EAI Endorsed Transactions
on Bioengineering and Bioinformatics
Research Article EAI.EU

Parallel Implementation of String-Based Clustering for HT-SELEX Data

Shintaro Kato[1,2*], Takayoshi Ono[2], Masaki Ito[2], Koichi Ito[2*], Hirotaka Minagawa[1], Katsunori Horii[1], Ikuo Shiratori[1], Iwao Waga[1], and Takafumi Aoki[2]

[1]NEC Solution Innovators, Ltd.
1-18-7, Shinkiba, Koto-ku, Tokyo, 136-8627, Japan.
[2]Graduate School of Information Sciences, Tohoku University,
6-6-05, Aramaki Aza Aoba, Aoba-ku, Sendai-shi, Miyagi, 980-8579, Japan.

Abstract

INTRODUCTION: A clustering method for HT-SELEX is crucial for selecting different types of aptamer candidates. We have developed FSBC method for HT-SELEX data implemented in R. FSBC exhibited the highest accuracy of sequence clustering compared with conventional methods, while the processing time of FSBC is longer than AptaCluster.
OBJECTIVES: The objective of this study is to improve the processing time of FSBC.
METHODS: We propose pFSBC, which reduces the processing time of ORS estimation in FSBC by introducing parallel implementation.
RESULTS: The processing time and clustering accuracy were evaluated with the last round of NCBI SRA data of SRR3279661 from BioProject PRJNA315881 comparing with other conventional clustering methods. We demonstrated that pFSBC exhibited the highest clustering accuracy and the shortest processing time.
CONCLUSION: We expect that pFSBC will help to avoid the time-consuming clustering task, and it will provide accurate clustering results for the HT-SELEX data.

Received on 30 June 2020; accepted on 01 October 2020; published on 19 October 2020

Keywords: sequence analysis, clustering, SELEX, next-generation sequencing, aptamer, parallel implementation

doi:10.4108/eai.19-10-2020.166664

1. Introduction

Nucleic acid aptamers [1], which are made from single-stranded DNA or RNA, fold into a specific three-dimensional structure and bind to the target molecules with high affinity and specificity. Aptamers have been used for different types of applications such as therapeutics [2], diagnostics [3], multi-protein quantitative measurement [4], and sensors [5] owing to a wide variety of the target molecules, e.g. proteins [6], small molecules [7], ions [8], toxins [9], and cells [10].

Systematic evolution of ligands by exponential enrichment (SELEX) has been used to determine the aptamer sequences from an initial random oligonucleotide pool filled with $10^{14\sim15}$ oligonucleotide molecules [11]. Each oligonucleotide molecule has 40~60 random regions flanked by primer sequences for polymerase chain reaction (PCR) amplification. SELEX is a repetition method and each round of SELEX consists of the following steps; the oligonucleotide selection with target molecules, the washing off of non-binding sequences, the elution of oligonucleotides from target molecules, and the amplification of selected oligonucleotides by PCR. After performing the above procedure for a sufficient number of rounds, around 10 rounds in general, the oligonucleotide pool is filled with enriched aptamers. Oligonucleotide sequencer is applied to such aptamer-enriched pools to obtain sequence data. Then, dozens of aptamer candidates are selected from sequence data and chemically synthesized for evaluation of binding affinity with experimental analysis. If the selected sequence shows enough

*Corresponding authors. Email: katou-s-mxn@nec.com, ito@aoki.ecei.tohoku.ac.jp

affinity to the target molecules, the sequence can be defined as the aptamer.

Recently, next-generation sequencing (NGS) has been used for obtaining a large amount of sequence data from oligonucleotide pools of SELEX. This combination of SELEX and NGS is called as high-throughput SELEX (HT-SELEX). Fig. 1 illustrates the procedure for HT-SELEX. It is possible to observe a huge amount of sequence data from SELEX pools using NGS. Owing to such a huge amount of sequence data, HT-SELEX enables to select different types of aptamer candidates. However, HT-SELEX data also includes non-aptamer sequences that are enriched in the oligonucleotide pools but do not bind to the target molecules with strong affinity. The number of sequences for verification with experimental analysis is limited due to the oligonucleotide synthesizing cost and time. Therefore, the sequence clustering method is important to estimate groups of aptamers and noise sequences and is also effective in reducing similar sequences from aptamer candidates.

We have developed a fast string-based clustering (FSBC) method for HT-SELEX data [12][1], which was implemented in R [13]. In general, aptamers include specific sequence regions, which are necessary to bind to the target molecules, and other parts of the sequence can be trimmed. For example, the length of Macugen [14], which is a drug for age-related macular degeneration, has only 27 nucleotides. The length of binding regions depends on target molecules, binding style, and/or epitopes of target molecules. The binding region of the aptamer is enriched during the SELEX process as well as the full-length aptamers. We define the enriched binding regions during the SELEX process as over-represented strings (ORS) in FSBC. FSBC estimates ORS with different lengths and makes sequence clusters according to the estimated ORS. Clustering accuracy and processing time of FSBC were compared with four conventional clustering methods: AptaTrace [15], AptaCluster [16], APTANI [17], and FASTAptamer [18], using H1 whole cell SELEX-Seq data [19]. FSBC exhibited the highest accuracy of finding sequences with the desired aptamers in all the methods and was faster than AptaTrace, FASTAptamer, and APTANI, while FSBC was slower than AptaCluster. Hence, the processing time of FSBC still needs to be improved.

In this study, We re-implemented the FSBC algorithm with parallel processing using Python with multi-threading, which is called pFSBC, to improve the processing time. pFSBC was applied to the fifth-round of NCBI SRA data of SRR3279661 from BioProject PRJNA315881 [20, 21] and was compared with FASTAptamer, the Usearch programs (Uclust and Unoise), and AptaCluster. We demonstrated through the experiments that pFSBC exhibited the shortest processing time and the highest accuracy in all the methods. We also analyzed the similarity of aptamer candidates obtained by pFSBC using the sequence diversity. All the non-redundant sequences were quantized based on ORS estimated by pFSBC and were mapped into the two-dimensional space by uniform manifold approximation and projection (UMAP) [22] to make a reference sequence diversity of the oligonucleotide pool. The sequences evaluated in the experimental analysis were also mapped into the two-dimensional space with the same procedure and were compared with the reference sequence diversity. We confirmed that the distance between sequences in this two-dimensional space could be a reasonable reference for identifying binding/non-binding sequences. Our implementation of pFSBC is publicly available at http://www.aoki.ecei.tohoku.ac.jp/fsbc/.

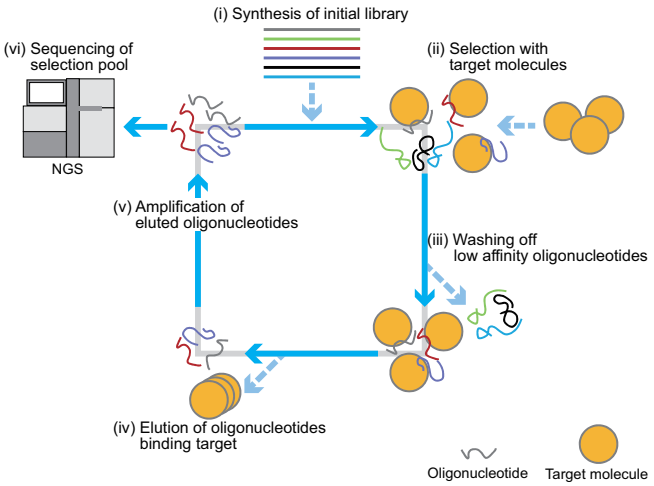

Figure 1. Procedure for high-throughput systematic evolution of ligands by exponential enrichment (HT-SELEX). The SELEX procedure starts with the initial library with random oligonucleotides. The oligonucleotides binding to the target molecules are selected, and non-binding oligonucleotides are washed off and excluded from the pool. The selected oligonucleotides are eluted and amplified by polymerase chain reaction (PCR) for the next round of SELEX. After performing the above procedure for a sufficient number of rounds, around 10 rounds in general, next-generation sequencing (NGS) is applied to the oligonucleotide pool to obtain sequence data.

2. Materials and Methods

2.1. HT-SELEX Sequence Data

The NCBI SRA data of SRR3279661 from BioProject PRJNA315881 [20, 21] was used for evaluating the

[1] http://www.aoki.ecei.tohoku.ac.jp/fsbc/

clustering accuracy and processing time. We used only the fifth-round, which is the last round of SELEX, of sequence data from the above NCBI SRA data. The numbers of all the sequences and non-redundant sequences were 3,946,124 and 584,167 obtained from the fifth-round of sequence data, respectively. Note that the number of sequences is not corresponding to that reported in [23], although we extracted the sequences with the primer information from the NCBI SRA data. Therefore, we also evaluated the performance of conventional methods using the fifth-round data on our environment for a fair comparison.

2.2. FSBC Algorithm

FSBC consists of two parts; ORS estimation and clustering with estimated ORS. FSBC estimates the different lengths of ORS since the length of binding regions of aptamers depends on the target molecules. The ORS estimation takes most of the processing time in the FSBC algorithm. Hence, FSBC employs search space reduction to estimate longer ORS such as more than 10-mer strings. FSBC uses only the single-round sequence data, which is usually the last round. This also helps for reducing the processing time comparing to that with multi-round sequence data. The last round of sequence data is biased due to the accumulation of selection and PCR effect, and the probabilities of nucleobases could be different from each other. Thus, we introduced a new string score considering the nucleobase probabilities of the sequence data. We describe (i) the definition of string score, which is used as a criterion of ORS estimation in our method, (ii) the procedure of ORS estimation, and (iii) the procedure of sequence clustering with ORS as follows.

Definition of String Score. Let assume that the probability of letter c is given by $p(c)$ $(c \in \Omega)$, where Ω is a set of letters. If nucleobases are used as letters, $\Omega = \{\mathrm{A}, \mathrm{C}, \mathrm{G}, \mathrm{T(U)}\}$, where each single-letter indicates adenine (A), guanine (G), cytosine (C), and thymine (T) (or uracil (U)), respectively. The probability of string m is the following equation:

$$Q(m) = \prod_{i=1}^{l} p(m[i]), \tag{1}$$

where $m[i]$ is the i-th letter in string m. Let τ be a set of self-overlapped regions in a string m. For example, if the string m is "ATATA," a set of self-overlapped regions is $\tau = \{\mathrm{A}, \mathrm{ATA}\}$. The probability of self-overlapped region t $(\in \tau)$ in string m is

$$q(t) = \prod_{i=1}^{|t|} p(t[i]), \tag{2}$$

where $|t|$ is the length of t, and $t[i]$ is the i-th letter in t. Let $P_a(m, L)$ be the probability that a sequence s with a length L includes string m with length l. The probability $P_a(m, L)$ is defined by $Q(m)$ and $q(t)$ as follows:

$$P_a(m, L) = \begin{cases} 0 & (L < l), \\ P_a(m, L-1) + Q(m)[1 - P_a(m, L-l)] & \\ \quad - \sum_{t \in \mathcal{T}} \frac{Q(m)}{q(t)} [P_a(m, L-l+|t|) & \\ \quad - P_a(m, L-l+|t|-1)] & (l \le L). \end{cases} \tag{3}$$

The balance of nucleobases is not equal for each other due to the accumulation of the selection and PCR effects. Eq. (3) is derived from $Q(m)$ and $q(t)$, which are defined with $p(c) \in \Omega$ for considering inbalance of nucleobases. $P_a(m, L-1)$ and $Q(m)$ indicate the probability of the L-length sequence with the string m at the position from 1 to $L-l-1$ and at $L-l$, respectively. $Q(m)P_a(m, L-l)$ indicates the probability of the sequence including the string m twice. $-\sum_{t \in \mathcal{T}} \frac{Q(m)}{q(t)} [P_a(m, L-l+|t|) - P_a(m, L-l+|t|-1)]$ indicates the probability of the sequence including the string m at self-overlapped region twice.

The lengths of sequences obtained by NGS are different due to the insertion and deletion during the SELEX process. Therefore, we need to take care of the different lengths of sequence in calculating the probability $P_a(m, L)$. The modified probability, $P_d(m, S)$, is defined by

$$P_d(m, S) = \frac{1}{|S|} \sum_{i}^{|S|} P_a(m, |s_i|), \tag{4}$$

where S is a set of all the sequence data obtained by NGS, $|S|$ is the number of sequences in the data S, and $|s_i|$ is the length of the i-th sequence. Let F_m be the frequency of sequences including string m in data S. Then, the Z-score for the string m is defined by

$$Z(m, S) = \frac{F_m - |S| P_d(m, S)}{\sqrt{|S| P_d(m, S) [1 - P_d(m, S)]}}, \tag{5}$$

since $P_d(m, S)$ is followed by the Bernoulli distribution.

ORS Estimation. This subsection describes the procedure for ORS estimation in FSBC. The probability of letter c is estimated by

$$p(c) = \sum_{i=1}^{|S|} \frac{n_i^c}{|s_i|} \tag{6}$$

in advance, where n_i^c is the number of letter c in the i-th sequence. FSBC searches ORS with lengths ranged from l_{min} to l_{max} with search space reduction. The following is the procedure of ORS estimation.

1. $l \leftarrow l_{min}$.

2. Calculate the Z-scores of all the l_{min}-length strings. Select the l_{min}-length string whose Z-score is higher than 0 as ORS.

3. Extend ORS by adding to one letter in Ω and calculate their Z-scores. Select the extended string whose Z-score is higher than that of the string before extension as ORS.

4. If $l + 1 > l_{max}$, then finish ORS estimation.

5. $l \leftarrow l + 1$ and go to step 3.

The detail of the procedure for ORS estimation is shown in the Algorithm 1.

Algorithm 1 Procedure for ORS estimation

Require: $S = \{s_1, s_2, \cdots, s_{|S|}\}$: Sequence data from NGS;
$\Omega = \{\mathrm{A}, \mathrm{C}, \mathrm{G}, \mathrm{T(U)}\}$: A set of letters;
l_{min}, l_{max}: The minimum/maximum length of ORS to search;
Ensure: M: Estimated ORS;
$l \leftarrow l_{min}$
$M_{l_{min}} \leftarrow \{m_1, m_2, \cdots, m_{|\Omega|^{l_{min}}}\}$
{$M_{l_{min}}$ is a set of all the l_{min}-length sequences}
$M_l \leftarrow \{\phi\}$
for $i = 1$ to $|M_{l_{min}}|$ **do**
  if $Z(m_i, S) > 0$ **then**
    Add m_i to M_l
    {M_l is a set of estimated ORS with length l}
  end if
end for
Add all the elements in M_l to M
$M_{l+1} \leftarrow \{\phi\}$
while $l + 1 \leq l_{max}$ **do**
  for $c \in \Omega$ **do**
    for $i = 1$ to $|M_l|$ **do**
      $m_i^c \leftarrow$ Extend the i-th string m_i in M_l with c
      if $Z(m_i^c, S) > Z(m_i, S)$ **then**
        Add m_i^c to M_{l+1}
      end if
    end for
  end for
  Add all the elements in M_{l+1} to M
  $M_l \leftarrow M_{l+1}$
  $M_{l+1} \leftarrow \{\phi\}$
  $l \leftarrow l + 1$
end while

Sequence Clustering. This subsection describes the procedure for sequence clustering in FSBC. Sequence clustering is based on the Z-score of estimated ORS in Eq. (5). The problem is that the Z-score of the ORS depends on the length of the ORS, that is, the Z-score of long ORS is large. Therefore, we need to normalize the Z-score to compare ORS with different lengths. The Z-scores is normalized by empirical distribution and is given by

$$Z^*(m, S) = \frac{Z(m, S) - \hat{\mu}_{|m|}}{\hat{\sigma}_{|m|}}, \tag{7}$$

where $\hat{\mu}_{|m|}$ and $\hat{\sigma}_{|m|}$ are the mean and standard deviation of Z-scores calculated from estimated ORS with length $|m|$, respectively. All the estimated ORS are sorted in descending order by Z^*-score. The following is the procedure of sequence clustering.

1. $i \leftarrow 1$.

2. Select sequences including the i-th ORS from the sequence data, where a set of selected sequences is referred to the i-th cluster. Remove the selected sequences from the sequence data. If there are no sequences in the sequence data, finish sequence clustering.

3. $i \leftarrow i + 1$ and go to step 2.

The detail of sequence clustering is shown in the Algorithm 2.

Algorithm 2 Procedure for sequence clustering

Require: S': Non-redundant sequences extracted from S, which are sorted by the frequency in descending order;
M': A set of ORS m sorted by $Z^*(m, S)$ in descending order;
Ensure: $C = \{C_1, C_2, \cdots\}$: Clusters of S;
$i \leftarrow 1$
$S'' \leftarrow \{\phi\}$
while $|S'| > 0$ and M' has the i-th element **do**
  for $j = 1$ to $|S'|$ **do**
    if s'_j includes m_i **then**
      {m_i is the i-th element of M'}
      {s'_j is the j-th element of S'}
      Add s'_j to cluster C_i
    else
      Add s'_j to S''
    end if
  end for
  $i \leftarrow i + 1$
  $S' \leftarrow S''$
  $S'' \leftarrow \{\phi\}$
end while

2.3. Parallel Implementation

FSBC consists of ORS estimation and sequence clustering as mentioned above, and ORS estimation takes most of the processing time in FSBC. Hence, we consider reducing the processing time of ORS estimation by introducing parallel implementation.

Figure 2. Outline of ORS estimation with reducing the search space. The strings in blue rectangles and red crossed are selected and excluded, respectively. Z-scores between parent and child nodes are compared, and if Z-score of the child node is smaller than that of the parent node, the sequence on the child node is excluded.

Figure 2 shows the outline of ORS estimation with reducing the search space, when extending strings by adding one letter recursively. As mentioned in Sect. 2.2, ORS are estimated by comparing the Z-scores between the l-length strings m_i ($\in M_l$) and the $(l+1)$-length string m_i^c, which is the concatenation of m_i ($\in M_l$) and the letter c ($\in \Omega$). If all the combinations of strings with length from l_{min} to l_{max} are considered in ORS estimation, the total number of strings is $\sum_{l=l_{min}}^{l_{max}} 4^l$. This is an extremely time-consuming task. Addressing this problem, the total number of strings is reduced by comparing their Z-scores. If the Z-score of the $(l+1)$-length string m_i^c is higher than that of l-length string m_i, its m_i^c is selected. On the other hand, if Z-score of m_i^c is less than that of m_i, its m_i^c are excluded. By using the above process, the search space can be reduced, and the number of selected strings can be significantly reduced. Calculation and comparison of Z-score are independent processes on each m_i on the parent node and its extended strings m_i^c on the child nodes in Fig. 2. For example, the string m_i, e.g., "AAA," and the extended strings m_i^c, e.g., "AAAA," "AAAC," "AAAG," and "AAAT," are independent each other in Z-score calculation. Therefore, the loop for $c \in \Omega$ and the loop from $i = 1$ to $|M_l|$ in Algorithm 1 are clearly independent of each other. For example, the final results are not changed, if the loop $i = 1, \cdots, |M_l|$ is replaced with $i = |M_l|, \cdots, 1$. Hence, we can implement this process in parallel to speed up ORS estimation in FSBC, which is called pFSBC.

2.4. Performance Evaluation

The NCBI SRA data of SRR3279661 from BioProject PRJNA315881 [20, 21], which includes the information of the dissociation constant of aptamers against target molecules, were used for evaluating the accuracy and processing time of sequence clustering methods. Allnutt et al. [23] used this data for evaluating the accuracy by ranking aptamers with the desired binding affinity at the top of the list. They used 3 criteria, r_s, r, and "Top 10 correct," to evaluate the accuracy of sequence clustering. r_s is defined by Spearman's rank correlation between the cluster rank and the K_d rank. r is defined by Pearson's correlation between the cluster rank and K_d, where K_d is a dissociation constant and K_d rank is a rank of aptamers with known K_d sorted in ascending order. "Top 10 correct" is the number of "good" binders, that is, $K_d < 100$ nM, observed in the top-10 ranked clusters, while "Top 10 correct" takes into account only the sensitivity of correct clusters. In addition to the above criteria, we used "Top 10 incorrect" and positive predictive value (PPV). "Top 10 incorrect" is the number of aptamers ($K_d \geq 100$ nM) observed in the top-10 ranked clusters, and PPV is defined by Top 10 correct/(Top 10 correct + Top 10 incorrect). We used the above 5 criteria, i.e., r_s, r, Top 10 correct, Top 10 incorrect and PPV, for evaluating the accuracy of clustering methods.

We compared the five clustering methods: FASTAptamer[2], the Usearch programs (Uclust and Unoise)[3], AptaCluster[4], and pFSBC. AptaTRACE was not included in this experiment since AptaTRACE requires multiple rounds of SELEX data. We used the default parameters for AptaCluster and Unoise. For FASTAptamer, the options '-d 7', '-c 500', and '-f 10' were used to specify the edit distance according to the user guide, restrict the maximum number of clusters to 500, and filter sequences with fewer than 100 identical copies, respectively. We also used the other options '-d 7' and '-f 100' for FASTAptamer for changing the frequency filtering. For Uclust, we used identity thresholds of 97% and 90%. The options of pFSBC were $l_{min} = 4, 5, 6$ and $l_{max} = 10$, and $l_{min} = 5$ and $l_{max} = 25$ for long-ORS estimation. The clusters obtained by Uclust, Unoise, and AptaCluster were ranked by their cluster size, and FASTAptamer ranked clusters by the sequence frequency. The cluster rank of pFSBC corresponds to the order of clusters since FSBC makes clusters in order of decreasing the Z^*-score of ORS.

All the methods were evaluated on the computer with CentOS 16.10 64bit, Intel®Xeon®CPU E5-2680@2.7GHz, and 132GB memory. The processing time was also evaluated in the same computational environment. We used Python with multi-threading for pFSBC and ran it on 32 CPU cores.

[2] https://burkelab.missouri.edu/fastaptamer.html
[3] https://drive5.com/usearch/
[4] https://github.com/drivenbyentropy/aptasuite

2.5. Sequence Diversity

We analyzed the similarity among aptamer candidates obtained by pFSBC comparing with the sequence diversity of the oligonucleotide pool. We introduced dimensional compression of the features by vector quantization in this analysis. Non-redundant sequences were encoded with ORS as a codebook to abstract the sequence representation. Let $\tilde{S}$ be a set of non-redundant sequences, and let $b_i \in R^{|M|}$ be the quantized vector of the i-the sequence $s_i \in \tilde{S}$. If the i-th sequence s_i includes j-th ORS, then $b_{ij} = 1$, otherwise $b_{ij} = 0$. By applying vector quantization to all the non-redundant sequences $\tilde{S}$ in the same way as above, we can obtain abstracted sequence representation, i.e., quantized vectors $b_i \in B$, where B is a matrix $R^{|\tilde{S}|\times|M|}$, which consists of $b_i, i = 1, 2, \cdots, |\tilde{S}|$. In order to visualize matrix B, we used UMAP [22], which is one of the dimension reduction techniques. In this experiment, the number of components was set to 2, therefore the matrix B reduced its dimension as $B \rightarrow U \in \mathcal{R}^{|\tilde{S}|\times 2}$. Other options of UMAP were set as the default. The matrix U represents the diversity of all the quantized vectors in two dimensions. We used all the non-redundant sequences with all the ORS estimated by pFSBC ($l_{min} = 5, l_{max} = 10$) in this analysis. The sequences in Table 1 were analyzed by their position in the distribution of all the non-redundant sequences.

3. Results and Discussion

Table 1 shows the summary of experimental results for FASTAptamer (f 100 and f 10), Uclust (90% and 97%), Unoise, AptaCluster and pFSBC (($l_{min} = 4, 5, 6$ and $l_{max} = 10$), and ($l_{min} = 5$ and $l_{max} = 25$)). Each Column indicates sequence ID, dissociation constant (K_d nM), sequence rank, and cluster rank for each method with options, respectively. The bottom 6 rows in the table indicate Spearman's rank correlation (r_s), Pearson's correlation (r), Top 10 correct, Top 10 incorrect, PPV, and processing time for each method and option, respectively. 33 sequences from 584,167 non-redundant sequences were evaluated for binding affinity to the target molecules. The sequences are ordered by K_d in this table. The 1st-ranked sequence L462 exhibited the strongest affinity to the target molecules, while sequences L409, H26, L417, H5, H15, and H24 did not show enough affinity to the target molecules. The best results for Spearman's rank correlation, Pearson's correlation, Top 10 correct, Top 10 incorrect, and PPV are pFSBC ($l_{min} = 6, l_{max} = 10$), Uclust (97%), pFSBC ($l_{min} = 4, l_{max} = 10$), pFSBC (($l_{min} = 5, l_{max} = 10$), ($l_{min} = 6, l_{max} = 10$), and ($l_{min} = 5, l_{max} = 25$)), and pFSBC (($l_{min} = 5, l_{max} = 10$) and ($l_{min} = 5, l_{max} = 25$)), respectively. Therefore, pFSBC exhibited the highest performance in all the criteria except for Pearson's correlation. PPVs for FASTAptamer (f 100 and f 10), Uclust (90% and 97%), Unoise, AptaCluster, pFSBC ($l_{min} = 4, l_{max} = 10$), ($l_{min} = 5, l_{max} = 10$), ($l_{min} = 6, l_{max} = 10$), and ($l_{min} = 5, l_{max} = 25$) were 0.7, 0.6, 0.67, 0.71, 0.6, 0.6, 0.67, 0.83, 0.67, 0.83, 0.67, and 0.83, respectively. Hence, PPVs of pFSBC with ($l_{min} = 5, l_{max} = 10$) and ($l_{min} = 5, l_{max} = 25$) were better than other conventional methods and those of pFSBC with options ($l_{min} = 4, l_{max} = 10$) and ($l_{min} = 6, l_{max} = 10$) were comparable with other methods. Only pFSBC with the options ($l_{min} = 4, l_{max} = 10$) and ($l_{min} = 5, l_{max} = 10$) selected the best aptamer L462 in the first cluster, and pFSBC with options ($l_{min} = 6, l_{max} = 10$) and ($l_{min} = 5, l_{max} = 25$) selected the best aptamer L462 in the top-10 clusters. On the other hand, FASTAptamer (f 100, f 10), Uclust (90%, 97%), Unoise, and AptaCluster do not select the sequence L462 in the top-10 clusters. The above results were from only the fifth-round data from the NCBI SRA data of SRR3279661 from BioProject PRJNA315881 [20, 21]. Unoise presented the best result with multi-round data, i.e., PPV = 0.875, as reported by [23]. PPVs of pFSBC with $l_{min} = 5, l_{max} = 10$ and $l_{min} = 5, l_{max} = 25$ were 0.83, which is comparable with Unoise with multi-round data, even though pFSBC used only single-round data from the fifth round.

The processing time of pFSBC with $l_{min} = 4$ and $l_{max} = 10$ was 38 sec, which is the fastest in all the methods. The processing time of FASTAptamer (f 100) was shorter than pFSBC ($l_{min} = 6, l_{max} = 10$) and ($l_{min} = 5, l_{max} = 25$) since FASTAptaer with 'f 100' option did not use all the sequences with a frequency filtering option. The option 'f 100' eliminates sequences whose frequency is less than 100 to dramatically speed up clustering, where FASTAptamer (f 100) is about 28.8 times faster than FASTAptamer (f 10). In other words, FASTAptamer (f 100) can find the only aptamers whose frequency is more than or equal to 100, although other methods can find all the aptamers even if their frequency is less than 100. PPV of FASTAptamer (f 100) in this experiment is good compared with other methods since the data used in this experiment include the only aptamer whose frequency is more than or equal to 100. Therefore, there is a strong trade-off between the accuracy and the processing time in the option of FASTAptamer. As a result, pFSBC with all the options exhibited the fastest processing time compared with other methods as shown in Table 1. pFSBC ($l_{min} = 5, l_{max} = 25$) can also estimate ORS with lengths from 5 to 25, where the processing time is only 152 sec. Hence, pFSBC could be useful to find long ORS in reasonable processing time.

Figure 3 shows the relation between the cluster rank and the sequence frequency for FASTAptamer (f 100), Uclust (97%), Unoise, AptaCluster, and pFSBC ($l_{min} = 5, l_{max} = 10$) and ($l_{min} = 5, l_{max} = 25$). The horizontal and vertical axes indicate the cluster rank and sequence frequency, respectively. The dark blue and light blue

Table 1. Experimental results of sequence clustering for each method. Each column indicates sequence ID, dissociation constant (K_d nM), sequence rank, and cluster rank for each method with options, respectively. pFSBC performed with different options, which are ($l_{min} = 4, l_{max} = 10$), ($l_{min} = 5, l_{max} = 10$), ($l_{min} = 6, l_{max} = 10$), and ($l_{min} = 5, l_{max} = 25$). The bottom 6 rows indicate Spearman's rank correlation (r_s), Pearson's correlation (r), Top 10 correct, Top 10 incorrect, positive predictive value (PPV), and processing time, respectively.

Sequence ID	Kd	rank	FASTAptamer f 100	FASTAptamer f 10	Uclust 90%	Uclust 97%	Unoise	AptaCluster	pFSBC $l_{min} = 4$ $l_{max} = 10$	pFSBC $l_{min} = 5$ $l_{max} = 10$	pFSBC $l_{min} = 6$ $l_{max} = 10$	pFSBC $l_{min} = 5$ $l_{max} = 25$
L462	2	1	12	12	12	16	12	13	1	1	3	3
L464	4	2	22	22	26	30	21	25	4	12	40	14
L455	4	3	20	20	24	29	20	23	20	29	137	31
L454	8	4	10	11	11	25	11	11	14	75	430	77
H33	10	5	39	39	51	70	38	48	37	75	84	77
L463	12	6	25	25	30	35	25	28	27	54	179	56
H4	18	7	5	5	5	8	5	5	15	144	156	146
H12	20	8	14	14	14	18	14	15	6	7	12	9
H22	20	8	26	27	31	42	26	29	6	7	12	9
H30	25	9	37	37	47	59	35	42	32	78	412	80
H0	25	9	1	1	1	1	1	1	1	1	1	1
L465	25	9	27	28	32	39	27	30	16	81	179	83
L418	35	10	13	13	13	17	13	14	9	148	486	150
L413	40	11	17	17	20	23	17	20	21	30	40	32
H6	50	12	7	7	7	10	7	7	8	100	140	102
H3	60	13	4	4	4	7	4	4	2	2	183	4
H2	65	14	3	3	3	4	3	3	7	11	92	13
H8	80	15	8	9	9	11	9	9	4	11	107	13
L420	80	15	21	21	33	34	24	24	13	78	93	80
H40	120	16	44	44	59	93	44	56	7	11	107	13
H1	120	16	2	2	15	2	2	2	1	1	3	3
L412	120	16	32	33	40	51	31	37	14	33	57	35
H14	123	17	16	16	19	21	16	19	15	81	121	83
H16	375	18	19	19	23	27	19	22	15	48	179	50
H7	375	18	9	8	8	12	8	8	4	11	17	13
H9	375	18	11	10	10	14	10	10	11	43	179	45
H20	375	18	24	23	27	31	22	26	35	81	179	83
L409	500	19	33	32	38	120	37	36	39	81	179	83
H26	500	19	36	36	43	57	34	40	28	81	179	83
L417	500	19	23	24	28	33	23	27	10	160	453	166
H5	500	19	6	6	6	9	6	6	2	74	179	76
H15	500	19	18	18	22	48	18	21	17	20	27	22
H24	500	19	29	29	34	44	28	32	11	52	171	54
r_s			0.11	0.08	0.11	0.14	0.12	0.08	0.09	0.22	0.23	0.22
r			0.12	0.14	0.12	0.26	0.19	0.13	0.18	0.20	0.13	0.12
Top 10 correct			7	6	6	5	6	6	10	5	2	5
Top 10 incorrect			3	4	3	2	4	4	5	1	1	1
PPV			0.70	0.60	0.67	0.71	0.60	0.60	0.67	0.83	0.67	0.83
Processing time (s)			131	3,772	1,674	2,230	539	258	38	63	175	152

dots represent sequences with $K_d < 100$ nM and $K_d \geq 100$ nM, respectively. The numbers on dots indicate their rank based on the dissociation constant, e.g. the number "1" indicates the 1st-ranked sequence in Table 1, i.e., the sequence ID L462. The dotted vertical line in each plot indicates the cluster rank at 10. FASTAptamer (f 100), Uclust (97 %), Unoise, and AptaCluster did not include the 1st-ranked aptamer, whose K_d is 2 nM, in the top-10 ranked clusters, and the strong correlation between the cluster rank and the sequence frequency was observed from Figure 3. These conventional methods are not useful for selecting low-frequency aptamers as high-ranking clusters since high-frequency non-aptamers are selected as high-ranking clusters, On the other hand, pFSBC select the 1st-ranked aptamer and only 1 non-aptamer in the top-10 ranked cluster for both parameters ($l_{min} = 5, l_{min} = 10$) and ($l_{min} = 5, l_{min} = 25$). Thus, pFSBC could be useful to select low-frequency aptamers and to avoid non-aptamers such as high-frequency noise sequences.

Table 2 shows the processing time for AptaCluster, the original FSBC implemented in R [12], and pFSBC with 1 and 32 cores implemented in Python. The processing time of the original FSBC was longer than that of AptaCluster, while the processing time of pFSBC with 32 cores for all the options was shorter than that of AptaCluster. Comparing R and Python implementation of FSBC, the processing time of Python is about 7 times faster than that of R. pFSBC with options ($l_{min} = 4, l_{max} = 10$), ($l_{min} = 5, l_{max} = 10$), and ($l_{min} = 6, l_{max} = 10$) using 32 cores were about 47, 53 and 55 times faster than the original FSBC (R) and about 6.5, 7.6, and

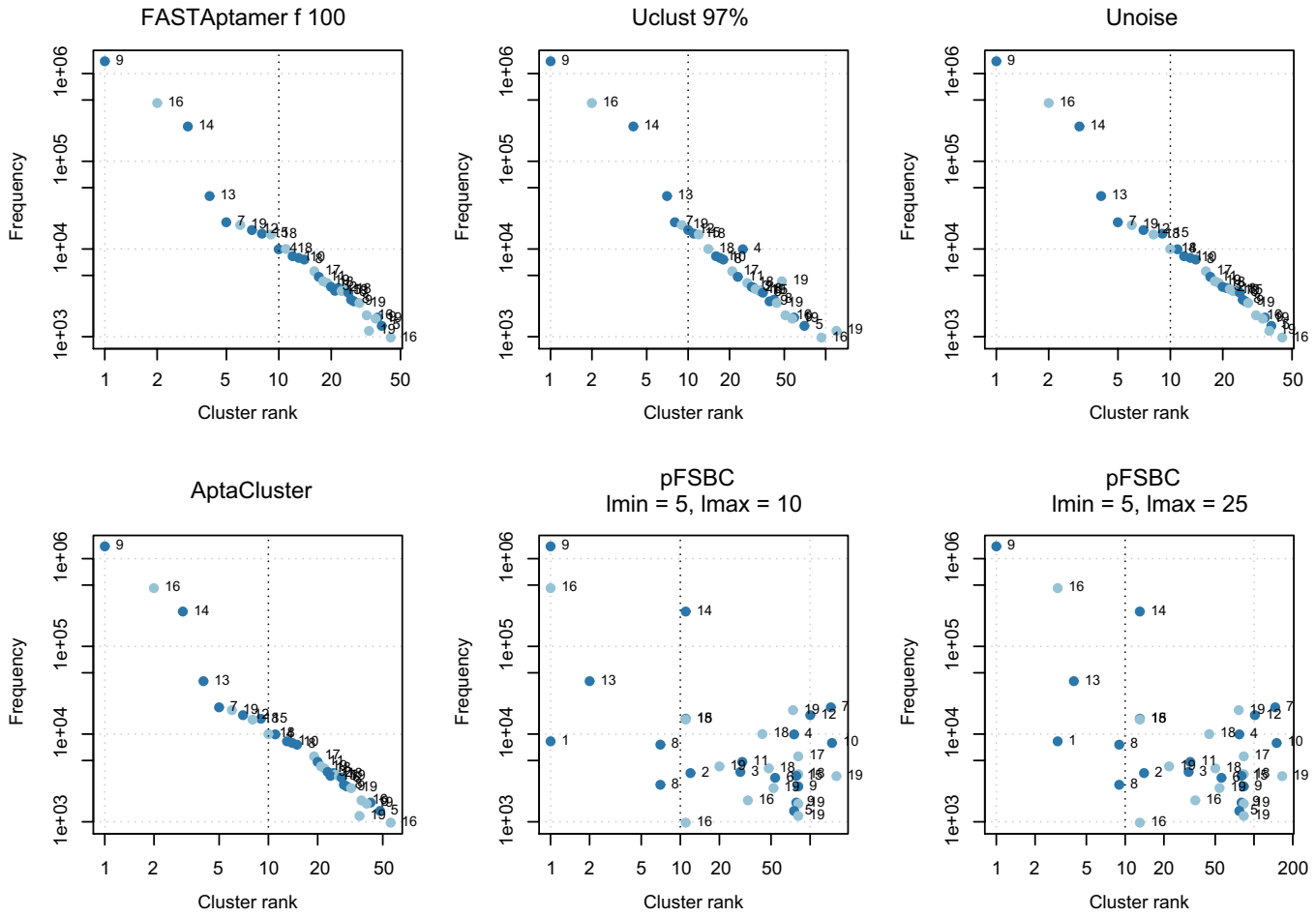

Figure 3. The relation between cluster rank and frequency for each clustering method: FASTAptamer (f 100), Uclusst (97%), Unoise, AptaCluster, pFSBC ($l_{min} = 5, l_{max} = 5$), and ($l_{min} = 5, l_{max} = 25$). The horizontal and vertical axes indicate the cluster rank and the frequency of sequences in Table 1, respectively. The dark blue and light blue dots represent sequences with $K_d < 100$ nM and $K_d \geq 100$ nM, respectively. The numbers on dots indicate their rank based on the dissociation constant. The dotted vertical line in each plot indicates the cluster rank at 10.

7.2 times faster than pFSBC with 1 core, respectively. This result indicated that the parallel processing of FSBC is effective to reduce the processing time. We perform pFSBC with the option ($l_{min} = 5, l_{max} = 25$) to find longer ORS. The total number of strings with lengths from 5 to 25 is $\sum_{l=5}^{25} 4^l = 1.5012 \times 10^{15}$. This is an extremely huge number, and it is difficult to compute all the strings in reasonable processing time. pFSBC with search space reduction can improve the processing time to 1 min 32 sec. The number of estimated ORS was 1,734 and the ratio to the total number was 1.1551×10^{-12}. As observed above, pFSBC is also effective in detecting long ORS in a reasonable time.

Figure 4 shows the sequence diversity of quantized sequence vectors based on the estimated ORS in the two-dimensional space by UMAP. The large and small dots represent evaluated sequences in Table 1 and non-evaluated sequences, respectively. Colors for small dots indicate the cluster ranks estimated by pFSBC with options ($l_{min} = 5, l_{max} = 10$). The large dots with dark and light blue are the dissociation constant $Kd < 100$ and $Kd \geq 100$, respectively. The words on the large dots indicate the sequence ID in Table 1. The sequences evaluated in the experiments were broadly distributed in the reference sequence diversity of oligonucleotide pool. Hence, the different types of sequences could be evaluated for binding affinity through experimental analysis. The 1st-ranked sequence L462 is close to H6, L454, L465, while L462 is far from H2, H4, H8, and L455. Thus, H6 could show a similar binding style to target molecules with L462, while H2 could have a different binding style to the target molecules from L462. The sequences L464, L412, H0, and H15 are close to each other, and these sequences are mixed with binding and non-binding sequences. On the other hand, binding sequences L462, H6, L465, and L454 are located close to each other. Similarly, binding sequences H2, H8, H14, H22, and L455 are close to each other,

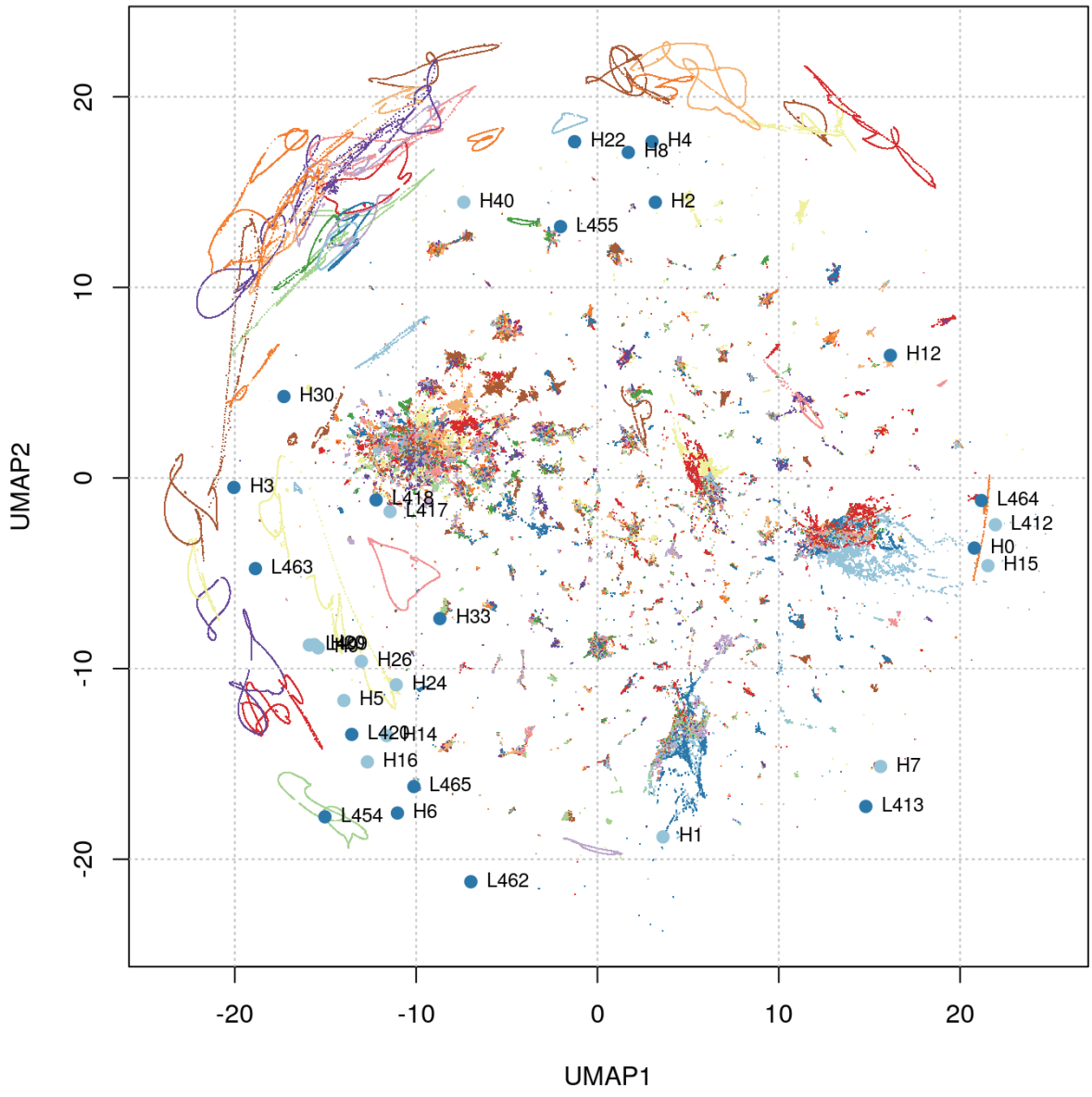

Figure 4. Sequence diversity of quantized vectors of estimated ORS in the two-dimensional space visualized by UMAP. The large and small dots represent evaluated sequences in Table 1 and non-evaluated sequences, respectively. Colors for small dots indicate the cluster ranks estimated by pFSBC with options ($l_{min} = 5, l_{max} = 25$). The large dots with dark and light blue are the dissociation constant $Kd < 100$ and $Kd \geq 100$, respectively. The words on the large dots indicate the sequence ID in Table 1.

and H3 H30 and L463 are also close to each other. Hence, the distance between sequences in this two-dimensional space could be a reasonable reference for identifying binding/non-binding sequences. ORS estimated by pFSBC are available to use for sequence clustering and are also useful to evaluate the sequence diversity of oligonucleotide pools. The comparison of the distributions between evaluated sequences and all the non-redundant sequences facilitates to understand the bias of evaluated sequences from oligonucleotide pools.

Table 2. Processing time for AptaCluster, the original FSBC implemented in R, and pFSBC with 1 and 32 cores implemented in Python.

Method	Option	Processing time
AptaCluster	Default options	4 min 18 sec
Original FSBC (R)	$l_{min} = 4$ $l_{max} = 10$	30 min 9 sec
	$l_{min} = 5$ $l_{max} = 10$	54 min 27sec
	$l_{min} = 6$ $l_{max} = 10$	159 min 2sec
pFSBC with 1 core (Python)	$l_{min} = 4$ $l_{max} = 10$	4 min 8 sec
	$l_{min} = 5$ $l_{max} = 10$	8 min 1 sec
	$l_{min} = 6$ $l_{max} = 10$	20 min 52 sec
pFSBC with 32 cores (Python)	$l_{min} = 4$ $l_{max} = 10$	38 sec
	$l_{min} = 5$ $l_{max} = 10$	1 min 3 sec
	$l_{min} = 6$ $l_{max} = 10$	2 min 55 sec
	$l_{min} = 5$ $l_{max} = 25$	1 min 32 sec

4. Conclusion

In this paper, we proposed the parallel implementation of FSBC (pFSBC) using Python with multi-threading to improve the processing time of the original FSBC implemented in R. Experimental evaluation with the NCBI SRA data of SRR3279661 from BioProject PRJNA315881 [20, 21] demonstrated that pFSBC exhibited the most accurate in sequence clustering and the fastest processing time in the conventional clustering methods: FASTAptamer, Uclust, Unoise, and AptaCluster. pFSBC with $l_{min} = 6, l_{max} = 10$ running on 32 CPU cores reduced 98% processing time from the original FSBC. Hence, the parallel processing of FSBC is effective to reduce the processing time. We expect that pFSBC will help to avoid the time-consuming clustering task, and it will provide accurate clustering results for the HT-SELEX data.

Acknowledgement. This research was partially supported by the Japan Society for the Promotion of Science (JSPS), KAKEN (17H00825).

References

[1] Ellington, A.D. and Szostak, J.W. (1990) In vitro selection of RNA molecules that bind specific ligands. *Nature* **346**(6287): 818–822.

[2] Bunka, D.H., Platonova, O. and Stockley, P.G. (2010) Development of aptamer therapeutics. *Current Opinion in Pharmacology* **10**(5): 557–562.

[3] Ruiz Ciancio, D., Vargas, M., Thiel, W., Bruno, M., Giangrande, P. and Mestre, M. (2018) Aptamers as diagnostic tools in cancer. *Pharmaceuticals* **11**(3): 86.

[4] Gold, L., Ayers, D., Bertino, J., Bock, C., Bock, A., Brody, E., Carter, J. *et al.* (2010) Aptamer-based multiplexed proteomic technology for biomarker discovery. *Nature Precedings* : 1.

[5] Kaneko, N., Horii, K., Akitomi, J., Kato, S., Shiratori, I. and Waga, I. (2018) An aptamer-based biosensor for direct, label-free detection of melamine in raw milk. *Sensors* **18**(10): 3227.

[6] Minagawa, H., Shimizu, A., Kataoka, Y., Kuwahara, M., Kato, S., Horii, K., Shiratori, I. *et al.* (2020) Fluorescence polarization-based rapid detection system for salivary biomarkers using modified DNA aptamers containing base-appended bases. *Analytical Chemistry* **92**(2): 1780–1787.

[7] Zimmermann, G.R., Wick, C.L., Shields, T.P., Jenison, R.D. and Pardi, A. (2000) Molecular interactions and metal binding in the theophylline-binding core of an RNA aptamer. *RNA* **6**(5): 659–667.

[8] Qu, H., Csordas, A.T., Wang, J., Oh, S.S., Eisenstein, M.S. and Soh, H.T. (2016) Rapid and label-free strategy to isolate aptamers for metal ions. *ACS Nano* **10**(8): 7558–7565.

[9] Farzin, L., Shamsipur, M. and Sheibani, S. (2017) A review: Aptamer-based analytical strategies using the nanomaterials for environmental and human monitoring of toxic heavy metals. *Talanta* **174**: 619–627.

[10] Marton, S., Cleto, F., Krieger, M.A. and Cardoso, J. (2016) Isolation of an aptamer that binds specifically to e. coli. *PLoS ONE* **11**(4): e0153637.

[11] Tuerk, C. and Gold, L. (1990) Systematic evolution of ligands by exponential enrichment: RNA ligands to bacteriophage T4 DNA polymerase. *Science* **249**(4968): 505–510.

[12] Kato, S., Ono, T., Minagawa, H., Horii, K., Shiratori, I., Waga, I., Ito, K. *et al.* (2020) FSBC: Fast string-based clustering for HT-SELEX data. *BMC Bioinformatics* **21**(263): 1–9.

[13] R Core Team (2013) *R: A Language and Environment for Statistical Computing*, R Foundation for Statistical Computing, Vienna, Austria.

[14] Ng, E.W.M., Shima, D.T., Calias, P., Cunningham, E.T., Guyer, D.R. and Adamis, A.P. (2006) Pegaptanib, a targeted anti-VEGF aptamer for ocular vascular disease. *Nature Reviews Drug Discovery* **5**(2): 123–132.

[15] Dao, P., Hoinka, J., Takahashi, M., Zhou, J., Ho, M., Wang, Y., Costa, F. *et al.* (2016) AptaTRACE elucidates RNA sequence-structure motifs from selection trends in HT-SELEX experiments. *Cell systems* **3**(1): 62–70.

[16] Hoinka, J., Berezhnoy, A., Sauna, Z.E., Gilboa, E. and Przytycka, T.M. (2014) Aptacluster - A method to cluster HT-SELEX aptamer pools and lessons from its application. *Research in Computational Molecular Biology* **8394**: 115–128.

[17] Caroli, J., Taccioli, C., De La Fuente, A., Serafini, P. and Bicciato, S. (2016) APTANI: A computational tool to select aptamers through sequence-structure motif analysis of HT-SELEX data. *Bioinformatics* **32**: 161–164.

[18] Alam, K.K., Chang, J.L. and Burke, D.H. (2015) FASTAptamer: A bioinformatic toolkit for high-throughput sequence analysis of combinatorial selections. *Molecular Therapy. Nucleic Acids* **4**(3): e230.

[19] Jiang, P., Meyer, S., Hou, Z., Propson, N.E., Soh, H.T., Thomson, J.A. and Stewart, R. (2014) MPBind: A meta-motif-based statistical framework and pipeline to predict binding potential of SELEX-derived aptamers. *Bioinformatics* **30**(18): 2665–2667.

[20] Hoinka, J., Berezhnoy, A., Dao, P., Sauna, Z.E., Gilboa, E. and Przytycka, T.M. (2015) Large scale analysis of the mutational landscape in HT-SELEX improves aptamer discovery. *Nucleic Acids Research* **43**(12): 5699–5707.

[21] Levay, A., Brenneman, R., Hoinka, J., Sant, D., Cardone, M., Trinchieri, G., Przytycka, T.M. *et al.* (2015) Identifying high-affinity aptamer ligands with defined cross-reactivity using high-throughput guided systematic evolution of ligands by exponential enrichment. *Nucleic Acids Research* **43**(12): e82–e82.

[22] McInnes, L., Healy, J. and Melville, J. (2018) UMAP: Uniform manifold approximation and projection for dimension reduction. *ArXiv* **abs/1802.03426**.

[23] Allnutt, T.R., Quinn, T.P., Richardson, M.F. and Crowley, T.M. (2018) Shortlisting aptamer candidates from HT-SELEX data. *Aptamers* **2**: 36–44.