M is the mean score, S' bit score, and L is the alignment length. To verify this equation, I compared it with the results collected using E-value.

5.2.4 Identity scoring matrix

Commonly used scoring matrices for alignment-based methods have been developed to better assess how to align sequences considering their residuals and evolutionary relationships. All 20 residues are not equally similar to each other. For instance, glutamic acid is very similar to aspartic acid because both are negatively charged.

They differ from lysine and arginine, which are positively charged. Therefore, scoring matrices assign positive values to residues that have similar properties and often mutate among themselves in evolutionarily related proteins. A negative score is assigned to pairs of residues that are different and non-interchangeable in evolutionarily related proteins. We have already discussed that LCRs may be responsible for an important protein function without being homologous. Therefore, common matrices such as BLOSUM and PAM should not be used for LCR analyses as they are general purpose matrices for HCRs [92]. In this work, I introduce an identity scoring matrix and compare it with the results collected with BLOSUM62 and PAM30 to see if a general purpose scoring matrix for LCRs is needed.

5.2.5 Workflow

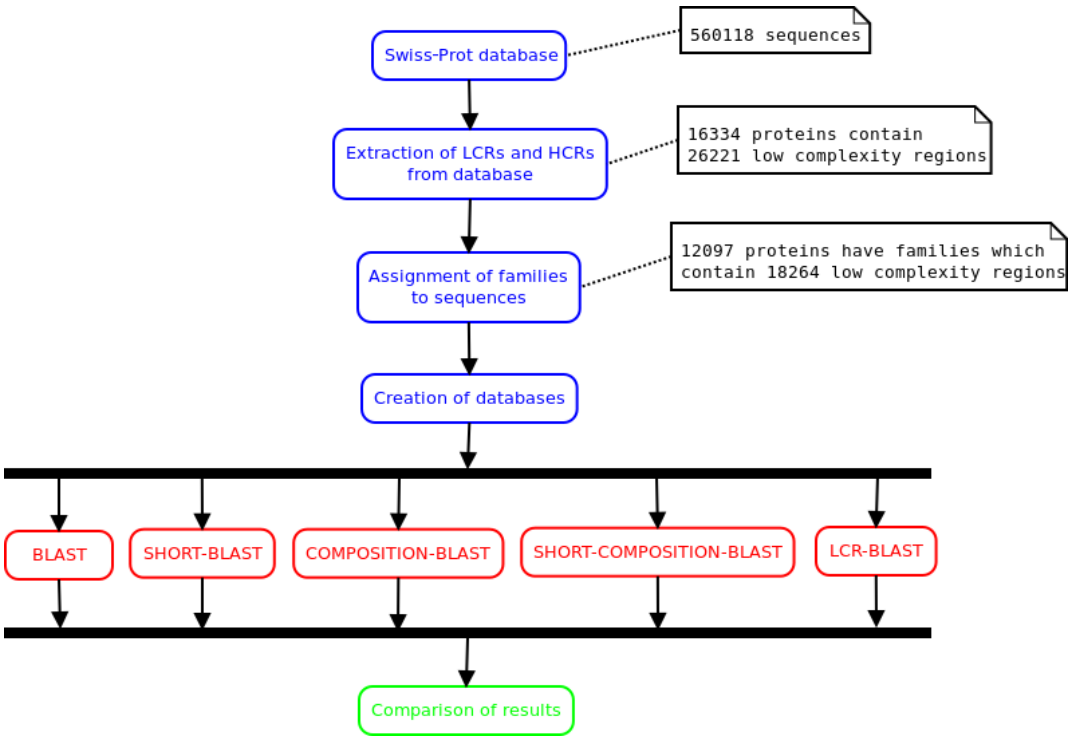

Figure 5.1: Workflow of the analysis. Firstly I prepared dataset (blue), then ran each BLAST modification (red) and finally compared and analysed results (green). Adapted from: Jarnot et al., International Conference on Man-Machine Interaction, 2020.

The workflow for BLAST analyses is similar to the workflow for comparing canonical methods from the previous chapter. Therefore, I review and discuss the

differences between them. I downloaded UniProtKB/Swiss-Prot version 04_2019 as input dataset. To extract the LCRs from the database sequences, I used the SEG tool with a strict set of parameters (*window*: 15, *locut*:1. 5, *hicut*: 1.8) [2, 79]. BLAST requires a database in its format, thus I converted the LCR dataset to BLAST specific format. Then I ran all BLAST modifications on each extracted LCR as a query sequence. Finally, I analyzed the results using an exploratory approach and examination of selected examples. Unlike the workflow in the previous chapter, I omitted running these modifications on the HCR dataset since I adjusted BLAST for LCRs and I considered these modifications not applicable to HCRs.

5.3 Results and analysis

In this section, I compare all the modifications made to BLAST for LCR similarity comparison. For an exploratory analysis of these results, I use Figures 5.3, 5.4 and 5.5. All of these figures help to quantify each BLAST modification and check their relationships.

5.3.1 Data exploration

LCR-BLAST is the only modification that follows the LCR length distribution. To demonstrate this, I use Figures 5.2 and 5.3. The first shows the number of LCRs identified by SEG of a given length. We can see that the LCRs are short fragments and most of them contain between 5 and 40 residues. This is a huge difference compared to HCR parts, which are highly deviated. The second figure shows distribution of BLAST alignment lengths for each modification. We can see that most of the methods draw a flat line when we put them on the same plot with LCR-BLAST. By analyzing the LCRs and alignment lengths, we can notice that the LCR-BLAST alignment distribution follows the pattern drawn by the LCR distribution. Interestingly, if we look at COMPOSITION-BLAST and SHORT-COMPOSITION-BLAST, we can see that the second modification line touches the first and is high for short alignments where the COMPOSITION-BLAST line is low. To conclude, *blastp-short* parameter set works for LCRs as expected.

The overlap of method modifications determines whether they have increased the number of results and whether they produce distinct results. Overlap between BLAST variants occur when when two or more methods have aligned the same LCR

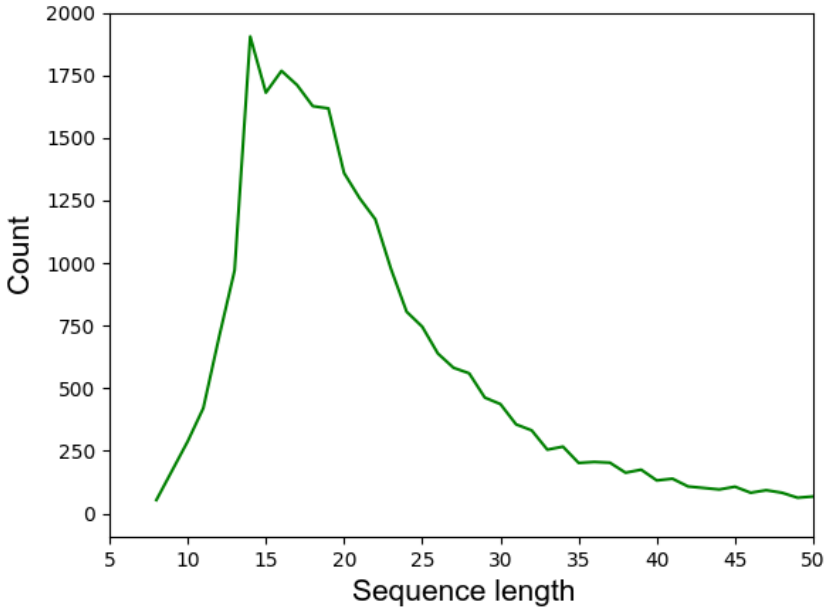

Figure 5.2: The chart shows number of LCRs by their length. Most of the LCRs fall into interval between 10 and 30 residues. LCRs were identified using SEG with strict parameters (*window*: 15, *locut*: 1.5, *hicut*: 1.8). Source: Jarnot et al., International Conference on Man-Machine Interaction, 2020.

regardless of how they were aligned. Venn diagram is shown in Figure 5.4. The number of similar pairs identified by each modification is as follows: BLAST - 5,260 sequences; SHORT-BLAST - 19,749; COMPOSITION-BLAST - 922,561; SHORT-COMPOSITION-BLAST - 2,123,087 and LCR-BLAST - 3,354,493. The total overlap contains 3,337 pairs, which is approximately 63.4% of all pairs identified by BLAST with the default parameter set. This value is quite high and reasonable as all results were calculated using alignment approach that is the core of BLAST. Interestingly, if we remove SHORT-BLAST from consideration, the number of similar pairs increases to 4,805 (91.3% of BLAST). The difference between BLAST and COMPOSITION-BLAST is 917,301 pairs. Therefore, the number of results increased by about 17,539% after turning off the scoring matrix recalculation. Knowing that this recalculation aims to filter LCRs, the fact that disabling the parameter results in the greatest increase in the number of results is expected. Changing the *task* parameter set to *blastp-short* increased the number of results by 14,489 (375%), while adding it to COMPOSITION-BLAST resulted in 1,200,526 (230%) new pairs. Quantitatively, this

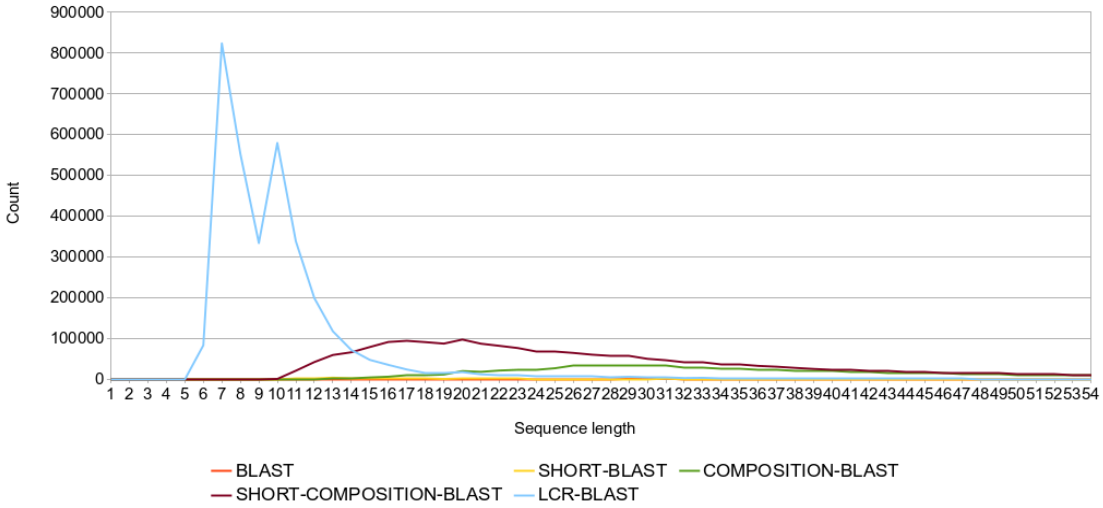

Figure 5.3: The chart shows distribution of alignment lengths found by BLAST modification. For LCR-BLAST, we can observe a pick in the alignment lengths characteristic also to LCR lengths, while other modifications are deviated. Source: Jarnot et al., International Conference on Man-Machine Interaction, 2020.

meta-parameter has a similar effect on BLAST as it has on COMPOSITION-BLAST. Because both SHORT-BLAST and COMPOSITION-BLAST increased the number of results compared to BLAST itself, thus SHORT-COMPOSITION-BLAST gives the largest number of pairs I was able to achieve by changing only the parameters. If we only look at SHORT-COMPOSITION-BLAST and LCR-BLAST, the percentage of pairs reported by LCR-BLAST that are missed by SHORT-COMPOSITION-BLAST is almost 55.6%, for LCR-BLAST about 29.8% and the overlap contains 14.7% pairs. Therefore, both BLAST modifications identified more pairs than overlap with the other method. Even though LCR-BLAST found more sequences than SHORT-COMPOSITION-BLAST, the second modification found 29.8% of similarities, in other words approximately 1,422,063 pairs. In the ,,Selected examples analysis", I characterize similar pairs belonging to these results.

Figure 5.5 shows the improvements of each approach applied to the BLAST method with default parameters. **(A)** shows results with *blastp-short* parameter set, **(B)** after turning off composition correction, **(C)** shows both changes and **(D)** comparison to LCR-BLAST. In **(A)** 31.6% $(1,661/(1,661+3,599)*100\%)$ of alignments were missed in SHORT-BLAST results. This is the highest value among all the presented overlaps, which suggests that the *blastp-short* parameter set increases the number of reported LCRs, but also detects different LCRs characteristics. This metaparameter changes

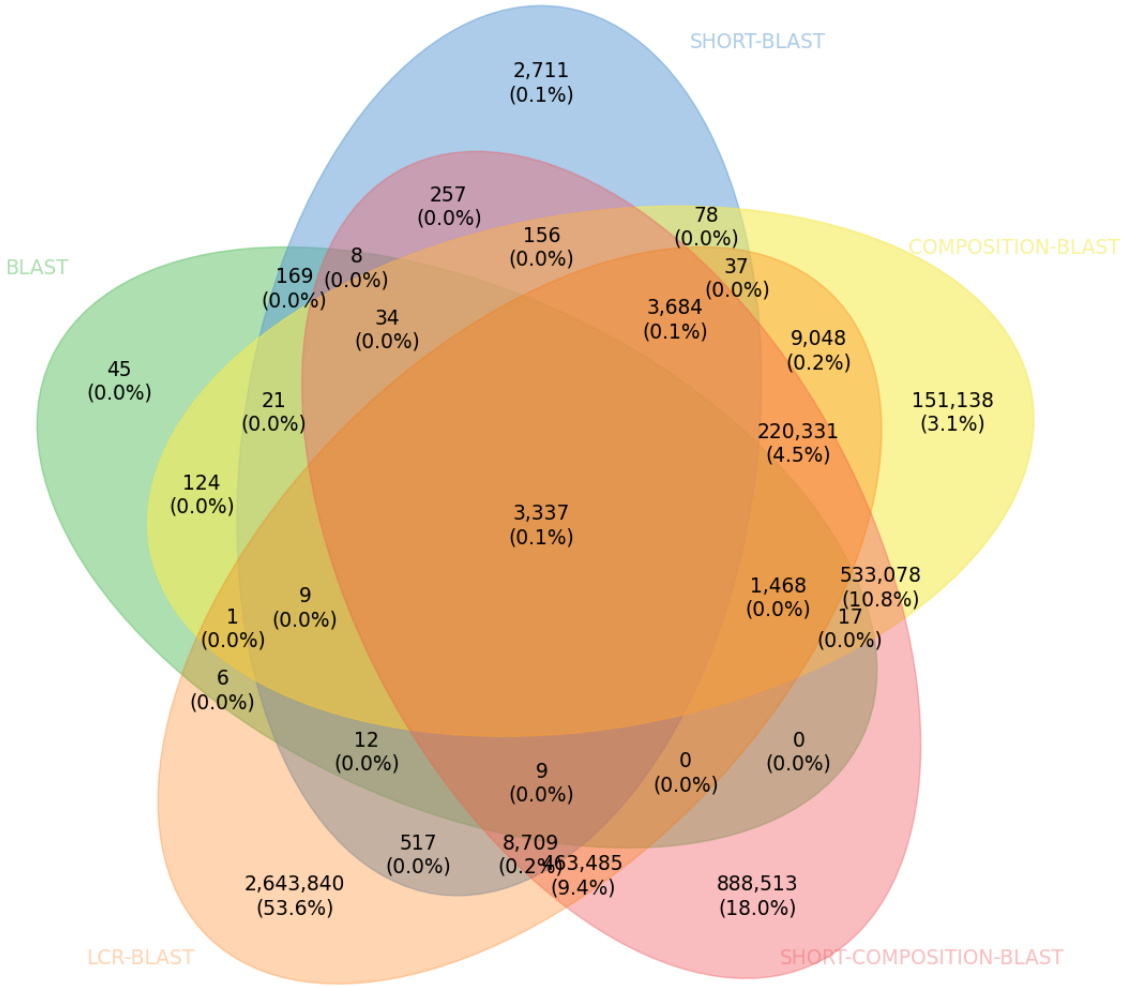

Figure 5.4: Venn diagram of overlap between each modification results. LCR-BLAST identified the highest number of results which partially overlap with other methods. COMPOSITION-BLAST and SHORT-COMPOSITION-BLAST identified many alignments missed by other methods. Detailed analysis is needed to check whether these alignments are significant. Adapted from: Jarnot et al., International Conference on Man-Machine Interaction, 2020.

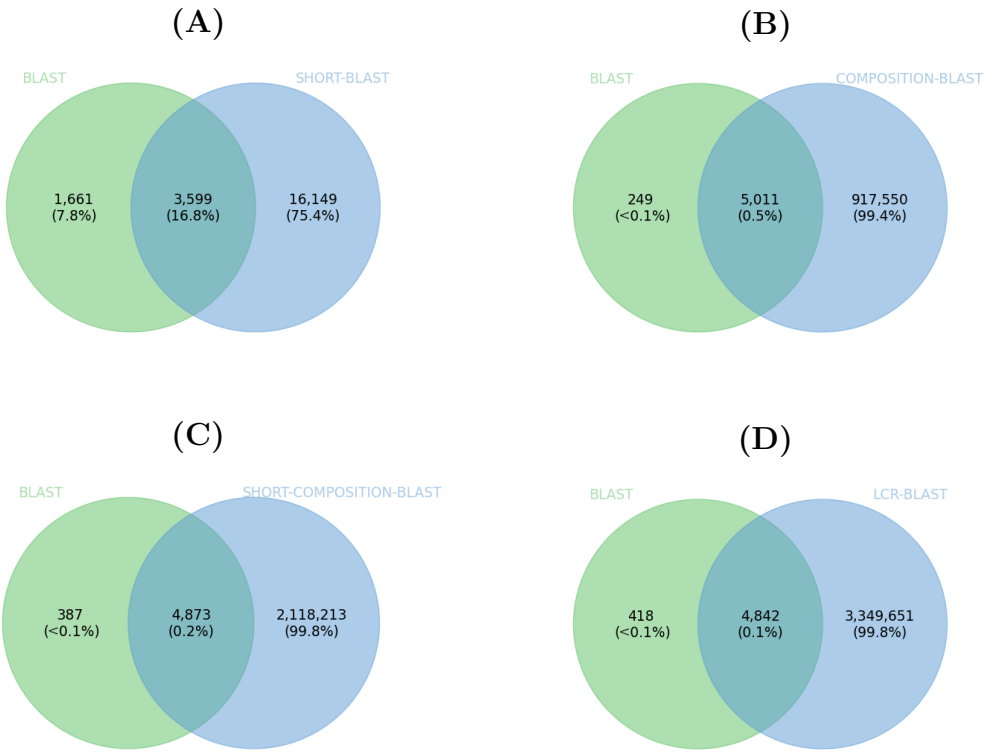

Figure 5.5: Overlap between BLAST and all modifications. BLAST is efficient for HCR searches, however, all presented modifications to this method improves searching for similar LCRs. Adapted from: Jarnot et al., International Conference on Man-Machine Interaction, 2020.

gap penalties and scoring matrix. COMPOSITION-BLAST, shown in **(B)**, has the highest common part with BLAST. Only 4.7% $(249/(249+5,011)*100\%)$ of BLAST alignments are missed if we disable scoring matrix correction using composition based statistics. Changing the default parameters to *blastp-short* set, increases the number of missing alignments as shown in Figure 5.5 **(C)**. The reason is the same as in the **(A)** diagram, but has less impact due to the number of alignments. LCR-BLAST from the panel **(D)** missed more records from BLAST which is due to the greater sensitivity to mutations in longer sequences.

5.3.2 Selected examples analysis

In this section, I show the most interesting examples of alignments collected from all BLAST variants. To simplify the examples, I used alignments in which the

query (upper) protein is CD2 homolog (P0C9V8). This protein was found in the *African swine fever virus* and is present in the results of all modifications. The protein has several functions. Among others, it can cause speed up replication by suppressing the immune system and cell penetration in the host organism [103] [104]. As examples, I chose alignments of very similar sequences and alignments where sequence similarity is debatable. The total number of alignments to this protein is as follows: BLAST (5), SHORT-BLAST (2), COMPOSITION-BLAST (271), SHORT-COMPOSITION-BLAST (57), LCR-BLAST (5).

```
P0C9V8  PCPPPKPCPPPKPCPPPKPCPPPKPCPPPKPCPPPKPCPPPKP
        P  P KP  P KP  P KP  P KP  P KP  P KP  P KP
P16053  PASPEKPRTPEKPASPEKPATPEKPRTPEKPATPEKPRSPEKP

P0C9V8  PCPPPKPCPPPKPCPPPKPCPPPKPCPPPKPCPPPKPCP
        P  +P P P+P P  +P    +P P P+P   P+P P P
Q2US45  PASRPTPKPQPLPQSQPATA-QPIPAPQPAAVPRPVPQP

P0C9V8  PNPCPPPKPCPPPKPCPPPKPCPPPKPCPPPKPCPPPKPCPPP
        P PCPPPKPCPPPKPCPPPKPCPPPKPCPPPKPCPPPKPCPPP
P0C9V6  PKPCPPPKPCPPPKPCPPPKPCPPPKPCPPPKPCPPPKPCPPP
```

Figure 5.6: In the first two examples BLAST with default parameters aligned LCR from CD2 homolog protein (P0C9V8) to sequences with low identity and similarity. These alignments are similar only in terms of composition bias. It also identifies almost perfect matches as presented in the last example. Source: Jarnot et al., International Conference on Man-Machine Interaction, 2020.

BLAST with default parameters found 5,260 alignments, which is the lowest number of all proposed modifications. Figure 5.6 shows three of the five identified alignments to the CD2 homolog protein (P0C9V8). The first compares the reference protein with the neurofilament medium polypeptide protein (P16053) found in *Gallus gallus (Chicken)* organism. In the organism, this protein is involved in maintenance of neuronal caliber and may help to create neuronal filamentous networks [105]. The first sequence in the alignment (P0C9V8) consists of KPCPPP repeats. The second sequence is also made of repetitive pattern, but different which is PEKP[R/A]T. It is possible to align at most three residues in both patterns, thus the identity of two

sequences composed of these repeats is about 50%. In the second alignment, the query is aligned to the myosin-1 protein (Q2US45), which is a motor protein involved in generating tension in the muscles. The aligned part of the protein sequence is an irregular LCR rich in prolines. The midline shows that only the prolines were matched to some other residues as similar. In conclusion, such sequences were aligned only because both are rich in prolines. The last alignment is almost a perfect match to the sequence that comes from the same protein sampled at a different time and place. In summary, BLAST with default parameters aligned repeats with the same unit length but a different pattern and compositionally biased residues of irregular LCRs.

```
P0C9V8  PKPCPPPKPCPPPKPCPPPKPCPPPKPCPPP
        PKPCPPPKPCPPPKPCPPPKPCPP KPCPPP
P0C9V7  PKPCPPPKPCPPPKPCPPPKPCPPSKPCPPP

P0C9V8  PKPCPPPKPCPPPKPCPPPKPCPPPKPCPPPKPCPPPKPCSSP
        PKPCPPPKPCPPPKPCPPPKPCPPPKPCPPPKPCPPPKPC  P
P0C9V6  PKPCPPPKPCPPPKPCPPPKPCPPPKPCPPPKPCPPPKPCPPP
```

Figure 5.7: BLAST with 'short' setting identified only two alignments for the LCR from CD2 homolog protein (P0C9V8). These alignments are almost perfect matches. Source: Jarnot et al., International Conference on Man-Machine Interaction, 2020.

SHORT-BLAST aligned more sequences than BLAST, but found only two similar sequences to the query presented in Figure 5.7. However, these sequences are almost perfectly aligned. This modification discarded the questionable sequences presented in the first two alignments in Figure 5.6. These alignments were rejected due to lower mismatch values in the PAM30 scoring matrix. On the other hand, the sequences are very similar and they belong to the same protein of the same organism. The only difference is that their samples were collected in different time and place. The results contain only strikingly similar pairs of sequences, but they lack false positive hits. It can therefore be concluded that changing the parameter set to *short* improved the quality of the results and slightly increased their number.

By disabling recalculation of the scoring matrix with composition-based adjustment in the default parameters, I significantly increased the number of results. The default

```
P0C9V8  PNPCPPPKPCPPPKPCPPPKPCPPPKPCPPPKPCPPPKPCPPP
        P PCPPPKPCPPPKPCPPPKPCPPPKPCPPPKPCPPPKPCPPP
P0C9V6  PKPCPPPKPCPPPKPCPPPKPCPPPKPCPPPKPCPPPKPCPPP

P0C9V8  PKPCPPPKPCPPPKPCPPPKPCPPPKPCPPPKP
        P+P  P  P+P  P  P+P  P  P+P  P  P+P  P  P+P
P06915  PQPQPRPQPQPQPQPQPQPQPQPQPRPQPQPQP

P0C9V8  PCPPPKPCPPPKPCPPPKPCPPPKPCPPPKPCPPPKPCPPPK
        P PPP P PPP P PPP P PPP P PPP P PPP P PPP+
P12978  PLPPPPPPPPPPPPPPPPPPPPP-PPPSPPPPPPPPPPQ
```

Figure 5.8: Turning off compositional based correction yelded in more results, but these results contain ill aligned records. In the last two examples SHORT-BLAST aligned different kind of STRs/homopolymers. Adapted from: Jarnot et al., International Conference on Man-Machine Interaction, 2020.

value of this parameter is the main reason why BLAST skips many positive alignments. The alignments for COMPOSITION-BLAST are shown in Figure 5.8. In the first alignment, the method still identifies almost perfect matches. I have shown this here because some methods or parameters can mask the most similar sequences and provide only distant similarities. An example of such a method is HHblits. The second alignment compares two short tandem repeats containing sequences. The first is already known, and its repeating unit is KPCPPP. The second is a short tandem repeat of proline and glutamine. Both sequences have been aligned even though they consist of different pattern composition and length. This situation is a common mistake in alignment-based methods, since they look at both sequences from a statistical point of view and ignore their patterns. The alignment relies on proline matches that are abundant in the query sequence and on the similarity between lysine and glutamine according to BLOSUM62. Circumsporozoite protein (P06915) can be found in *Plasmodium berghei*, which is the parasite responsible for malaria in rodents. In the last example, the query is aligned to a homopolymer of proline. This is another wrong way how alignment based on methods match sequences. The query sequence is rich in prolines, which are 4 out of 6 residues in

the repeat pattern. For this reason, homopolymers of proline are good candidates for alignment. Epstein-Barr nuclear antigen 2 (P12978) protein occurs in *Epstein-Barr virus (strain B95-8) (HHV-4) (Human herpesvirus 4)*. This protein plays an important role in transcription activation and causes chromosomal instability in the host organism [106] [107]. Disabling composition biased correction increases the number of hits, but also introduces alignments of potentially irrelevant sequences.

```
P0C9V8  PFPLPNPCPPPKPCPPPKPCPPPKPCPPPKPCPPPKPC
        PFPLP PCPPPKPCPPPKPCPPPKPCPPPKPC PPKPC
Q89501  PFPLPKPCPPPKPCPPPKPCPPPKPCPPPKPCSPPKPC

P0C9V8  PCPPPKPCPPPKPCPPP--KPCPPPKPCPPP--KPCPPPKPCPPP--KPCSSP
        P PPP P PPP P PPP   P PPP P PPP   P PPP P PPP   P S P
Q84ZL0  PSPPPPPSPPPPP-PPPGARPGPPPP-PPPPGARPGPPPPP-PPPGGRP-SAP

P0C9V8  PPPKPCPPPKPCPPPKPCPPPKPCPPPKPCPPPKPCPPPKP
        PPP P PPP P PPP P PPP   PPP P PPP P PPP P
P12978  PPPPPPPPPPPPPPPPPPP---PPPPPSPPPPPPPPPP
```

Figure 5.9: In the first alignment it is visible that SHORT-COMPOSITION-BLAST modification identifies highly similar alignments. It also aligns STRs to poly-P sequences. Source: Jarnot et al., International Conference on Man-Machine Interaction, 2020.

SHORT-COMPOSITION-BLAST collected the largest number of alignments obtained only by changing the BLAST parameters. It generated 2,123,086 alignments. Example alignments are shown in Figure 5.9, where the first aligns the query to a similar sequence. This is another sample of the same organism collected in a different place and time. In the second example, the hit is a poly-P sequence linked with two GARPG motifs. The alignment relies on proline matches. Therefore, before each linker the method added two gaps in the query sequences to align more prolines. The formin-like protein 5 (Q84ZL0) protein in *Oryza sativa subsp. japonica (Rice)* affects cell expansion and biological process of plant shape development [108]. The last alignment is the homopolymer known from Figure 5.8. The algorithm inserted a longer gap into homopolymer to omit a single KPC motif in the query and align one more proline at the end. On the other hand, COMPOSITION-BLAST

found that lysine is similar to glutamine and thus can cause a shorter gap. Note that COMPOSITION-BLAST uses BLOSUM62 scoring matrix that assigns 1 to mutations between lysine and glutamine. A positive value between residues means that they are similar. SHORT-COMPOSITION-BLAST, however, use PAM30 and assign -3 which is considered dissimilar.

```
P0C9V8  PKPCPPPKPCPPPKPCPPPKPCPPPKPCP
        PK CPPPKPCPPPKPCPPPKPCPPPKPCP
P0C9V9  PKLCPPPKPCPPPKPCPPPKPCPPPKPCP

P0C9V8  PPKPCPPPKPCPPPKPCPPPKPCPPPKPCPPPKPCP
        PPKPC PPKPCPPPKPCPPPKPCPPPKPCPP KPCP
Q89501  PPKPCRPPKPCPPPKPCPPPKPCPPPKPCPPSKPCP

P0C9V8  PKPCPPPKPCPPPKPCPPPKPCPPPKPCPPP
        PKPCPPPKPCPPPKPCPPPKPCPP KPCPPP
P0C9V7  PKPCPPPKPCPPPKPCPPPKPCPPSKPCPPP
```

Figure 5.10: LCR-BLAST identifies well aligned sequences regardless of the alignment length. Source: Jarnot et al., International Conference on Man-Machine Interaction, 2020.

LCR-BLAST found more sequences than BLAST with any of the analysed parameter sets, which are more similar at the same time. The method found 5 matches to the query. All these matches come from the same protein and again come from different samples. In this method, the threshold is the mean score value. When decreasing it, the method finds more distant results. The current threshold value is 4.7, and accepts 2.1 mutations for every 10 residues. This value allows for some mutations in the LCRs, ensuring that the short tandem repeats are also well aligned, since it rejects sequences with mutations in each repeat unit. LCR-BLAST alignments are also better in terms of quality. Compared to BLAST with default parameters, it rejects sequences that are strikingly different. SHORT-BLAST missed several alignment which are obviously similar to the query that LCR-BLAST included in the results. COMPOSITION-BLAST aligned sequences consisting of different repetitive patterns, while the new method rejected them. SHORT-COMPOSITION-BLAST identifies sequences that are compositionally biased to the query. LCR-BLAST,

on the other hand, also is easier to configure for STR comparison and other LCR types since the mean score is intuitive for these purposes, while E-value depends on multiple variables including sequence length.

5.4 Discussion

In this chapter I introduced LCR-BLAST, a modification of BLAST that is capable of handling LCRs. I presented the description of the method along with its analysis, and compared with BLAST for different parameter settings. All new features of the method are analysed including changed parameters, scoring matrix and mean score statistic.

I showed, step by step, how LCR-BLAST was developed and how individual changes affect the results. Each presented modification increased the number of results. *blastp-short* parameters also increase results quality, while disabling composition based adjustment additionally includes distant results that are less likely to be biologically significant. The final version, LCR-BLAST, provides a large number of results while maintaining reasonable quality of alignment. I showed that COMPOSITION-BLAST and SHORT-COMPOSITION-BLAST badly aligned some sequences. However, these sequences were rejected in LCR-BLAST by the mean score threshold. Quantitatively, this can be seen in Figure 5.4, where such alignments are not overlapping with LCR-BLAST. In Figures 5.2 and 5.3 I show that LCR-BLAST is the only method where alignment lengths consistently match LCR lengths.

LCRs have different residual preferences than HCRs and therefore a new scoring matrix for LCRs is needed [57]. Wherever possible, dedicated scoring matrix should be used to find sequences with similar functional or structural properties. However, we lack a scoring matrix dedicated to analyze similarities between LCRs [92]. Therefore, I introduced identity scoring matrix which ignores evolutionary relationships between proteins. I plan to continue development of a new scoring matrix that will reflect amino acid preferences of LCRs.

In the core of BLAST is the Smith-Waterman algorithm, which may result in incorrect alignment of sequences with different lengths [18]. Because it is a local alignment algorithm, it cuts longer sequences than query without information that the length of sequences differ. However, this information may be crucial to conclude on protein function by similarity to another sequence fragment. In the ,,Analysis of

canonical methods for protein sequence comparison in the case of LCRs" section I present a detailed analysis of this issue. I plan to solve it in the future work.

E-value is a metric designed to find evolutionary related proteins. By default, BLAST masks LCRs using composition based correction. This is because LCRs are often the cause of false positive hits. Similar LCRs playing the same function may be evolutionarily related, but this is not the rule. For instance, Schaefer et al. in their work published in 2012, concluded that, most likely, zebrafish poly-P regions in different locations evolved independently [95]. If we choose the bit score as the threshold metric then longer sequences will have a greater chance of being similar, resulting in the alignments as in Figure 5.9. I introduced the mean score to make the threshold metric independent of the sequence length. The intuition behind this is simple and indicates the minimal score value in average for a single aligned residue. Examination of selected examples presented in Figure 5.10 shows that it helps to find similar LCRs that share similar compositional properties and repeating patterns.

To summarize, I improved and analyzed several BLAST variants for LCR comparison. I optimized the available BLAST parameters to search for similar LCRs. I also introduced the mean score and the identity scoring matrix, which are more efficient in similarity comparison of LCRs by their composition.

6 GBSC - clustering short tandem repeats

6.1 Introduction

Several methods for LCR identification already exist, which are presented in the ,,Low complexity identification methods" chapter. These methods are important in protein sequence analyses since sequence complexity, composition and regularity can affect protein properties [109–111]. Additional value can be added if we use similarity comparison methods to find similar protein sequences. The three state-of-the-art methods were used for LCR analyzes and compared [29]. These methods can be used for LCRs but definitely need improvements. Therefore, LCR-BLAST was introduced as a modification of the BLAST method for LCRs [30]. LCRs are domains that can be divided into STRs (including homopolymers) and sequences with an irregular order of residues. The same sequence composition with a different arrangement of amino acids can result in different biological properties [112]. Therefore, such a division is justified. We already know that methods based on statistical models incorrectly compare STRs in sequences [29]. In particular, we lack a method that properly clusters short tandem repeats not by their composition, but by the repetitive patterns they consist of. In this chapter, I describe a new method for clustering similar STRs, which identifies them in protein sequences, builds a corresponding graph model and clusters them by similar patterns. Comparison with other LCR identification methods is described in the ,,Low complexity identification methods" chapter, hence this chapter focuses only on cluster analysis and comparison with other methods.

Algorytm 1 Identification and clustering of the GBSC method

```
1: let kmers ← k-mers of the input sequence
2: let lifetime_t ← value of lifetime threshold
3: let weight_t ← value of weight threshold
4: let G_s ← empty graph
5: for each kmer in kmers do
6:     let prev_kmer ← previous k-mer
7:     let curr_kmer ← current k-mer
8:     if node of curr_kmer not exist in G_s then create node and add it to G_s
9:     if prev_kmer not exist then start next iteration
10:    let prev_node ← node of prev_kmer
11:    let curr_node ← node of curr_kmer
12:    let curr_edge ← edge between prev_node and curr_node
13:    if curr_edge not exist then create curr_edge and assign 0 to its weight
14:    add 1 to weight of curr_edge
15:    set lifetime of curr_edge to lifetime_t
16:    subtract 1 from lifetime of all edges
17:    let repetitive_edges ← edges of G_s with lifetime = 0 and weight ⩾ weight_t
18:    if repetitive_edges exist then
19:        let G_r ← graph consisting of repetitive_edges
20:        let str ← sequence fragment of G_r
21:        if str passed postfiltering criteria then
22:            append str to list of identified repeats
23:            append str to cluster(s) represented by G_r
24:        end if
25:    end if
26:    delete all edges of which lifetime is 0 from seq_graph
27:    delete all nodes without edges
28: end for
```

6.2 Methods

The Graph Based on Sequence Clustering (GBSC) method is designed to identify STRs in protein sequences and cluster them according to similar patterns. It scans the sequence, identifies repetitions and tags them with the corresponding graph model of repetitive pattern. The detected domains are then clustered according to a similar graph model. Additionally, it can identify and cluster repeats using reduced alphabet. For clustering, it also handles adjacent STRs. In addition, I compared GBSC with other protein clustering methods. The identification and clustering process of GBSC is presented in Algorithm 1.

6.2.1 Identification

GBSC identifies STRs by scanning the sequence and building De Bruijn-like graphs that represent the STRs [113]. To identify STRs, GBSC iterates over k-mers of the sequence in the order they appear. In each iteration, it creates a node from the current k-mer and connects the previous node to the current one, creating an edge between them. Each edge has a lifetime and a weight with initial values of 0 and 1, respectively. The lifetime increases on all existing edges each time the algorithm jumps to the next k-mer. If the lifetime reaches a user-specified threshold then it disappears. In case an edge between the current and previous k-mers already exists, the edge lifetime is reset to 0 and its weight increments. If a vanishing edge has a weight assigned above a user-specified threshold then all of these edges form a graph representing the STR. The first and last k-mers, which belong to the graph, define the boundaries of the identified STR. For an infinite lifetime threshold, the algorithm creates a weighted De Bruijn graph of the sequence and removes edges with a weight below the threshold. The results of this process are identified STRs and graphs representing each domain.

6.2.2 Identification parameters

The identification process has several parameters that can be changed to detect different STR types. In the algorithm, nodes can behave the same way as edges. This change includes more degenerate STRs. Lifetime determines the maximum distance between edge occurrence, therefore it limits the maximum gap length between repeating units and the maximum length of repeating units. The weight threshold

corresponds to the minimum number of repeats that must occur to treat them as STR. I also added four parameters for post-filtering which determine whether the identified STR will be reported. The first parameter is the maximum gap length that can be used to detect STRs with a limited number of insertions between repetitions. This option is useful when detecting TRs with a longer repeat unit by increasing the lifetime parameter to keep gaps between repeats short. The algorithm can filter out homopolymers, considering them well-studied for some studies [114]. It also enables specifying maximum and minimum number of nodes in the domain model.

6.2.3 Identification example

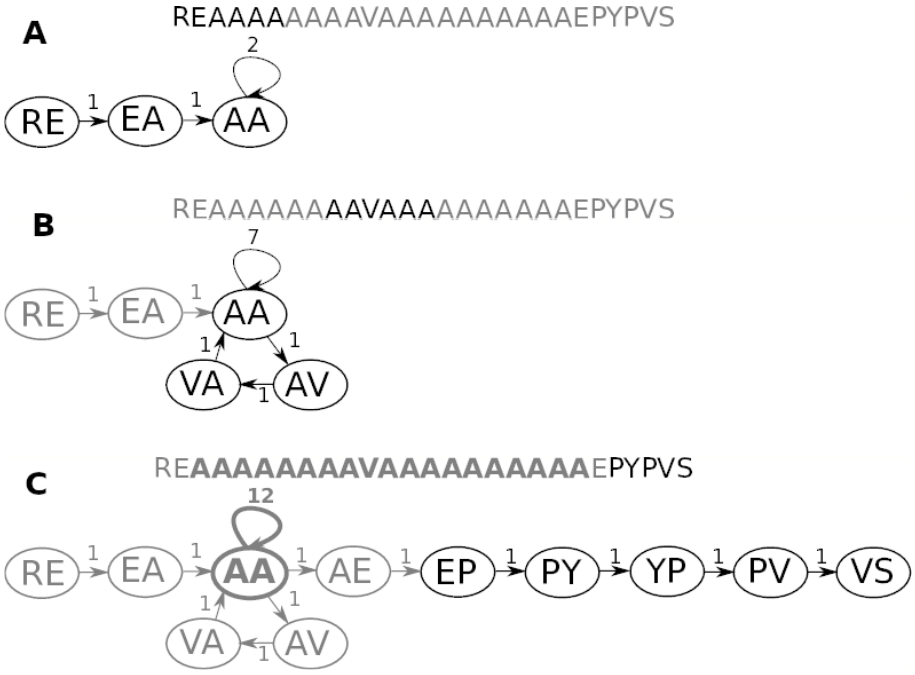

Figure 6.1: An exemplary process of identifying STRs and creating a graph model of repetitions. **(A)** shows the state after processing the first five k-mers and approaching the poly-A domain. In **(B)**, the method handles valine insertion, and in **(C)** leaves the poly-A domain.

Figure 6.1 presents the identification of a poly-A domain with a single valine insertion. The algorithm successively processes the k-mers as nodes. At the beginning, it creates new nodes and connects them with edges, processing non-repetitive residues. Next, it approaches the poly-A region, adding weight to self-loop of AA-node, as shown in the panel **(A)**. It then moves forward, deleting nodes that exceed the

lifetime threshold. It also creates nodes to overcome the valine insertion and goes back to increasing the self-loop weight of AA-node. The processes of deleting nodes and handling insertions are shown in the panel **(B)**. After several iterations, the algorithm removes the nodes created for valine insertion. Finally, the algorithm exits the homopolymer, tags it as a region containing repeats with 12 as the weight value, and assigns the graph to the poly-A fragment. This is shown in the panel **(C)**.

6.2.4 Clustering

The GBSC clustering is the next step in the STR analysis, which is based on the identified fragments and their assigned graph models. The algorithm assign sequences to clusters of the same graph model. Sequences are assigned to cluster just after a sequence is identified. Assigning sequences to clusters this way ensures that each cluster contains sequences consisting of a similar pattern. On the other hand, it allows sequence diversity in clusters, which is introduced by random residues in repeating patterns. Such random residues sometimes follow a fixed pattern in the STR domain, thus they can be considered an integral part of the STR [52]. Another common STR configuration is when one type of STR is side by side with another type. An example of such an adjacent STR can be seen in Figure 1.1 where QA repeats are followed by a poly-Q domain. GBSC handles adjacent STRs by assigning them to clusters of their separate and combined types. For two STRs called A and B, where A is followed by B, these STRs will join three clusters: A, B and A-B. These operations make GBSC able to cluster protein fragments by a wide variety of STRs.

6.2.5 Alphabet reduction

The method may optionally reduce the alphabet of residues by merging similar amino acids to identify and cluster STRs. Some amino acids share similar biochemical and biophysical properties. Among others, Li et al. showed that reducing the alphabet of canonical amino acids to nine groups performs well for protein contact interactions [115]. GBSC reduces the alphabet just before creating the model during identification. Using this approach, it produces as output untouched sequences and GBSC models built from a reduced alphabet.

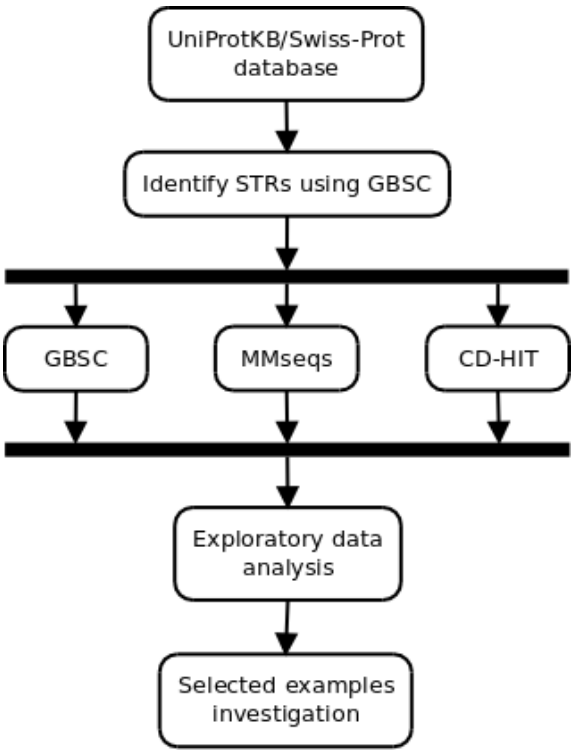

Figure 6.2: Workflow of the analysis. I identified STRs using GBSC. I then ran GBSC, MMseqs and CD-HIT on identified fragments. Finally, I analyzed results.

6.2.6 Clustering comparison

I compared GBSC with other methods for protein sequence clustering. For comparison, I chose the commonly used methods, which are MMseqs and CD-HIT [27, 46]. They compare sequence composition between sequences, while GBSC clusters sequences according to similar repetitive patterns. The selected methods accept FASTA formatted sequences as input. I used STRs identified by GBSC to make the input to all methods unified and focus only on clustering characteristics. I compared these methods by exploring the results and examining selected examples. For exploration, I provide statistics describing the clusters, and use a Venn diagram to show the quantitative relationships between the sets resulting from each method. These relationships demonstrate how the results of one method overlap with the results of other methods. I generated overlap by combining all sequences in all clusters into pairs of similar sequences. For instance, a cluster containing three sequences (S1, S2 and S3), will generate three pairs of similar sequences, which are S1-S2, S1-S3 and S2-S3. I use these pairs to calculate all areas of the Venn diagram. To show what

Method	Identification parameters	Cluster count	Orphan cluster count	Mean sequence count	Std dev. of sequence count
GBSC	Strict	2,609	1,754	15.0	193.9
	Relaxed	5,511	3,571	16.0	201.4
MMseqs	Strict	6,502	5,066	6.0	63.3
	Relaxed	28,765	20,832	3.0	29.9
CD-HIT	Strict	1,979	900	19.7	184.3
	Relaxed	14,152	7,278	6.2	65.3

Table 6.1: Cluster statistics. The columns contain (1) the name of the method, (2) the number of clusters, (3) the number of clusters containing only one sequence, (4) the average number of sequences per cluster and (5) the standard deviation of the number of sequences per cluster.

similarities exist in the different areas of the Venn diagram, I also selected examples from Venn areas and described them. The workflow of cluster comparison is shown in Figure 6.2.

I used two sets of parameters to show how GBSC clustering relates to other methods when the input consists of clear and degenerate repetitions. To get clear repetitions, I used a *strict* set of parameters. This set of parameters changes the weight threshold (w) to 4, the edge lifetime (l) to 10, the maximum gap length (m) between repetitions to 2 and the maximum number of nodes (x) to 6. For degenerate repeats I used *relaxed* parameters, which are default in the method. Compared to the *strict* parameters, I changed the lifetime (l) to 20, increased the maximal gap length (m) to 3 and allowed the nodes to behave as edges (n).

6.3 Results

6.3.1 Data exploration

A summary of the generated clusters can be read from Table 6.1. Along with relaxing the parameters, most of the cluster statistics changed significantly. The number of GBSC clusters increased by 211%, but this is the smallest change compared to other methods. For MMseqs, the number of clusters increased by 442%, and for

	Strict STRs	Relaxed STRs
GBSC	49,323,736 (97.4%)	112,374,211 (94.8%)
MMseqs	13,130,732 (25.9%)	12,979,339 (10.9%)
CD-HIT	33,957,979 (67.0%)	30,428,496 (25.7%)
Total	50,643,618 (100%)	118,540,331 (100%)

Table 6.2: Total number of pairs of similar sequences identified by each method for *strict* and *relaxed* GBSC parameters.

CD-HIT by as much as 715%. The rationale for this is that relaxing the parameters increases the number of mutations between repeats, which may vary from sequence to sequence. GBSC omits these mutations for comparison, hence the increase in the number of clusters for this method is smaller than for the other methods. The rest of the statistics also follow this rule. For composition based methods, the average and standard deviation of the number of sequences per cluster are much smaller, which is due to the greater difference between the sequences.

GBSC found the largest number of similar pairs, the second was CD-HIT and the lowest number of similar pairs, but still high, were found by MMseqs. The Table 6.2 contains the number of pairs of similar sequences identified by each method. The obtained results cannot be directly compared with those known from the chapter ,,Analysis of canonical methods for protein sequence comparison in the case of LCRs" since I used different methods for LCR identification. There I used the SEG method while here I used GBSC. However, overall 25.9% for *strict* parameters and 10.9% for *relaxed* parameters for MMseqs is high since the intersection of BLAST, HHblits and CD-HIT is close to 0%. Therefore, the methods compared in this chapter (GBSC, MMseqs and CD-HIT) may be considered as designed for a similar purpose. Intuitively, relaxing the parameters of the GBSC method, which is used for identification, will result in more similar pairs. This is true for GBSC, but the other methods found fewer pairs. However, this does not lead to any conclusions as the sequences used for clustering are identified by GBSC, and each sequence initially has assigned a repetitive model for GBSC clustering. In addition, the *relaxed* parameters may extend the fragments identified by the *strict* parameters, and thus may belong to different clusters. Moreover, if one sequence is moved from a large cluster to a small cluster, the total number of similar pairs decreases, but the sequences are better

distributed across the clusters. Therefore, the quantitative results of the number of similar pairs are only illustrative.

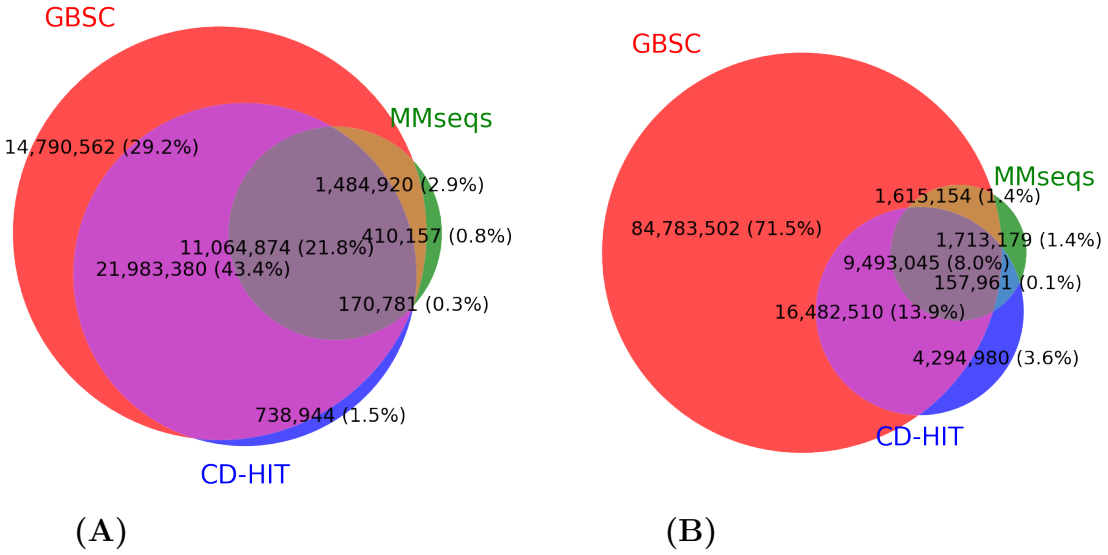

Figure 6.3: The diagrams present overlap among GBSC, MMseqs and CD-HIT for input STRs identified by GBSC-strict and GBSC-relaxed, respectively. For *strict* STRs, GBSC has more overlapping sequences with other methods than for *relaxed* STRs.

Intersection of all methods for *strict* parameters of GBSC is large, while for *relaxed* parameters MMseqs and CD-HIT deviate from GBSC. The overlap is shown in Figure 6.3. Relaxing the parameters in GBSC introduces more mutations and therefore the sequences are more diverse in composition. The GBSC method finds similar patterns in STRs, while the rest of the methods compare the composition of two fragments. In theory, if the mutations between STRs differ, GBSC will ignore them while the other methods will take them into account. This is particularly noticeable when comparing *strict* and *relaxed* parameters. For *strict* parameters, the intersection between MMseqs and CD-HIT is 31.3% of similar pairs, while for *relaxed* parameters it is 27.5%. The difference in the intersection of these methods between *strict* and *relaxed* parameters is 2.7%. This is small compared to the difference between GBSC and CD-HIT, which is 43.6%. Therefore, GBSC responds differently to inter-repeat mutations than MMseqs and CD-HIT.

6.3.2 Selected examples examination

Data exploration provides an overview of the results from which some conclusions can be drawn, while the examination of selected examples provides details. We can quantitatively compare methods and observe their trends as parameters change, but we ignore detailed differences. On the other hand, the examination of selected examples shows the detailed characteristics of the methods, but it lacks a comprehensive overview. In this section, I provide an analysis of selected examples to show what STR types each method supports.

```
GBSC model 1: GR→RG→GG
Sequence 1: GRGGRGGYGRGRGGYGGRGG
Sequence 2: RGRGRGRGRGRG
GBSC model 2: GR→RG→GR

GBSC model 1: AA→AA
Sequence 1: AAAAAAA
Sequence 2: PAAAPAPAAAAPAPAAAAPAPAPAPAAAPAAAPAA
GBSC model 2: PA→AA→AA→AP→PA
```

Figure 6.4: Two similar pairs identified by CD-HIT only for *strict* and *relaxed* parameter sets, respectively. The GBSC models are added to the sequences for reference.

The Figure 6.4 presents two example pairs of sequences belonging to the same clusters; found only in CD-HIT results. The first pair are sequences composed of RGG and RG repeats, respectively, where the first is degenerate and the second is a perfect repeat. The GBSC assigned them to different clusters represented by RGG and RG repeats. Sequences from the second pair are also different kinds of repeats. However, in this case the first sequence is a poly-A fragment while the second can be represented as a regular expression by PA* pattern. It consists of proline and a varying number of alanines. For alignment algorithms that do not recognize repeats, both pairs of sequences are similar, thus CD-HIT assigned them to the same clusters.

Similar sequence pairs identified only by MMseqs are shown in Figure 6.5. The first pair consists of the poly-N sequence and MNNNN pattern. For alignment algorithms, it is difficult to compare such sequences properly because the poly-N

```
GBSC model 1: NN→NN
Sequence 1: NNNNNNNNNNNN
Sequence 2: MNNNNMNNNNMNNNNMNNNNMNNNNMNNNNMN
GBSC model 2: MN→NN→NN→NM→MN

GBSC model 1: SS ST TS
Sequence 1: TSSTTSTTSSTSTSSSS
Sequence 2: SSSSSSSSSSSSSSSS
GBSC model 2: SS→SS
```

Figure 6.5: Similar pairs identified by MMseqs only for *strict* and *relaxed* parameters, respectively.

and MNNNN repeats are identical in 80%. Such statistical approaches cannot handle these cases because if we adjust the parameters to handle it, then it will only find nearly identical matches, but for sequence comparison we want to allow some mutations. The sequences in the second pair are rich in serine, but the first sequence also contains some threonines. These threonines form irregular repeats of TS patterns, while both residues may occur multiple times in the pattern. MMseqs classified both sequences as similar, even if their identity is only 58.8%. On the other hand, GBSC identified the first sequence fragment as a degenerate TS repeat. In this case, we cannot indisputably judge which method correctly classified the sequences. This is because it depends on the application. Protein fragments may be required for protein function due to their composition, repetitiveness, complexity etc. GBSC classifies these sequences by repeats. On the other hand, MMseqs classifies them by composition, and this is rational since serine and threonine are similar residues, and MMseqs uses a reduced alphabet. GBSC, for different parameter settings, is also able to treat these sequences as two homopolymers of hydroxylic residues.

In Figure 6.6, I show what kind of pairs of similar sequences were clustered by MMseqs and CD-HIT, but missed by GBSC. The first pair is glutamine-rich, but the first sequence also contains several proline insertions that create repeats of a single proline residue and multiple glutamines. The number of glutamines varies between repeat runs. This shows that the issue of classifying different repeat types into the same cluster is common to MMseqs and CD-HIT. The second pair consists of two similar sequences of similar composition and even a single proline insertion, but the

```
GBSC model 1: PQ→QQ→QQ→QP→PQ
Sequence 1: QQQPQQQPQQQQPQQQQPQQQQPQQQQQQQPQQQQQPQQQ
Sequence 2: QQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQ
GBSC model 2: QQ→QQ

GBSC model 1: AA
Sequence 1: AAPAAAAA
Sequence 2: AAAAPAAAAA
GBSC model 2: AA→AA
```

Figure 6.6: Similar pairs identified by MMseqs and CD-HIT, but not by GBSC for *strict* and *relaxed* parameters, respectively.

first sequence is two amino acids shorter than the second sequence. Therefore, I provide the second example to illustrate the GBSC issue with the *relaxed* parameter set. If one of the sequences is shorter and its length is on the verge of detecting it as a repeating node or self-loop, it can be assigned to different clusters. The solution to this issue is to postprocess the clusters and move such sequences from one cluster to another, or to place such sequences in multiple clusters, or simply delete them, or leave unchanged. Nevertheless, the solution is independent of the algorithm core and depends in particular on the dataset being analyzed. Therefore, I would like to warn the community about the fact that this issue may occur. These cases can be easily handled since GBSC assigns graph models to clusters, and such sequences can be located by looking at specific clusters.

The Figure 6.7 shows examples found only by GBSC and no other methods. The first two pairs of similar sequences represent repeats identified by *strict* and *relaxed* parameters, respectively. The first pair is an irregular repetitive fragment composed of proline and varying numbers of alanines. The second pair is more irregular with more insertions between repetitions. These examples demonstrate the usefulness of the GBSC method for comparing irregular repetitions over other methods. A final example is two adjacent sequences, the first is homopolymer of asparagine and the second is tandem repeat of glycine and asparagine. In addition, both subsequences are also assigned separately to poly-N and GN repeat clusters. Such adjacent sequences are handled in this way only by the GBSC method. In the MMseqs and CD-HIT results, these subsequences are assigned only to homopolymer of asparagine and the

```
GBSC model 1: AP→PA→AP
Sequence 1: APAAAPAAPAAGAPAPAPAPAAPAPAPA
Sequence 2: APAPAPAAAPAPAPA
GBSC model 2: AP→PA→AP

GBSC model 1: EL
Sequence 1: ELAAELDELESEEL
Sequence 2: ELLELVELEVREL
GBSC model 2: EL

GBSC model 1: NN→NN GN→NG→GN
Sequence 1: NNNNNNGNGNGNGNGN
Sequence 1: NNNNNNNNNGNGNGNGNGN
GBSC model 2: NN→NN GN→NG→GN
```

Figure 6.7: Similar pairs identified by GBSC only for *strict* and *relaxed* parameters, respectively.

cluster containing the GN repeats. It is worth noting that in the SEG results from ,,Analysis of canonical methods for protein sequence comparison in the case of LCRs" chapter, if the CD-HIT method created a cluster from such adjacent LCRs, then all types of sub-LCRs were also included in this cluster. As described in the mentioned chapter, this leads to a situation where dissimilar sequences are assigned to the same cluster. In summary, canonical methods handle adjacent LCRs in different ways depending on the method used for identification.

The next examples, shown in Figure 6.8, come from the GBSC and MMseqs overlap. The first pair consists of a clear GGM pattern with a single repetition longer in the second sequence. CD-HIT added these sequences to different clusters because they were similar to different cluster representatives. Similarity between these sequences was not checked by this method. The second pair consists of irregular repeats. Both sequences differ in some residues between repeats and were assigned to different clusters by CD-HIT for the same reason as the first pair.

The Figure 6.9 presents two pairs of similar sequences clustered only by GBSC and CD-HIT. These pairs are serine/arginine-rich, consisting of RS repeats, and

```
GBSC model 1: GG→GM→MG→GG
Sequence 1: GGMGGMGGMGGMGGMGGMGGM
Sequence 2: GGMGGGMGGMGGMGGMGG
GBSC model 2: GG→GM→MG→GG

GBSC model 1: CK SC
Sequence 1: SCKCKDCKCASCKKSCCSC
Sequence 2: SCKCKECKCTSCKKSCCSC
GBSC model 2: CK SC
```

Figure 6.8: Similar pairs identified by GBSC and MMseqs but not by CD-HIT for *strict* and *relaxed* parameters, respectively.

are alanine-rich, which are homopolymers of alanine with two insertions in each sequence. The longer RS repeat sequence has three mutations while the shorter sequence has two mutations. These sequences belong to different clusters in MMseqs because they form multiple clusters for RS repeats. Interestingly, MMseqs appends exact repetitions of the RS pattern of the same length to different clusters. The only difference in these sequences is that in one cluster the sequences start with serine while in the other clusters the sequences start with arginine. The alanine-rich pair has two mutations in both sequences, where one mutation is into a valine, which is at the same position in both LCRs. The second mutation is proline in the first STR and histidine in the second STR. Proline and histidine mutations occur at different positions. The sequences in the cluster containing the first sequence can be described by the formula AAA[A][A][A]VXAAA where X is a random amino acid. The cluster with the second sequence has only identical LCRs. The first cluster contains 17 sequences and the second 26. In conclusion, MMseqs divides similar sequences into multiple clusters, which explains why this method found the least number of similar pairs (Table 6.2).

The last set of sample pairs is shown in Figure 6.10, and they belong to the same cluster in all three methods. These pairs consist of exact PE repeats and irregular EL repeats. The first pair was found in a cluster of STRs identified by *strict* GBSC parameters, and the second by *relaxed*. From a compositional point of view, both sequences are perfect matches. The only difference between the sequences is the length in the first pair, where the short sequence is 26.2% shorter than the

```
GBSC model 1: RS→SR→RS
Sequence 1: RSRSRSWSRSRSRSRRYSRSRSR
Sequence 2: RSRSPSSSRSRSRSRSRS
GBSC model 2: RS→SR→RS

GBSC model 1: AA
Sequence 1: AAAAVPAAA
Sequence 2: AAHAVAAAA
GBSC model 2: AA
```

Figure 6.9: Similar pairs identified by GBSC and CD-HIT but not by MMseqs for *strict* and *relaxed* parameters, respectively.

longer sequence. To summarize, the intersection area of Venn diagrams presented in Figure 6.3 contains highly similar pairs of sequences.

6.4 Discussion

In this chapter I introduced a new method of identifying and clustering STRs. Depending on the configuration, this method can handle exact and degenerate repetition of short motifs. In the ,,Low complexity identification methods" chapter, I compared the identification feature of GBSC with a wide range of LCR identification methods. Therefore, in this chapter I have focused on clustering and compared this method with the state-of-the-art statistical methods of clustering protein sequences. I explored the data to overview the results and I examined selected examples to demonstrate the differences between the methods in detail.

GBSC accepts different insertions between repeats. The method clusters similar sequences by repetitive pattern that it finds during identification. Therefore, unlike other protein sequence clustering methods that are analyzed in this work, GBSC handles repeats that partly contain a specific pattern and partly accept random residues. Alignment based on methods count these random residues as mismatches and therefore treat them as different STRs. GBSC allows gaps between repeating units. It has already been found that STR can be composed of conserved and unconserved parts. For example, the KSPXXX/KSPXK repeat can be found in

```
GBSC model 1: EP→PE→EP
Sequence 1: PEPEPEPEPEPEPEPEPEPEPEPEPEPEPEPEPEPEPEPEPEPEPEPEP
            EPEPEPEPEPEPEP
Sequence 2: PEPEPEPEPEPEPEPEPEPEPEPEPEPEPEPEPEPEPEPEPEPEPEPEP
GBSC model 2: EP→PE→EP

GBSC model 1: EL
Sequence 1: ELLELVELEVREL
Sequence 2: ELLELVELEVREL
GBSC model 2: EL
```

Figure 6.10: Similar pairs identified by all methods for *strict* and *relaxed* parameters, respectively.

neurofilament heavy polypeptide protein [116, 117]. The first part of this repeating pattern consists of conserved Lys-Ala-Pro motif. The second part is flexible and consists of random residues of various lengths, but most units contain three random residues. However, to also take the residues that are between the repeat units, researchers can generate MSA from clusters [118].

GBSC handles adjacent STRs when clustering. This type of STRs is generally problematic for canonical methods of protein sequence clustering. These methods either produce a cluster containing all subtypes of STRs, as shown in Figure 4.9 **(A)**, or compare them globally. To properly handle this type of STRs, we need a more sophisticated method that is aware of such sub-LCRs. Since GBSC first identifies STRs, builds a representation model from it, and uses it for clustering, thus it is aware of STR types and their distance from each other. Therefore, it is able to properly handle such adjacent STRs. The practical application of this feature is demonstrated in the next chapter in the case of DNA-directed RNA polymerase subunit delta [33].

Alphabet reduction enables identifying and clustering repetitive patterns of physicochemical properties of residues. For instance, aspartic acid and glutamic acid have very similar properties, thus in some research can be treated identically [119]. It is also possible to analyze tandem repeats of negatively and positively charged residues. In the RNA-binding RGG box, an RGG repeat unit is usually also composed of YGG [120]. By reducing arginine and tyrosine to a single amino acid group, GBSC

can identify such domains as exact matches. Alphabet reduction can also be used to detect collagen structures that require glycine and two random residues [121]. The collagen repeat can be described by the GXX pattern, where X stands for random residue. Therefore, one can keep the glycine unchanged and reduce the rest of the amino acids to one group in order to cluster all the collagen-like domains. In the next chapter, I show how to use GBSC with alphabet reduction in practice to find K-DE motifs.

GBSC tags identified and clustered sequences using interpretable graph models that describe the identified domains. Most LCR identification methods lack information on why the fragment was identified. Also protein similarity analysis methods cluster sequences with similar compositions without labeling produced clusters. Such labels can be crucial in choosing the right algorithm for a given analysis, even if the performance of uninterpretable models is better [122]. For example, T-REKS is a method for identifying highly degenerated tandem repeats. The method lacks labels describing the detected domains, and it is sometimes difficult to find repeats in these sequences. Examples of such degenerated repeats are shown in Figure 3.7 **(A)**. Protein sequence clustering methods also suffer from the same issue. The sequences in the first pair shown in Figure 6.9 belong to different clusters in MMseqs method. These clusters contain similar sequences, and thus could be merged. However, such clusters are difficult to find because they lack labels of clusters. Relaxing the MMseqs parameters is also not a solution as the method already assigns sequences of different repeats to the same clusters as shown in Figure 6.5 in the first pair. Therefore, GBSC may be especially a choice when interpretability of repeats is an important factor.

In this chapter, I introduced a new method for identifying and clustering STRs in protein sequences. This method can use alphabet reduction when building the model for clustering, and identifying degenerate repeats. It also handles adjacent repeats that may be relevant to the biological properties of LCRs [33]. In the next chapter, I show how this method can be used in practice.

7 Applications

7.1 Introduction

In the scope of this thesis, I introduced two methods for the similarity comparison of protein sequences. The first is LCR-BLAST, which is for searching for similar LCRs using local alignment. And the other is GBSC; a method for clustering STRs according to their repeating patterns. I have already compared them with other solutions, but I did not check whether these new methods can be applied to biological analyses. In this chapter, I present examples of such biological analyses where the above methods have helped to study protein sequence datasets. These analyses are aimed at finding possible assembly errors in STRs, searching for similar fragments to K-D/E domain from delta subunit of bacterial RNA polymerase and finding common epitopes for SARS-COV-2 and the human genome.

7.2 Assembly errors in STRs

To create protein sequence databases, scientists extract the genome from organisms and use specialized technology and methods to read it. Such technologies include the Illumina sequencing platform (Illumina), Single Molecule Real Time Sequencing (PacBio) or Nanopore Sequencing (Nanopore) [123–125] Excluding simple organisms like viruses, we lack technology that can read whole genomes without complications. They all have their advantages and disadvantages. Illumina has a relatively low error rate but reads genomes in short chunks (< 250 bp). PacBio and Nanopore, on the other hand, have a higher error rate, are more expensive, but are able to read longer sequence fragments (usually 10-40 kbp). In the next step, computational methods assemble these reads into a single genome sequence [126, 127]. Short read sequencing technologies can be affected by sequence assembly errors. This is due to the fact that if we have two repetitive regions of the same type that occur in different parts of the genome, and the reads do not cover entire such regions, then we do not know

which repetitive region the reads should belong to. The hypothesis is that existing genomes and their products are affected by sequence assembly errors in databases.

Protein sequences stored in databases are often translated from genes. Several genome annotation methods already exist which mark gene positions [128]. The genes are further translated to RNA and then to protein sequences. Therefore, sequence assembly errors in genomes also affect protein databases. In this study, I checked whether such errors possibly exist in protein sequences by investigating the STRs.

7.2.1 STR length variation in different sequence versions

```
NGPVGDQLSLLFGDVTSLKSFDSLTGCGDIIAEQDMDSMTDSMASGGQRANRDGTKRSSC
NGPVGDQLSLLFGDVTSLKSFDSLTGCGDIIAEQDMDSMTDSMASGGQRANRDGTKRSSC

LVTYQGGGEEMALPDDDDNDDEEEEEEEEKKKKKKKKKKKKKKKK----------------
LVTYQGGGEEMALPDDDDNDDEEEEEEEEEEEEEEEEEEEEEEEEEELLEDEEEVKDG

--------------------------------------------------------------

EENDDLEYLWASAQIYPRFNMNLGYHTAISPSHQGYMLLDPVQSYPNLGLGELLTPQSDQ
```

Figure 7.1: The older (upper) version of the sequence in the poly-E fragment becomes poly-K and finally terminates, while the newer version of the sequence continues with poly-E and then turns into a high-complexity sequence. The figure shows a fragment of a pairwise alignment of protein sequences in two different versions. The sequences comes from APC membrane recruitment protein 1, mus musculus (mouse) (Q7TS75). Adapted from: Tørresen et al., Nucleic Acids Research, 2019.

In this part of the analysis, I examined the variation in length between different versions of sequences in the UniProtKB/Swiss-Prot [78]. For this purpose, I downloaded the first and last versions of each sequence. I rejected sequences with only one version. Within each pair of sequences, I identified STRs using GBSC with a strict parameter set (w: 4, l: 10, g: 2, x: 6). Then I also rejected proteins without STRs in at least one version. Next I aligned both versions of sequences using the Kalign method with the default parameters [129]. I checked if the aligned STRs differ in length and computed the differences. In case only one version contains an STR, I count it as a difference in length. An example alignment with identified STR is presented in the Figure 7.1.

Proteins (n)	Proteins with different sequence between versions (n)	Proteins with different repetitive region lengths (n)	Average/standard deviation of the length of repetitive regions in original version of the sequence	Average/standard deviation of the length of repetitive regions in the version 2018_06 of the sequence	Average/standard deviation of the difference in lengths of repetitive regions
554 241	74 434	1 669	31.14/72.09	35.20/84.08	13.57/45.69

Table 7.1: Protein sequence versions with different lengths of STRs occur in the UniProtKB/Swiss-Prot database. Source: Tørresen et al., Nucleic Acids Research, 2019.

The results of the analysis are placed in the Table 7.1. The input dataset consists of 554,241 sequences, of which 74,434 have multiple sequence versions. GBSC identified 1,669 sequences where the repeat regions differ between versions. The average length of the first versions is shorter than the average length of the last versions. Even if this does not yet mean that we have found 1,669 sequences with assembly errors, since both collected genome samples may simply be different, it is a good overview which supports stated hypothesis.

7.2.2 STR length variation in different taxonomies

Database name	Number of proteins	Number of proteins with STRs	% of proteins with STRs	Median	Average	Standard deviation	Number of clusters
UniProtKB/ Swiss-Prot (total)	554 241	28003	5.05%	14.75	15.14	3.69	6237
Archaea	19 525	351	1.80%	10.71	10.63	1.27	45
Bacteria	333 691	6794	2.04%	17.38	17.45	2.66	1048
Euk: Fungi	33 613	3996	11.89%	13.46	13.79	3.65	893
Euk: Invertebrata	27 607	3372	12.21%	17.34	18.62	7.95	812
Euk: Vertebrata	18 292	1461	7.99%	13.66	13.90	2.42	1801
Euk: Plants	42 101	3601	8.55%	12.51	12.82	2.98	795
Viruses	16 852	889	5.28%	14.07	14.15	2.57	203

Table 7.2: STRs in protein sequences in the UniProtKB/Swiss-Prot database divided by taxonomy. Source: Tørresen et al., Nucleic Acids Research, 2019.

Another approach to prove or disprove the hypothesis is to check variability of STRs in taxonomies. To calculate it, I used the UniProtKB/Swiss-Prot database and the

taxonomic division of sequences from the UniProtKB [78]. I used this data to divide the protein sequence database into taxonomies and analyze them separately. The UniProtKB divides sequences into the following taxonomies: archaea, bacteria, fungi, human, invertebrates, mammals, plants, rodents, vertebrates and viruses. In this division, the vertebrate taxonomy does not contain entries for mammals, which does not contain entries for human and rodents. Invertebrates, on the other hand, contain eukaryotic records excluding vertebrates, fungi and plants. Therefore, I combined the division of UniProtKB into the following taxonomies: archaea, bacteria, fungi, invertebrates, vertebrates, plants and viruses. In these subdatabases, I identified STRs using the GBSC method and clustered them by similar pattern. Finally, I calculated a summary for each dataset presented in the Table 7.2.

Invertebrates have in avarage long STRs with a high standard deviation. Interestingly, in bacteria, GBSC identified STRs in a low percentage of sequences, but these domains are long with a low length variability. We can conclude that bacterial genomes, if they contain STRs, are stable. Vertebrate taxonomy consists of the most diverse set of STRs. In archaea, the number of STRs is low, as expected. In all cases, the median is close to the average. Although the results did not the show error rate of STRs in the UniProtKB/Swiss-Prot database, they demonstrate that variations in repeat lengths are common and errors might often be unnoticed. This is especially crucial in databases that are not as well curated as UniProtKB/Swiss-Prot.

7.3 Analysis of LCR from RNA polymerase

Intrinsically disordered proteins, which frequently overlap with LCRs, often use electrostatic interactions in their functional mechanisms [57, 130]. An example of such a protein is the delta subunit of bacterial RNA polymerase. Its sequence is presented in Figure 7.2. This protein contains the intrinsically disordered region of adjacent LCRs, the first of which is positively charged, followed by an acidic LCR. Experimental methods supported by computational methods are able to precisely describe the structural properties of the K-D/E motif. In this research, the experimental methods were supported by computational analysis of similar protein sequences to check whether such motifs occur frequently in proteins and have similar properties.

>sp|P12464|RPOE_BACSU DNA-directed RNA polymerase subunit delta MGIKQYSQEELKEMALVEIAHELFEEHKKPVPFQELLNEIASLLGVKKEELGDRIAQFYT DLNIDGRFLALSDQTWGLRSWYPYDQLDEETQPTVKAKKKKAKKAVEEDLDLDEFEEIDE DDLDLDEVEEELDLEADDFDEEDLDEDDDDLEIEEDIIDEDDEDYDDEEEEIK

Figure 7.2: DNA-directed RNA polymerase subunit delta (P12464) with K and D/E LCRs highlighted in red and blue, respectively. The first part (red) of the motif is positively charged, the second (blue) is acidic.

7.3.1 Methods

In this section, I demonstrate how to use LCR-BLAST and GBSC to find similarities to K-D/E motif with potentially known properties. As a dataset, I used the UniprotKB/Swiss-Prot. I left the default parameters of LCR-BLAST unchanged. For GBSC, I adjusted the parameters to identify the K-D/E motif of polymerase, and used them to cluster the database by similar repeats. The changed parameters are: alphabet reduction (a), nodes equal edges (n), weight threshold (w), lifetime (l) and maximum number of insertions between adjacent LCRs (max-linked-strs-gap). I reduced the alphabet by merging {D, E} to {D} and changed node behavior (n) since the D/E motif is irregular. The first part of the motif is short, thus the weight threshold also must be low, and I changed it to 3. The LCR is irregular and may have some insertions in between the sub-LCRs, thus I changed the lifetime to 12. I set the maximum number of insertions between adjacent LCRs to 5 in order to allow more insertions than in the query motif. For LCR-BLAST, the query sequences are both LCRs highlighted in Figure 7.2 which I used to search the database. I used both sub-LCRs separately since LCR-BLAST uses local alignment which aligns whole LCR to sub-LCRs only. In the research published in the Journal of American Chemical Society, in addition to LCR-BLAST and GBSC, the MotifLCR method (Ziemska-Legiecka et al., in preparation) was also used to search for similar protein fragments [33]. After adjusting the parameters, I analyzed the results collected from LCR-BLAST and GBSC, and proposed common regular expression for all three methods to identify and present the most similar examples. These examples were further manually curated.

7.3.2 Results

GBSC identified 153 LCRs of K-D/E motif, and LCR-BLAST found 126 proteins containing both K and D/E LCRs. The motif contains mutations that can form repetitive patterns. Even the D/K-rich fragment of RNA polymerase shown in Figure 7.2 contains several mutations to leucine that may be considered repetitive. Therefore, the motifs can join multiple GBSC clusters. In total, 153 sequences found by GBSC were placed in 31 clusters with similar labels. It is worth mentioning that GBSC parameters used in this study were more relaxed than those used in the ,,GBSC - clustering short tandem repeats" chapter. The LCRs found by LCR-BLAST provide more information on motifs rich in lysine or both acidic residues. The results of this method, compared to GBSC, were limited by the composition of the query sequence. To put it in other words, the query sequence contains mutations to leucine, thus if database sequences do not contain such mutations, they are penalized. LCR-BLAST could be additionally adjusted to this specific problem. For instance, it could produce better results by changing the scoring matrix in the way that mutations between glutamic acid and aspartic acid are not penalized, or by creating an artificial query sequence without mutations, and use it for searching. However, the current results provide sufficient data for further analysis.

As a result, we identified K-D/E motifs in 51 proteins, the best matches of which are presented in the Table 7.3. We analyzed the results collected from all three methods, i.e. GBSC, LCR-BLAST and MotifLCR, and created a regular expression that unifies the results from all methods. The regular expression has the following form: K2.,4K2.*[DE]5,. It requires each motif to contain at least four lysines, which can be separated by up to four insertions, and a D/E region of at least 5 residues. The results were further manually curated, annotated and used for similarity analyses.

7.3.3 Summary

I used the methods introduced in this thesis, which are GBSC and LCR-BLAST, to find a similar protein fragments to the K-D/E motif from the DNA-directed RNA polymerase subunit delta. GBSC was used to identify and cluster sequences by a similar pattern, while LCR-BLAST searched for well-aligned protein fragments. The results, along with those from MotifLCR, were used to create a common regular expression representing found motifs. The similarities found with the known properties

Uniprot AC	K-D/E motif
Q93148	KLKKKKKVVASSDEDEDDEDDEEEGRKEMQ GFIADEDDEEEDARSEKSDRSRRSEINDEL DDEDLDLIDENLDRQGERKKNRVRLGDSSD EDEPIRRSNQDDDDLQSERGSDDGDKRRGH GGRGGGGYDSDSDRSEDDFIEDDGDAPRRH RKRHRGDEHIPEGAEDDARDVFGVEDFNFD EFYDDDDGEEGLEDEEEEIIEDDGE
Q54MB8	KKKKKKSKKNKNRHSTEEDETMEDANENLD FATGAGEEEEEEEPE
O60841	KKKKDKKKKKGEKEEKEKEKKKGPSKATVK AMQEALAKLKEEEERQKREEEERIKRLEEL EAKRKEEERLEQEKRERKKQKEKERKERLK KEGKLLTKSQREARARAEATLKLLQAQGVE VPSKDSLPKKRPIYEDKKRKKIPQQLESKE VSESMELCAAVEVMEQGVPEKEETPPPVEP EEEEDTEDAGLDDWEAMASDEETEKVEGNK VHIEVKENPEEEEEEEEEEEEDEESEEEEE EE
P30681	KKGKKKDPNAPKRPPSAFFLFCSENRPKIK IEHPGLSIGDTAKKLGEMWSEQSAKDKQPY EQKAAKLKEKYEKDIAAYRAKGKSEAGKKG PGRPTGSKKKNEPEDEEEEEEEEEEEDDEE EEEDEE
O13741	KSKKKKKKLNDSSDDIEGKYFEELLAEEDE EKDKD
Q8BTT6	KKQKKHLRDFGEEHPFYDRVSKKEAKPQIC QLPESSDSSHSESESESESEQEHVSGYHRLLA TLKNVSEEEEEEEEEEEEEEEEEEEEEEE EDD
A7TJM9	KKGKKNDPVKINGNGDEVAEDLNPDFRFNI DGGELSTNFQGWDFVGDQKENDDDMKKDVD LDGIIRRKGGLLNMAHIEVAEEEEEEEEEE EEEEEEEEEE
Q84MH1	KKKNKKKKSKKTNLKQKKAAEPKPPRDTDD DEDDEEEADDD
Q8VZE7	KKKKKKKSKKVIVKKDKVKSKSIPEDDFDT EDEDLDFED
Q26486	KKAKPDKKAGKNSAPAAESDSDDDDEDQLQ KFLDGEDIDTDENDESFKMNTSAEGDDSDE EDDDEDEEDEEDDDEDDEEEEE
B4MR46	KKKKKTKKRKRKSSDDDDDESSSSDSESSS SSEEEEDEEE
Q70Z19	KKKPKKGSKNAKPITKTEQCESFFNFFSPP QVPEDEEDIDEDAAEELQSLMEQDYDIGST IRDKIISHAVSWFTGEAAEDDFADLEDDDD DDEEDDDDEDEEEEDDEDDEDEEDEDD
Q70Z18	KKKPKKGSKNAKPITKTEQCESFFNFFSPP QVPEDEEDIDEDAAEELQNLMEQDYDIGST IRDKIIPHAVSWFTGEAAEDDYAELEDDED EDDDEEDDEDEDEEEEDEEDDEDEEEDEDE
Q70Z17	KKKPKKGSKNSKPITKIEQCESFFNFFSPP QVPDDEEDIDEDAAEELQNLMEQDYDIGST IRDKIIPHAVSWFTGEAAQDEDYIDLEDDE DEEDDEDEDEDEEDEEEEDEDEDDDDEDE
Q5VND6	KKKPKKGSKNAKPITKTEVCESFFNFFSPP QVPDDDEDIDEDTADELQGQMEHDYDIGTT IRDKIIPHAVSWFTGEAVQAEDFDDMEDDE EDDEDDDEDEEEEEEDEDEDEDDEEE
Q9VNG1	KKLKKKKKKDEKKNLLHRQCDDTEANESDE EEEELRNEELDLEEESQMQHEELSD
A3LXX5	KDKKKHKKRRRRQYDDDVPKDSETKEAAED DEEEEDGEFDENNLENEEDVEDDLAEIDTA NIITTGRRTRRKVIDFAKAAKELDAENGVV REDDEEEEDGEFEVKE
Q9U7C9	KKKFSQKKNHLLNLKKSYQDPEIIAHSRPR KSSGGVSLVEALSDHANYISNLDGFKYYAR ANKSSLNSNATTSGGNNRSIKLNEYKYDDE EEDEEDEDEEDEEEDEEEEEEEEEEED
Q6S003	KGKKKKKSSSPDSLSPNKDDDSSIMIDEDE EDDEEDEDDD
Q23FE2	KPKKKKKKSKKDKQQGDTEKKEEEEGEAED EEEDEEDEEEEE
Q86IZ4	KDKKKKKKNLKIHMRSDSDDNDDNEDEDED ETEE

Table 7.3: Selected K-D/E motifs from the UniProtKB/Swiss-Prot. Adapted from: Kuban et al., Journal of the American Chemical Society, 2019.

were then used to draw other conclusions about the electrostatic interactions of the K-D/E motif.

7.4 SARS-COV-2

At the end of 2019, a new strain of coronavirus was discovered in the city of Wuhan, China. The virus causes severe acute respiratory syndrome, and was spreading rapidly, causing many deaths [131]. The situation was serious, which is why the World Health Organization declared a global pandemic on January 30, 2020. To fight the pandemic, scientists have started a race to create effective vaccines and drugs. These vaccines and drugs are frequently targetted to particular fragments of the viral proteome, known as epitopes. Severe acute respiratory syndrome coronavirus 2 (SARS-CoV-2) has a moderate mutation rate [132]. Therefore, epitopes to which antigens bind should be selected from stable protein fragments that are exposed to antigens. Additionally, similar protein fragments should not exist in the human proteome. Such similarity, known as mimicry, can lead to autoimmune disease [133]. This study aimed to list LCRs in the SARS-CoV-2 proteome and point out these fragments whose similar LCRs can be found in the human proteome.

7.4.1 Results

#	LCR sequence	potein	Number of similar B-cell epitopes	Number of similar T-cell epitopes
1	PPDEDEEEGDCEEEEFE	nsp3	5	2
2	QPEEEQEEDWLDDDSQ	nsp3	3	0
3	MLCCMTSCCSCLKGCCSCGSCC	S protein	2	2
4	GSRGGSQASSRSSSRSRNSSRNSTPGSSRGTSP	N protein	10	9
5	KTFPPTEPKKDKKKKADE	N protein	6	8

Table 7.4: LCRs from SARS-CoV-2 with homologs in the human proteome. Adapted from: Gruca et al., BMC Bioinformatics, 2021.

GBSC, LCR-BLAST and MotifLCR (Ziemska-Legiecka et al., in preparation) were used to search for similar LCRs. The GBSC method was designed to identify and cluster STR, thus it was run independently on the SARS-CoV-2 and human proteomes. LCR-BLAST needs a method for LCR identification to generate query sequences for search. Therefore, SEG with the default parameters was used to retrieve them. Then, LCR-BLAST and MotifLCR were used for all LCRs from both proteomes. SEG found LCRs in the following proteins: nsp2 (one LCR), nsp3 (six LCRs), nsp4 (one LCR), nsp6 (one LCR), nsp7 (two LCRs), nsp8 (two LCRs), S protein (two LCRs), E protein (one LCR), orf7a (one LCR), orf7b (one LCR), N protein (four LCRs) and orf14 (one LCR). In total, the viral proteome contains 23 LCRs, where 5 of them are similar to the human proteome. These 5 LCRs overlap with the 27 B-cell and 21 T-cell epitopes found by [134–136] The table 7.4 shows all LCRs common to virus and human proteomes.

The first two LCRs are located near the N-terminal of the nsp3 protein. Both LCRs are compositionally biased towards aspartic and glutamic acids, which are considered to be physicochemically similar. The first LCR is similar to 5 B-cell epitopes and 2 T-cell epitopes, while the second is similar to only 3 B-cell epitopes. For both LCRs, the maximum sequence coverage is with B-cell epitopes which is equal to 90% and 70%, respectively. The third LCR is located at the C-terminal of the Spike protein and has two similar epitopes in both B-cell and T-cell epitopes. It fully covers two T-cell epitopes. The LCR is compositionally biased towards cysteine, which is interesting because it is a rare and expensive residue in proteomes [137]. Therefore, the probability of a biological role for this fragment in both organisms is relatively high. Serine/arginine-rich proteins often play important roles in proteins, including maintaining gene expression and RNA interactions [61]. Such a fragment is the fourth identified LCR. Due to critical biological roles, this type of LCRs is abundant in proteomes. The LCR is located in the N protein, and overlaps with as many as 10 B-cell epitopes and 9 T-cell epitopes in the list of human proteome. In addition, it fully covers most epitopes. The last LCR is rich in lysine. It also has a large overlap with several epitopes. It covers 6 B-cell and 8 T-cell epitopes, where two B-cell and three T-cell epitopes are fully covered.

7.4.2 Conclusions

The GBSC and LCR-BLAST methods described in this thesis were used to search for similar LCRs in the SARS-CoV-2 and human proteome. Together with MotifLCR, these methods found 5 LCRs in three proteins that are similar to LCRs in the human proteome. 27 B-cells and 21 T-cells epitopes have been found to coincide with the LCRs in the human proteome and should not be considered as targets for vaccine and drug design. This is because vaccines and drugs that target these epitopes may be ineffective or cause autoimmune diseases.

8 Summary and conclusions

As part of the work, I introduced three methods for LCR analyses in proteins and compared existing solutions. At the beginning, I compared existing methods for LCR identification and showed how to combine them to obtain new results using consensus. The selected methods, along with the consensus method, were also used to develop a visualization approach for LCR exploration. Scientists can use visual exploratory analyzes of LCRs to formulate new hypotheses about the biological properties of proteins [28]. LCRs have been ignored for a long time, leading to the hypothesis that existing methods for protein similarity analyses are designed mainly to analyse HCRs of protein sequences [29]. Therefore, I then analyzed three state-of-the-art protein sequence comparison methods to check whether the hypothesis was correct. The results confirmed this hypothesis, showing that these methods were developed for protein domains of high complexity and are not efficient enough to compare LCRs, even if we optimize their parameters. This should be taken into account when using these methods to search for similar protein sequences containing fragments of low complexity. I then showed how to adjust the BLAST parameters to better search for similar LCRs and described LCR-BLAST, which is a BLAST modification for these protein fragments [30]. I demonstrated that each of the proposed modifications changes the method in such a way that it gives better results for LCR. Then I introduced GBSC, which is a new method for identifying and clustering STRs. This method is capable of analyzing a wide range of STRs, from very clear to highly degenerate covering a wide variety of LCR types. Finally, I presented three studies in which I used LCR-BLAST and GBSC in practice. These studies are related to sequencing assembly errors in tandem repeats, electrostatic interactions in RNA Polymerase and common LCRs to human and SARS-CoV-2 proteomes. In the first study, I used GBSC to show differences in STR fragments between protein sequence versions and STR diversity in chosen taxonomies [31]. To analyze the K-D/E domain in RNA polymerase, I used LCR-BLAST and GBSC to find this motif in other proteins. I also showed that GBSC is able to detect irregular LCRs in the case

of a D/E sub-LCR [33]. In the last study, related to SARS-CoV-2, both methods were used to find common LCRs in human and viral proteomes to warn scientists of the possible risk of selecting certain epitopes for drugs and vaccines [32]. The methods developed in the scope of this dissertation have been already used for low complexity sequence analysis in several articles in high impact scientific journals in cooperation with international teams, which indicates the need to develop LCR comparison methods and their practical application in scientific research.

List of Figures

List of Tables

Bibliography

[1] Elisabetta Pizzi and Clara Frontali. Low-complexity regions in plasmodium falciparum proteins. *Genome Research*, 11(2):218–229, 2001.

[2] Núria Radó-Trilla and MMar Albà. Dissecting the role of low-complexity regions in the evolution of vertebrate proteins. *BMC evolutionary biology*, 12(1):1–10, 2012.

[3] Fabia U Battistuzzi, Kristan A Schneider, Matthew K Spencer, David Fisher, Sophia Chaudhry, and Ananias A Escalante. Profiles of low complexity regions in apicomplexa. *BMC evolutionary biology*, 16(1):1–12, 2016.

[4] James AW Stowell, Jane L Wagstaff, Chris H Hill, Minmin Yu, Stephen H McLaughlin, Stefan MV Freund, and Lori A Passmore. A low-complexity region in the yth domain protein mmi1 enhances rna binding. *Journal of Biological Chemistry*, 293(24):9210–9222, 2018.

[5] Bandana Kumari, Ravindra Kumar, Vipin Chauhan, and Manish Kumar. Comparative functional analysis of proteins containing low-complexity predicted amyloid regions. *PeerJ*, 6:e5823, 2018.

[6] Michael P Hughes, Michael R Sawaya, David R Boyer, Lukasz Goldschmidt, Jose A Rodriguez, Duilio Cascio, Lisa Chong, Tamir Gonen, and David S Eisenberg. Atomic structures of low-complexity protein segments reveal kinked β sheets that assemble networks. *Science*, 359(6376):698–701, 2018.

[7] Wilfried Haerty and G Brian Golding. Low-complexity sequences and single amino acid repeats: not just "junk" peptide sequences. *Genome*, 53(10):753–762, 2010.

[8] Sung W Shin and Sam M Kim. A new algorithm for detecting low-complexity regions in protein sequences. *Bioinformatics*, 21(2):160–170, 2005.

[9] John C Wootton and Scott Federhen. Statistics of local complexity in amino acid sequences and sequence databases. *Computers & chemistry*, 17(2):149–163, 1993.

[10] Vasilis J Promponas, Anton J Enright, Sophia Tsoka, David P Kreil, Christophe Leroy, Stavros Hamodrakas, Chris Sander, and Christos A Ouzounis. Cast: an iterative algorithm for the complexity analysis of sequence tracts. *Bioinformatics*, 16(10):915–922, 2000.

[11] M Mar Albà, Roman A Laskowski, and John M Hancock. Detecting cryptically simple protein sequences using the simple algorithm. *Bioinformatics*, 18(5):672–678, 2002.

[12] Xuehui Li and Tamer Kahveci. A novel algorithm for identifying low-complexity regions in a protein sequence. *Bioinformatics*, 22(24):2980–2987, 2006.

[13] Paul M Harrison. flps: Fast discovery of compositional biases for the protein universe. *Bmc Bioinformatics*, 18(1):1–9, 2017.

[14] Sean M Cascarina, David C King, Erin Osborne Nishimura, and Eric D Ross. Lcd-composer: an intuitive, composition-centric method enabling the identification and detailed functional mapping of low-complexity domains. *NAR genomics and bioinformatics*, 3(2):lqab048, 2021.

[15] Aaron M Newman and James B Cooper. Xstream: a practical algorithm for identification and architecture modeling of tandem repeats in protein sequences. *BMC bioinformatics*, 8(1):1–19, 2007.

[16] Julien Jorda and Andrey V Kajava. T-reks: identification of tandem repeats in sequences with a k-means based algorithm. *Bioinformatics*, 25(20):2632–2638, 2009.

[17] Stephen F Altschul, Warren Gish, Webb Miller, Eugene W Myers, and David J Lipman. Basic local alignment search tool. *Journal of molecular biology*, 215(3):403–410, 1990.

[18] Temple F Smith and Michael S Waterman. Identification of common molecular subsequences. *Journal of molecular biology*, 147(1):195–197, 1981.

[19] Yi-Kuo Yu and Stephen F Altschul. The construction of amino acid substitution matrices for the comparison of proteins with non-standard compositions. *Bioinformatics*, 21(7):902–911, 2005.

[20] Steven Henikoff and Jorja G Henikoff. Amino acid substitution matrices from protein blocks. *Proceedings of the National Academy of Sciences*, 89(22):10915–10919, 1992.

[21] Stephen F Altschul, Thomas L Madden, Alejandro A Schäffer, Jinghui Zhang, Zheng Zhang, Webb Miller, and David J Lipman. Gapped blast and psi-blast: a new generation of protein database search programs. *Nucleic acids research*, 25(17):3389–3402, 1997.

[22] Michael Remmert, Andreas Biegert, Andreas Hauser, and Johannes Söding. Hhblits: lightning-fast iterative protein sequence searching by hmm-hmm alignment. *Nature methods*, 9(2):173–175, 2012.

[23] Sean R Eddy. Accelerated profile hmm searches. *PLoS computational biology*, 7(10):e1002195, 2011.

[24] Johannes Söding. Protein homology detection by hmm–hmm comparison. *Bioinformatics*, 21(7):951–960, 2005.

[25] Juan Carlos Aledo. A census of human methionine-rich prion-like domain-containing proteins. *Antioxidants*, 11(7):1289, 2022.

[26] Erik W Martin and Tanja Mittag. Relationship of sequence and phase separation in protein low-complexity regions. *Biochemistry*, 57(17):2478–2487, 2018.

[27] Weizhong Li and Adam Godzik. Cd-hit: a fast program for clustering and comparing large sets of protein or nucleotide sequences. *Bioinformatics*, 22(13):1658–1659, 2006.

[28] Patryk Jarnot, Joanna Ziemska-Legiecka, Laszlo Dobson, Matthew Merski, Pablo Mier, Miguel A Andrade-Navarro, John M Hancock, Zsuzsanna Dosztányi, Lisanna Paladin, Marco Necci, et al. Platoloco: the first web meta-server for visualization and annotation of low complexity regions in proteins. *Nucleic acids research*, 48(W1):W77–W84, 2020.

[29] Patryk Jarnot, Joanna Ziemska-Legiecka, Marcin Grynberg, and Aleksandra Gruca. Insights from analyses of low complexity regions with canonical methods for protein sequence comparison. *Briefings in Bioinformatics*, 23(5):bbac299, 2022.

[30] Patryk Jarnot, Joanna Ziemska-Legiecka, Marcin Grynberg, and Aleksandra Gruca. Lcr-blast—a new modification of blast to search for similar low complexity regions in protein sequences. In *International Conference on Man–Machine Interactions*, pages 169–180. Springer, 2019.

[31] Ole K Tørresen, Bastiaan Star, Pablo Mier, Miguel A Andrade-Navarro, Alex Bateman, Patryk Jarnot, Aleksandra Gruca, Marcin Grynberg, Andrey V Kajava, Vasilis J Promponas, et al. Tandem repeats lead to sequence assembly errors and impose multi-level challenges for genome and protein databases. *Nucleic acids research*, 47(21):10994–11006, 2019.

[32] Aleksandra Gruca, Joanna Ziemska-Legiecka, Patryk Jarnot, Elzbieta Sarnowska, Tomasz J Sarnowski, and Marcin Grynberg. Common low complexity regions for sars-cov-2 and human proteomes as potential multidirectional risk factor in vaccine development. *BMC bioinformatics*, 22(1):1–18, 2021.

[33] Vojtěch Kubáň, Pavel Srb, Hana Štégnerová, Petr Padrta, Milan Zachrdla, Zuzana Jaseňáková, Hana Šanderová, Dragana Vítovská, Libor Krasny, Tomáš Koval', et al. Quantitative conformational analysis of functionally important electrostatic interactions in the intrinsically disordered region of delta subunit of bacterial rna polymerase. *Journal of the American Chemical Society*, 141(42):16817–16828, 2019.

[34] Marcos Gil-Garcia, Valentín Iglesias, Irantzu Pallarès, and Salvador Ventura. Prion-like proteins: from computational approaches to proteome-wide analysis. *FEBS open bio*, 11(9):2400–2417, 2021.

[35] Nicolas Leurs, Camille Martinand-Mari, Sylvain Marcellini, and Mélanie Debiais-Thibaud. Parallel evolution of ameloblastic scpp genes in bony and cartilaginous vertebrates. *Molecular biology and evolution*, 39(5):msac099, 2022.

[36] Vladimir N Uversky. Natively unfolded proteins: a point where biology waits for physics. *Protein science*, 11(4):739–756, 2002.

[37] Predrag Radivojac, Lilia M Iakoucheva, Christopher J Oldfield, Zoran Obradovic, Vladimir N Uversky, and A Keith Dunker. Intrinsic disorder and functional proteomics. *Biophysical journal*, 92(5):1439–1456, 2007.

[38] Diethard Tautz, Martin Trick, and Gabriel A Dover. Cryptic simplicity in dna is a major source of genetic variation. *Nature*, 322(6080):652–656, 1986.

[39] John M Hancock and John S Armstrong. Simple34: an improved and enhanced implementation for vax and sun computers of the simple algorithm for analysis of clustered repetitive motifs in nucleotide sequences. *Bioinformatics*, 10(1):67–70, 1994.

[40] Olivier Navaud, Patrick Dabos, Elodie Carnus, Dominique Tremousaygue, and Christine Hervé. Tcp transcription factors predate the emergence of land plants. *Journal of molecular evolution*, 65(1):23–33, 2007.

[41] Elke Schaper and Maria Anisimova. The evolution and function of protein tandem repeats in plants. *New Phytologist*, 206(1):397–410, 2015.

[42] Luke CM Mackinder, Moritz T Meyer, Tabea Mettler-Altmann, Vivian K Chen, Madeline C Mitchell, Oliver Caspari, Elizabeth S Freeman Rosenzweig, Leif Pallesen, Gregory Reeves, Alan Itakura, et al. A repeat protein links rubisco to form the eukaryotic carbon-concentrating organelle. *Proceedings of the National Academy of Sciences*, 113(21):5958–5963, 2016.

[43] Macy L Sprunger and Meredith E Jackrel. Prion-like proteins in phase separation and their link to disease. *Biomolecules*, 11(7):1014, 2021.

[44] William R Pearson and David J Lipman. Improved tools for biological sequence comparison. *Proceedings of the National Academy of Sciences*, 85(8):2444–2448, 1988.

[45] Fabian Sievers and Desmond G Higgins. Clustal omega, accurate alignment of very large numbers of sequences. In *Multiple sequence alignment methods*, pages 105–116. Springer, 2014.

[46] Martin Steinegger and Johannes Söding. Mmseqs2 enables sensitive protein sequence searching for the analysis of massive data sets. *Nature biotechnology*, 35(11):1026–1028, 2017.

[47] Uniprot: the universal protein knowledgebase in 2021. *Nucleic acids research*, 49(D1):D480–D489, 2021.

[48] John Jumper, Richard Evans, Alexander Pritzel, Tim Green, Michael Figurnov, Olaf Ronneberger, Kathryn Tunyasuvunakool, Russ Bates, Augustin Žídek, Anna Potapenko, et al. Highly accurate protein structure prediction with alphafold. *Nature*, 596(7873):583–589, 2021.

[49] Anton J Enright, Stijn Van Dongen, and Christos A Ouzounis. An efficient algorithm for large-scale detection of protein families. *Nucleic acids research*, 30(7):1575–1584, 2002.

[50] Alessandro Varani, Susu He, Patricia Siguier, Karen Ross, and Michael Chandler. The is6 family, a clinically important group of insertion sequences including is26. *Mobile Dna*, 12(1):1–18, 2021.

[51] Sávio Souza Costa, Luís Carlos Guimarães, Artur Silva, Siomar Castro Soares, and Rafael Azevedo Baraúna. First steps in the analysis of prokaryotic pangenomes. *Bioinformatics and Biology Insights*, 14:1177932220938064, 2020.

[52] Anna Sorushanova, Luis M Delgado, Zhuning Wu, Naledi Shologu, Aniket Kshirsagar, Rufus Raghunath, Anne M Mullen, Yves Bayon, Abhay Pandit, Michael Raghunath, et al. The collagen suprafamily: from biosynthesis to advanced biomaterial development. *Advanced materials*, 31(1):1801651, 2019.

[53] Kun-Hua Yu, Mei-Yu Huang, Yi-Ru Lee, Yu-Kie Lin, Hau-Ren Chen, and Cheng-I Lee. The effect of octapeptide repeats on prion folding and misfolding. *International Journal of Molecular Sciences*, 22(4):1800, 2021.

[54] Katsunori Hase, Viorica Raluca Contu, Chihana Kabuta, Ryohei Sakai, Masayuki Takahashi, Naoyuki Kataoka, Fumihiko Hakuno, Shin-Ichiro Takahashi, Yuuki Fujiwara, Keiji Wada, et al. Cytosolic domain of sidt2 carries an arginine-rich motif that binds to rna/dna and is important for the direct transport of nucleic acids into lysosomes. *Autophagy*, 16(11):1974–1988, 2020.

[55] Bandana Kumari, Ravindra Kumar, and Manish Kumar. Low complexity and disordered regions of proteins have different structural and amino acid preferences. *Molecular BioSystems*, 11(2):585–594, 2015.

[56] Pedro Romero, Zoran Obradovic, Xiaohong Li, Ethan C Garner, Celeste J Brown, and A Keith Dunker. Sequence complexity of disordered protein. *Proteins: Structure, Function, and Bioinformatics*, 42(1):38–48, 2001.

[57] Sean M Cascarina, Mikaela R Elder, and Eric D Ross. Atypical structural tendencies among low-complexity domains in the protein data bank proteome. *PLoS computational biology*, 16(1):e1007487, 2020.

[58] Pablo Mier, Lisanna Paladin, Stella Tamana, Sophia Petrosian, Borbála Hajdu-Soltész, Annika Urbanek, Aleksandra Gruca, Dariusz Plewczynski, Marcin Grynberg, Pau Bernadó, et al. Disentangling the complexity of low complexity proteins. *Briefings in Bioinformatics*, 21(2):458–472, 2020.

[59] Pablo Mier and Miguel A Andrade-Navarro. Assessing the low complexity of protein sequences via the low complexity triangle. *PloS one*, 15(12):e0239154, 2020.

[60] Andrés J Gutiérrez Escobar, Aylan Farid Arenas, and Jorge Enrique Gómez-Marin. Molecular evolution of serine/arginine splicing factors family (sr) by positive selection. *In silico biology*, 6(4):347–350, 2006.

[61] Peter J Shepard and Klemens J Hertel. The sr protein family. *Genome biology*, 10(10):1–9, 2009.

[62] Andrej Michalik and Christine Van Broeckhoven. Pathogenesis of polyglutamine disorders: aggregation revisited. *Human molecular genetics*, 12(suppl_2):R173–R186, 2003.

[63] Aislinn J Williams and Henry L Paulson. Polyglutamine neurodegeneration: protein misfolding revisited. *Trends in neurosciences*, 31(10):521–528, 2008.

[64] Emma L Bunting, Joseph Hamilton, and Sarah J Tabrizi. Polyglutamine diseases. *Current Opinion in Neurobiology*, 72:39–47, 2022.

[65] Barbara Brodsky and Anton V Persikov. Molecular structure of the collagen triple helix. *Advances in protein chemistry*, 70:301–339, 2005.

[66] Lisanna Paladin, Mathieu Schaeffer, Pascale Gaudet, Monique Zahn-Zabal, Pierre-André Michel, Damiano Piovesan, Silvio CE Tosatto, and Amos Bairoch.

The feature-viewer: a visualization tool for positional annotations on a sequence. *Bioinformatics*, 36(10):3244–3245, 2020.

[67] Lilia M Iakoucheva, Predrag Radivojac, Celeste J Brown, Timothy R O'Connor, Jason G Sikes, Zoran Obradovic, and A Keith Dunker. The importance of intrinsic disorder for protein phosphorylation. *Nucleic acids research*, 32(3):1037–1049, 2004.

[68] Jie Liu, Qi Zheng, Yiqun Deng, Chao-Sheng Cheng, Neville R Kallenbach, and Min Lu. A seven-helix coiled coil. *Proceedings of the National Academy of Sciences*, 103(42):15457–15462, 2006.

[69] Kevin Kok-Phen Yan, Ikenna Obi, and Nasim Sabouri. The rgg domain in the c-terminus of the dead box helicases dbp2 and ded1 is necessary for g-quadruplex destabilization. *Nucleic Acids Research*, 49(14):8339–8354, 2021.

[70] Liwei Ma, Wenting Zhao, Quanhui Zheng, Tianda Chen, Ji Qi, Guodong Li, and Tanjun Tong. Ribosomal l1 domain and lysine-rich region are essential for csig/rsl1d1 to regulate proliferation and senescence. *Biochemical and biophysical research communications*, 469(3):593–598, 2016.

[71] Elke Schaper, Alexander Korsunsky, Jūlija Pečerska, Antonio Messina, Riccardo Murri, Heinz Stockinger, Stefan Zoller, Ioannis Xenarios, and Maria Anisimova. Tral: tandem repeat annotation library. *Bioinformatics*, 31(18):3051–3053, 2015.

[72] Andreas Biegert and Johannes Söding. De novo identification of highly diverged protein repeats by probabilistic consistency. *Bioinformatics*, 24(6):807–814, 2008.

[73] Hongsun Park, Tomoyuki Yamanaka, and Nobuyuki Nukina. Proteomic analysis of heat-stable proteins revealed an increased proportion of proteins with compositionally biased regions. *Scientific reports*, 12(1):1–13, 2022.

[74] Leandro Cruz Rodríguez, Nahuel N Foressi, and M Soledad Celej. Modulation of α-synuclein phase separation by biomolecules. *Biochimica et Biophysica Acta (BBA)-Proteins and Proteomics*, page 140885, 2022.

[75] Ioannis Kirmitzoglou and Vasilis J Promponas. Lcr-exxxplorer: a web platform to search, visualize and share data for low complexity regions in protein sequences. *Bioinformatics*, 31(13):2208–2210, 2015.

[76] Sean M Cascarina and Eric D Ross. The lcd-composer webserver: high-specificity identification and functional analysis of low-complexity domains in proteins. *Bioinformatics*, 38(24):5446–5448, 2022.

[77] Sean R Eddy et al. Multiple alignment using hidden markov models. In *Ismb*, volume 3, pages 114–120, 1995.

[78] Alex Bateman, Maria-Jesus Martin, Sandra Orchard, Michele Magrane, Shadab Ahmad, Emanuele Alpi, Emily H Bowler-Barnett, Ramona Britto, Hema Bye-A-Jee, Austra Cukura, et al. Uniprot: the universal protein knowledgebase in 2023. *Nucleic Acids Research*, 2022.

[79] Núria Radó-Trilla, Krisztina Arató, Cinta Pegueroles, Alicia Raya, Susana de la Luna, and M Mar Albà. Key role of amino acid repeat expansions in the functional diversification of duplicated transcription factors. *Molecular biology and evolution*, 32(9):2263–2272, 2015.

[80] Sean R Eddy. A new generation of homology search tools based on probabilistic inference. In *Genome Informatics 2009: Genome Informatics Series Vol. 23*, pages 205–211. World Scientific, 2009.

[81] Milot Mirdita, Lars Von Den Driesch, Clovis Galiez, Maria J Martin, Johannes Söding, and Martin Steinegger. Uniclust databases of clustered and deeply annotated protein sequences and alignments. *Nucleic acids research*, 45(D1):D170–D176, 2017.

[82] Giddy Landan and Dan Graur. Characterization of pairwise and multiple sequence alignment errors. *Gene*, 441(1-2):141–147, 2009.

[83] N Srinivasan and A Krupa. A genomic perspective of protein kinases in plasmodium falciparum. *Proteins: Structure, Function, and Bioinformatics*, 58(1):180–189, 2005.

[84] Narcis Fernandez-Fuentes, Carlos J Madrid-Aliste, Brajesh Kumar Rai, J Eduardo Fajardo, and András Fiser. M4t: a comparative protein structure modeling server. *Nucleic acids research*, 35(suppl_2):W363–W368, 2007.

[85] William R Pearson. Selecting the right similarity-scoring matrix. *Current protocols in bioinformatics*, 43(1):3–5, 2013.

[86] Michael Wolf, Petra Lommes, Elisabeth Sock, Simone Reiprich, Ralf P Friedrich, Jana Kriesch, C Claus Stolt, John R Bermingham Jr, and Michael Wegner. Replacement of related pou transcription factors leads to severe defects in mouse forebrain development. *Developmental biology*, 332(2):418–428, 2009.

[87] Sabine Heger, Claudio Mastronardi, Gregory A Dissen, Alejandro Lomniczi, Ricardo Cabrera, Christian L Roth, Heike Jung, Francesco Galimi, Wolfgang Sippell, Sergio R Ojeda, et al. Enhanced at puberty 1 (eap1) is a new transcriptional regulator of the female neuroendocrine reproductive axis. *The Journal of clinical investigation*, 117(8):2145–2154, 2007.

[88] Eulàlia Salichs, Alice Ledda, Loris Mularoni, M Mar Albà, and Susana de la Luna. Genome-wide analysis of histidine repeats reveals their role in the localization of human proteins to the nuclear speckles compartment. *PLoS genetics*, 5(3):e1000397, 2009.

[89] Andreas H Rasmussen, Hanne B Rasmussen, and Asli Silahtaroglu. The dlgap family: neuronal expression, function and role in brain disorders. *Molecular Brain*, 10(1):1–13, 2017.

[90] Manuela Murariu, Ecaterina Stela Dragan, and Gabi Drochioiu. Model peptide-based system used for the investigation of metal ions binding to histidine-containing polypeptides. *Biopolymers: Original Research on Biomolecules*, 93(6):497–508, 2010.

[91] Jose Miguel Laffita-Mesa, Martin Paucar, and Per Svenningsson. Ataxin-2 gene: a powerful modulator of neurological disorders. *Current Opinion in Neurology*, 34(4):578, 2021.

[92] Rakesh Trivedi and Hampapathalu Adimurthy Nagarajaram. Substitution scoring matrices for proteins-an overview. *Protein Science*, 29(11):2150–2163, 2020.

[93] Pauline C Ng, Jorja G Henikoff, and Steven Henikoff. Phat: a transmembrane-specific substitution matrix. *Bioinformatics*, 16(9):760–766, 2000.

[94] Wing-Cheong Wong, Sebastian Maurer-Stroh, and Frank Eisenhaber. Not all transmembrane helices are born equal: Towards the extension of the sequence homology concept to membrane proteins. *Biology direct*, 6(1):1–30, 2011.

[95] Martin H Schaefer, Erich E Wanker, and Miguel A Andrade-Navarro. Evolution and function of cag/polyglutamine repeats in protein–protein interaction networks. *Nucleic acids research*, 40(10):4273–4287, 2012.

[96] Donald R Demuth and Douglas C Irvine. Structural and functional variation within the alanine-rich repetitive domain of streptococcal antigen i/ii. *Infection and immunity*, 70(11):6389–6398, 2002.

[97] Punto Bawono, Maurits Dijkstra, Walter Pirovano, Anton Feenstra, Sanne Abeln, and Jaap Heringa. Multiple sequence alignment. In *Bioinformatics*, pages 167–189. Springer, 2017.

[98] Vincent Ranwez and Nathalie Chantret. Strengths and limits of multiple sequence alignment and filtering methods, 2020.

[99] Robert Hubley, Travis J Wheeler, and Arian FA Smit. Accuracy of multiple sequence alignment methods in the reconstruction of transposable element families. *NAR genomics and bioinformatics*, 4(2):lqac040, 2022.

[100] Bruno Almeida, Sara Fernandes, Isabel A Abreu, and Sandra Macedo-Ribeiro. Trinucleotide repeats: a structural perspective. *Frontiers in neurology*, 4:76, 2013.

[101] Megerditch Kiledjian and Gideon Dreyfuss. Primary structure and binding activity of the hnrnp u protein: binding rna through rgg box. *The EMBO journal*, 11(7):2655–2664, 1992.

[102] Vladimir Espinosa Angarica, Salvador Ventura, and Javier Sancho. Discovering putative prion sequences in complete proteomes using probabilistic representations of q/n-rich domains. *BMC genomics*, 14(1):1–17, 2013.

[103] MV Borca, C Carrillo, L Zsak, WW Laegreid, GF Kutish, JG Neilan, TG Burrage, and DL Rock. Deletion of a cd2-like gene, 8-dr, from african swine fever virus affects viral infection in domestic swine. *Journal of virology*, 72(4):2881–2889, 1998.

[104] Shunli Yang, Xinming Zhang, Yuying Cao, Shuo Li, Junjun Shao, Shiqi Sun, Huichen Guo, and Shuanghui Yin. Identification of a new cell-penetrating peptide derived from the african swine fever virus cd2v protein. *Drug delivery*, 28(1):957–962, 2021.

[105] Aidong Yuan, Takahiro Sasaki, Asok Kumar, Corrinne M Peterhoff, Mala V Rao, Ronald K Liem, Jean-Pierre Julien, and Ralph A Nixon. Peripherin is a subunit of peripheral nerve neurofilaments: implications for differential vulnerability of cns and peripheral nervous system axons. *Journal of Neuroscience*, 32(25):8501–8508, 2012.

[106] Thomas Henkel, Paul D Ling, S Diane Hayward, and Michael Gregory Peterson. Mediation of epstein-barr virus ebna2 transactivation by recombination signal-binding protein jκ. *Science*, 265(5168):92–95, 1994.

[107] Shih-Hsuan Pan, Chia-Ching Tai, Chang-Shen Lin, Wei-Bin Hsu, Shu-Fan Chou, Chih-Chang Lai, Jen-Yang Chen, Hwei-Fang Tien, Fen-Yu Lee, and Won-Bo Wang. Epstein-barr virus nuclear antigen 2 disrupts mitotic checkpoint and causes chromosomal instability. *Carcinogenesis*, 30(2):366–375, 2009.

[108] Weibing Yang, Sulin Ren, Xiaoming Zhang, Mingjun Gao, Shenghai Ye, Yongbin Qi, Yiyan Zheng, Juan Wang, Longjun Zeng, Qun Li, et al. Bent uppermost internode1 encodes the class ii formin fh5 crucial for actin organization and rice development. *The Plant Cell*, 23(2):661–680, 2011.

[109] Masato Kato, Xiaoming Zhou, and Steven L McKnight. How do protein domains of low sequence complexity work? *RNA*, 28(1):3–15, 2022.

[110] Yingbo Liang, Shichun Cui, Xiaoli Tang, Yi Zhang, Dewen Qiu, Hongmei Zeng, Lihua Guo, Jingjing Yuan, and Xiufen Yang. An asparagine-rich protein nbnrp1 modulate verticillium dahliae protein pevd1-induced cell death and disease resistance in nicotiana benthamian a. *Frontiers in plant science*, 9:303, 2018.

[111] Andrey V Kajava. Tandem repeats in proteins: from sequence to structure. *Journal of structural biology*, 179(3):279–288, 2012.

[112] Mukund V Katti, R Sami-Subbu, Prabhakar K Ranjekar, and Vidya S Gupta. Amino acid repeat patterns in protein sequences: their diversity and structural-functional implications. *Protein Science*, 9(6):1203–1209, 2000.

[113] Nicolaas Govert De Bruijn. A combinatorial problem. In *Proc. Koninklijke Nederlandse Academie van Wetenschappen*, volume 49, pages 758–764, 1946.

[114] Pablo Mier and Miguel A Andrade-Navarro. Regions with two amino acids in protein sequences: A step forward from homorepeats into the low complexity landscape. *Computational and Structural Biotechnology Journal*, 20:5516–5523, 2022.

[115] Xiang Li, Changyu Hu, and Jie Liang. Simplicial edge representation of protein structures and alpha contact potential with confidence measure. *Proteins: Structure, Function, and Bioinformatics*, 53(4):792–805, 2003.

[116] Pushkar Sharma, Joseph J Barchi, Xiaolin Huang, Niranjana D Amin, Howard Jaffe, and Harish C Pant. Site-specific phosphorylation of lys-ser-pro repeat peptides from neurofilament h by cyclin-dependent kinase 5: structural basis for substrate recognition. *Biochemistry*, 37(14):4759–4766, 1998.

[117] Norio Matsushima, Hitoshi Yoshida, Yasuhiro Kumaki, Masakatsu Kamiya, Takanori Tanaka, Yoshinobu Izumi, and Robert H Kretsinger. Flexible structures and ligand interactions of tandem repeats consisting of proline, glycine, asparagine, serine, and/or threonine rich oligopeptides in proteins. *Current Protein and Peptide Science*, 9(6):591–610, 2008.

[118] Robert C Edgar and Serafim Batzoglou. Multiple sequence alignment. *Current opinion in structural biology*, 16(3):368–373, 2006.

[119] SAA Sajadi et al. Metal ion-binding properties of l-glutamic acid and l-aspartic acid, a comparative investigation. *Natural Science*, 2(02):85, 2010.

[120] Alfredo Castello. The emerging universe of rna-binding proteins. *The Biochemist*, 37(2):33–38, 2015.

[121] Matthew D Shoulders and Ronald T Raines. Collagen structure and stability. *Annual review of biochemistry*, 78:929, 2009.

[122] Pantelis Linardatos, Vasilis Papastefanopoulos, and Sotiris Kotsiantis. Explainable ai: A review of machine learning interpretability methods. *Entropy*, 23(1):18, 2020.

[123] David R Bentley, Shankar Balasubramanian, Harold P Swerdlow, Geoffrey P Smith, John Milton, Clive G Brown, Kevin P Hall, Dirk J Evers, Colin L Barnes, Helen R Bignell, et al. Accurate whole human genome sequencing using reversible terminator chemistry. *nature*, 456(7218):53–59, 2008.

[124] John Eid, Adrian Fehr, Jeremy Gray, Khai Luong, John Lyle, Geoff Otto, Paul Peluso, David Rank, Primo Baybayan, Brad Bettman, et al. Real-time dna sequencing from single polymerase molecules. *Science*, 323(5910):133–138, 2009.

[125] Felix Olasagasti, Kate R Lieberman, Seico Benner, Gerald M Cherf, Joseph M Dahl, David W Deamer, and Mark Akeson. Replication of individual dna molecules under electronic control using a protein nanopore. *Nature nanotechnology*, 5(11):798–806, 2010.

[126] Daniel R Zerbino and Ewan Birney. Velvet: algorithms for de novo short read assembly using de bruijn graphs. *Genome research*, 18(5):821–829, 2008.

[127] Sergey Koren, Brian P Walenz, Konstantin Berlin, Jason R Miller, Nicholas H Bergman, and Adam M Phillippy. Canu: scalable and accurate long-read assembly via adaptive k-mer weighting and repeat separation. *Genome research*, 27(5):722–736, 2017.

[128] Girum Fitihamlak Ejigu and Jaehee Jung. Review on the computational genome annotation of sequences obtained by next-generation sequencing. *Biology*, 9(9):295, 2020.

[129] Timo Lassmann and Erik LL Sonnhammer. Kalign–an accurate and fast multiple sequence alignment algorithm. *BMC bioinformatics*, 6(1):1–9, 2005.

[130] Jing Yang, Yifan Zeng, Yunfei Liu, Meng Gao, Sen Liu, Zhengding Su, and Yongqi Huang. Electrostatic interactions in molecular recognition of intrinsically disordered proteins. *Journal of Biomolecular Structure and Dynamics*, 38(16):4883–4894, 2020.

[131] Who director-general's opening remarks at the media briefing on covid-19 - 11 march 2020. https://www.who.int/director-general/speeches/detail/who-director-general-s-opening-remarks-at-the-media-briefing-on-covid-19---11-march-2020. Accessed: 2023-02-12.

[132] Nadim Sharif and Shuvra Kanti Dey. Impact of population density and weather on covid-19 pandemic and sars-cov-2 mutation frequency in bangladesh. *Epidemiology & Infection*, 149, 2021.

[133] Janet M Davies. Molecular mimicry: can epitope mimicry induce autoimmune disease? *Immunology and cell Biology*, 75(2):113–126, 1997.

[134] Syed Faraz Ahmed, Ahmed A Quadeer, and Matthew R McKay. Preliminary identification of potential vaccine targets for the covid-19 coronavirus (sars-cov-2) based on sars-cov immunological studies. *Viruses*, 12(3):254, 2020.

[135] Alba Grifoni, John Sidney, Yun Zhang, Richard H Scheuermann, Bjoern Peters, and Alessandro Sette. A sequence homology and bioinformatic approach can predict candidate targets for immune responses to sars-cov-2. *Cell host & microbe*, 27(4):671–680, 2020.

[136] Chunguang Liang, Elena Bencurova, Edita Sarukhanyan, Priya Neurgaonkar, Carsten Scheller, and Thomas Dandekar. Population-predicted mhcii-epitope presentation of sars-cov-2 spike protein correlates to the case fatality rates of covid-19 in different countries. *Available at SSRN 3576817*, 2020.

[137] Dawn J Brooks and Jacques R Fresco. Increased frequency of cysteine, tyrosine, and phenylalanine residues since the last universal ancestor. *Molecular & Cellular Proteomics*, 1(2):125–131, 2002.