Detection of topologically associating domains with modified TopDom method

Julia Rymuza, Ewa Kizling, Anna Semik, Konrad Łukaszyk

January 2022

1 Introduction

DNA particles are stored in cell nuclei together with packaging proteins in the form of chromosomes. The organisation of those structures regulates expression of genes they contain. Euchromatin, which is a lightly packed form of chromatin, can be transcriptionally active, whereas its closed counterpart, heterochromatine stays silenced. Chromosomes consist of multi-Mb compartments containing either open or closed chromatin, called A- and B-compartments respectively. Position of those compartments varies among cell types due to their different gene expression patterns[DH15]. Topologically associating domains (TADs) are an additional level of compartmentalization. They are segments of chromosomes smaller than A/B-compartments, defined by areas of enriched chromatin interactions. TADs are believed to regulate transcription by enabling enhancers and other regulatory elements direct contact with their target genes. TADs have been identified in a variety of species, tissue types, and differentiation stages. The proper identification of these domains using high-throughput chromatin conformation capture (Hi-C) assays is crucial to the reliability and reproducibility of these findings[Zuf+18].

CTCF, cohesin and loops TAD boundaries in mammals are enriched in binding sites for architectural proteins such as CTCF and cohesin. Nevertheless these sites also exist within TADs and depletion of aforementioned architectural proteins reduce number of contacts inside TADs without changing their overall localization, as shown in Zuin et al., 2014 [Zui+14]. It proves their role in managing contacts between enhancers and promoters inside TADs, yet does not explain their impact on TADs distribution.

In our project, we modified a well-known TAD detection tool TopDom (described in Shin et al., 2016)[Shi+16], in order to improve it.

2 Materials and Methods

2.1 Datasets

In our work, we used Hi-C data obtained from GM12878, HMEC, HUVEC, IMR90 and NHEK cell types [Rao+14]. We compared the results with reference sets of gold-standard definitions of the TADs available in the MEET-EU repository.

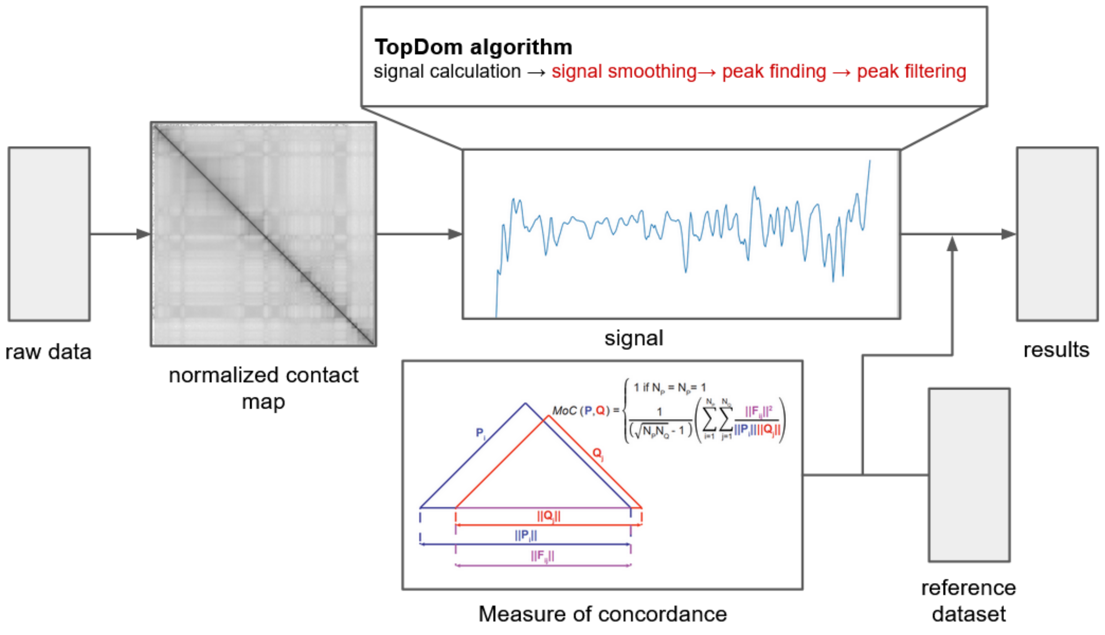

TopDom algorithm
signal calculation → signal smoothing→ peak finding → peak filtering
raw data
normalized contact map
signal
results
$MoC(P,Q) = \begin{cases} 1 \text{ if } N_P = N_P = 1 \\ \frac{1}{(\sqrt{N_P N_Q} - 1)}\left(\sum_{i=1}^{N_P}\sum_{j=1}^{N_Q}\frac{\|F_{ij}\|^2}{\|P_i\|\|Q_j\|}\right)\end{cases}$
P_i
Q_j
$\|Q_j\|$
$\|P_i\|$
$\|F_{ij}\|$
Measure of concordance
reference dataset

Figure 1: Proposed TADs detection pipeline

2.2 TAD detection algorithm

In our approach to detecting TADs we relied strongly on the algorithm proposed in TopDom [Shi+16]. This method has been proven to detect TADs very efficiently [Zuf+18][DB17]. TopDom demonstrated robust and consistent TAD partitions, independent of the normalization and bin size. What is more TADs generated with TopDom are highly reproducible and in concordance with results obtained by other TAD callers. One of TopDom's disadvantages is it's disability to identify TAD hierarchies or overlapping TADs [Zuf+18]

The TopDom algorithm consists of three steps: generating binSignal, detecting TAD boundaries and statistical filtering.

2.2.1 Generating binSignal

In the TopDom algorithm the main idea is that the contact frequencies between TADs is lower then inside one TAD. For each bin with index i we can define upstream region $U_i = \{i - w - 1, .., i - 1\}$ and downstream $D_i = \{i + 1, .., i + w\}$ around it, where w is a free parameter defining the window around a bin. This way we can calculate average contacts around the bin as:

$$binsignal(i) = \frac{1}{w^2}\sum_{l=1}^{w}\sum_{m=1}^{w} cont.freq(U_i(l), D_i(m))$$

Since this step is essential for the TopDom algorithm we decided not to change it and in our approach we calculate the binSignal the same way.

2.2.2 Detecting TAD boundaries

Original TopDom algorithm expects to find TAD boundaries in local minima of binSignal function. It adapts the linear-time algorithm of Kumar Ray et al. It takes into consideration the possibility that some local minima are noise and smooths the function and based on that detects boundaries. Following that idea this step can be further split into two substeps: smoothing and detecting. This presents the possibility for improvement.

Smoothing binSignal function

In our approach we re-implemented the original function in python. In our scrip it is called `detect_local_extrema`. Moreover we wanted to check the performance of different smoothing function present in python libraries scipy and numpy such a as:

1. savgol_filter in code denoted as `savgol_filter`
   implementation of Savitzky–Golay filter, that can be used to a series of data points to smooth the data, that is, to increase precision without altering the signal tendency. This is accomplished using convolution, which involves fitting successive sub-sets of adjacent data points with a low-degree polynomial using the linear least squares method. When the data points are evenly spaced, an analytical solution to the least-squares equations can be found in the form of a single set of "convolution coefficients" that can be applied to all data sub-sets to produce smoothed signal estimates (or derivatives of the smoothed signal) at the central point of each sub-set. Method was published in 1964.

2. convolve in code denoted as `smooth`
   operation specified for two functions (or the signals they describe) resulting with third function that can be viewed as a modified version of the original functions. Scipy includes two methods of convolution: convolution determined directly from sums and the Fourier Transform.

3. qspline1d_eval together with qspline1d in code denoted as `qspline`
   allows to find the quadratic spline coefficients for a 1-D signal assuming mirror-symmetric boundary conditions and evaluate a quadratic spline at the new set of points.[Vir+20]

Detecting local minima

In this substep we also experiment with different functions able to detect local minima that are implemented in scipy including:

1. argrelextrema in code denoted as `find_min`
   allows to calculate the relative extrema of data [Vir+20].

2. find_peaks in code denoted as `find_peaks`
   finds all local maxima by simple comparison of neighboring values. Also, a subset of these peaks can be selected by specifying conditions for a peak's properties [Vir+20].

3. find_peaks_cwt in code denoted as `find_peaks_2`.
   finds peaks in a 1-D array with wavelet transformation. The general approach is to smooth vector by convolving it with wavelet(width) for each width in widths. Relative maxima which appear at enough length scales, and with sufficiently high SNR, are accepted [Vir+20].

2.2.3 Statistical filtering

TopDom algorithm after detecting TAD boundaries allows statistical filtering of false positives. The idea behind it is that interactions between two different TADs should be less frequent than between different upstream/downstream neighbor bins. This concept can be summarised for bin i as:

$$\begin{aligned} between.interactions(i) &= cont.freq(U_i(l), D_i(m))||l-i|| \leq w \text{ and } |m-i| \leq w \\ within.interactions(i) &= cont.freq(U_i(l), U_i(m))||l-i|| \leq w \text{ and } |m-i| \leq w \\ &\text{or} \\ &= cont.freq(D_i(l), D_i(m))||l-i|| \leq w \text{ and } |m-i| \leq w. \end{aligned}$$

At TAD boundary *between.interactions* should be smaller then *within.interactions*. If there is no significant difference we have reasons to believe that there is no TAD boundary. To statistically compere the metrics authors used Wilcox Rank Sum test and calculated $z-score$ to account for distance between bins. Based on that algorithm filters out minima with $p-value$ higher than 0.05. In our approach on one hand we implemented statistical filtering in python in function `statFilter`. Moreover we added possibility to filter peaks with scipy function peak_prominences in code denoted as `filter_peaks`. Only minima that are stand out from surrounding more than given threshold are kept.

2.3 TAD sets comparison

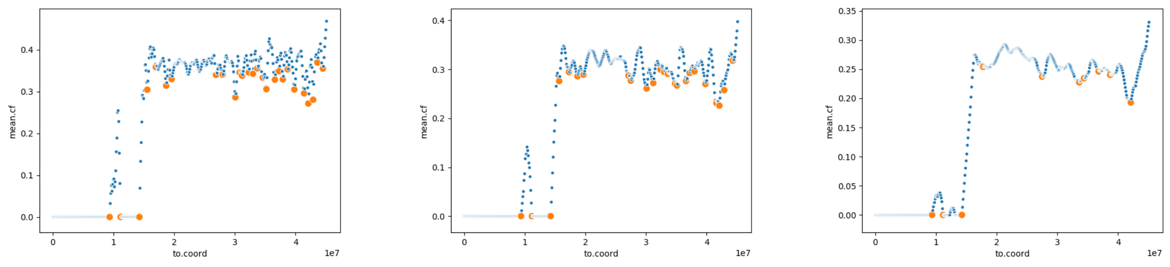
We compared our results with two different tools. Firstly we used the overlapscore function from TopDom [Shi+16], which is an asymmetrical measure, computing the degree of matching between each TAD and overlapping subset of TADs in reference data. Secondly, we implemented the Measure of Concordance (MoC) as in [Zuf+18]. It sums over each pair of TADs in two datasets dividing the squared length of their common part divided by lengths of both TADs. The resulting sum is then divided by square root of sets sizes, reduced by one. It is thus a symmetrical measure.

3 Results

3.1 Generating binSignal

To evaluate the impact of window size we checked TopDom results for their different sizes. We chose for testing $w = 5, 10, 20$ since the authors suggest that it should be between 5 and 20. From figure 2 we can clear see that different window sizes have a big impact on resulting binSignal and detected minimums. For $w = 5$ we can find the highest number of TADs. With the increasing window size the binSignal function has less local extremes. It is also worth noticing that bigger w we mainly detect bigger TADs but not exclusively. Still some small TADs are present even if that do not correspond to strong local minimum in binSignals computed using smaller window size.

Based on presented results and findings presented in [Shi+16] we decided to use window size equal to 5. It is able to detect most TADs and has the highest average PCC/wPCC scores, where PCC stand for Pearson's correlation coefficient [Shi+16].



(a) Window size 5 (b) Window size 10 (c) Window size 20

Figure 2: Bin signals plots created with three different window sizes for chromosome 21. Bigger orange dots correspond to detected TAD boundaries TopDom algorithm.

3.2 Data comparison

We tried different combinations of functions to process data. The results achieved by our approach can be seen in Table 1. In the table we omitted combination that include `smooth` function since it turned out to not produced sensible results. Moreover usage of `qspline` dose not influence the outcome. With parameters that we set for `find_min` and `find_peaks` both give the same results. The number of detected TADs is between 3603 and 538. That shows that used approaches have a big impact on achieved results.

We produced three different metrics to compare results of our approach with different parameters and golden standard. From the table 1 we can see that as the number of detected TADs increases the MoC metrics is higher. Similar trend can be observed for overlap metrics. For combinations of functions producing more TADs the overlap between Arrowhead and them becomes higher. Moreover when one overlap metric increases the another one decreases. For more detail see 3. Arrowhead in general tends to return much more and much smaller TADs, than our approach based on the signal curve as can be seen in Figure 4.

By balancing different metrics we chose results of combination find_peaks and filter_peaks without smoothing as the best.

Table 1: Comparison of results to golden standard produced by Arrowhead depending on different functions used to posses data. Bold row corresponds to original TopDom algorithm.

Minimum finding function	Smoothing function	Filtering function	No. detected TADs	Overlap score (our results, Arrowhead)	Overlap score (Arrowhead, our results)	MoC
find peaks 2	savgol filter	filter peaks	538	0.4188	0.9216	0.17564
find peaks 2	None	filter peaks	570	0.4336	0.8857	0.17276
find peaks 2	qspline	filter peaks	570	0.4336	0.8857	0.17276
find peaks 2	savgol filter	statFilter	910	0.4134	0.8803	0.21736
detect local extrema	savgol filter	statFilter	936	0.4173	0.8617	0.22652

find min	savgol filter	statFilter	968	0.4191	0.8588	0.22945
find peaks	savgol filter	statFilter	968	0.4191	0.8588	0.22945
detect local extrema	savgol filter	filter peaks	1203	0.4182	0.8568	0.25617
find min	savgol filter	filter peaks	1203	0.4182	0.8568	0.25617
find peaks	savgol filter	filter peaks	1203	0.4182	0.8568	0.25617
find peaks 2	None	statFilter	1221	0.4316	0.844	0.24761
find peaks 2	qspline	statFilter	1221	0.4316	0.844	0.24761
detect local extrema	savgol filter	None	1814	0.4215	0.8003	0.30167
detect local extrema	None	filter peaks	1998	0.4437	0.7898	0.31634
find min	None	filter peaks	1998	0.4437	0.7898	0.31634
find peaks	None	filter peaks	1998	0.4437	0.7898	0.31634
detect local extrema	qspline	filter peaks	1998	0.4437	0.7898	0.31634
find min	qspline	filter peaks	1998	0.4437	0.7898	0.31634
find peaks	qspline	filter peaks	1998	0.4437	0.7898	0.31634
find peaks 2	savgol filter	None	1559	0.4264	0.7892	0.27002
find min	savgol filter	None	1968	0.4253	0.7854	0.30988
find peaks	savgol filter	None	1968	0.4253	0.7854	0.30988
find peaks 2	None	None	1776	0.4453	0.7717	0.28754
find peaks 2	qspline	None	1776	0.4453	0.7717	0.28754
detect local extrema	**None**	**statFilter**	**2440**	**0.4478**	**0.7617**	**0.33765**
detect local extrema	qspline	statFilter	2440	0.4478	0.7617	0.33765
find min	None	statFilter	2598	0.4487	0.7559	0.344
find peaks	None	statFilter	2598	0.4487	0.7559	0.344
find min	qspline	statFilter	2598	0.4487	0.7559	0.344
find peaks	qspline	statFilter	2598	0.4487	0.7559	0.344
detect local extrema	None	None	3064	0.4628	0.7203	0.36627
detect local extrema	qspline	None	3064	0.4628	0.7203	0.36627
find min	None	None	3603	0.4744	0.688	0.38158
find peaks	None	None	3603	0.4744	0.688	0.38158
find min	qspline	None	3603	0.4744	0.688	0.38158
find peaks	qspline	None	3603	0.4744	0.688	0.38158

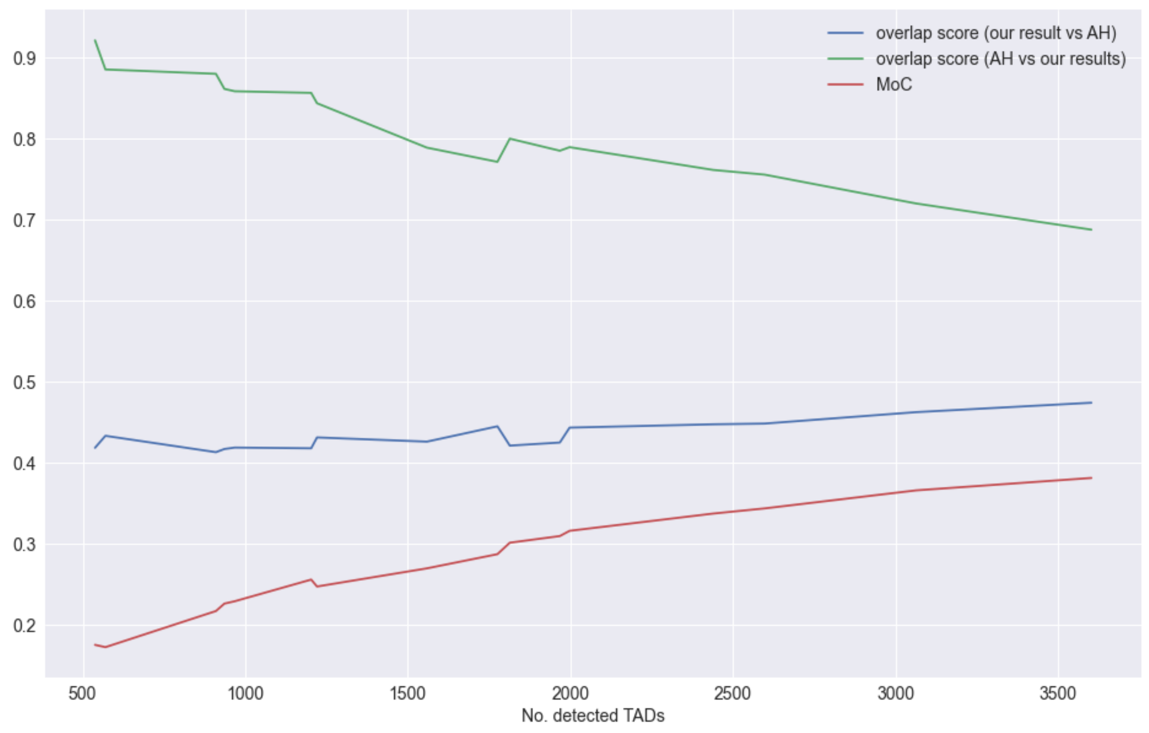


Figure 3: Plot of evaluation metrics depending on number of detected TADs.

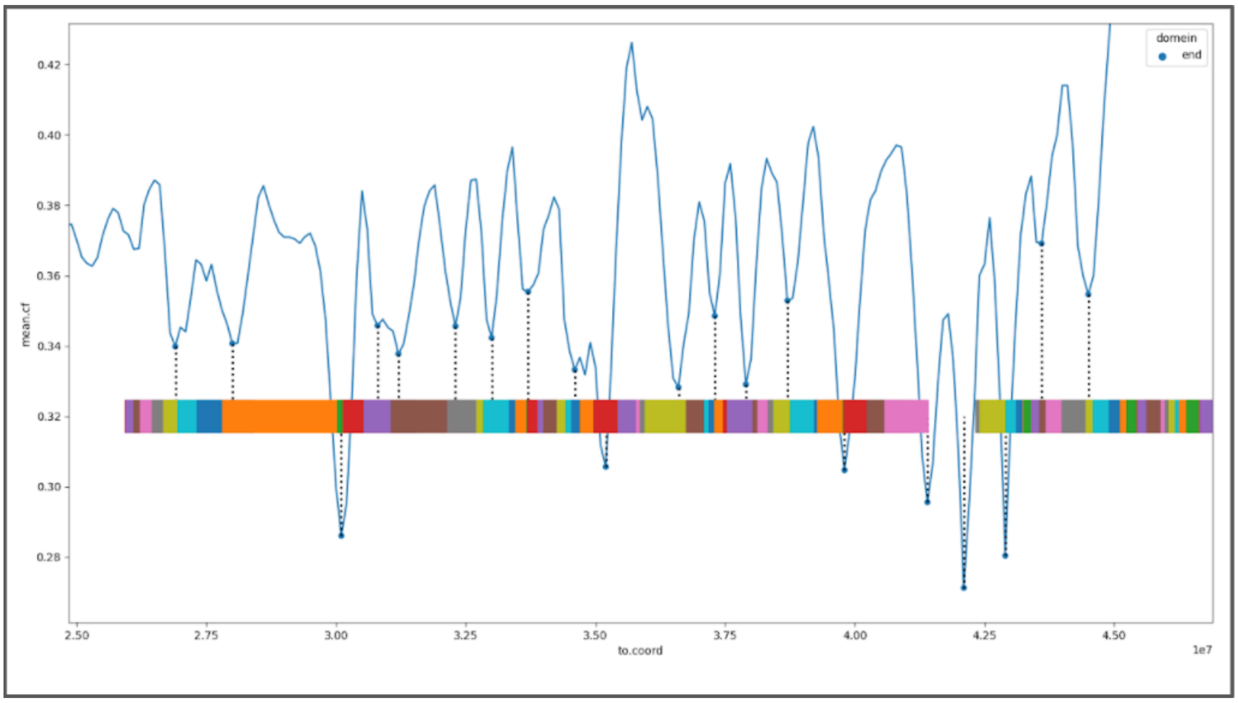


Figure 4: Fragment of chromosome 21. TADs from reference data depicted as rectangles and signal curve received with TopDom (window=5) with marked TAD ends on it.

3.3 Comparison with results of team WA1

Team WA1 was able to detect 2423 TADs. Their approach seems to produce results closer to the gold standard than ours. Overlap for comparisons both ways produced higher values. Also MoC is significantly better than for our method. It is worth noticing that even though our algorithm was able to find more TADs the evaluation metrics were worse. This suggests high number of false positives in our results. For exact metrics see Table 2.

Table 2: Comparison of WA1 results.

Approach 1	Approach 2	Overlap score (approach 1, approach 2)	Overlap score (approach 2, approach 1)	MoC
WA1	Arrowhead	0.6084	0.9342	0.4884
WA1	Our approach	0.5717	0.7169	0.4379

3.4 Attempt to implement CTCF peaks into our algorithm

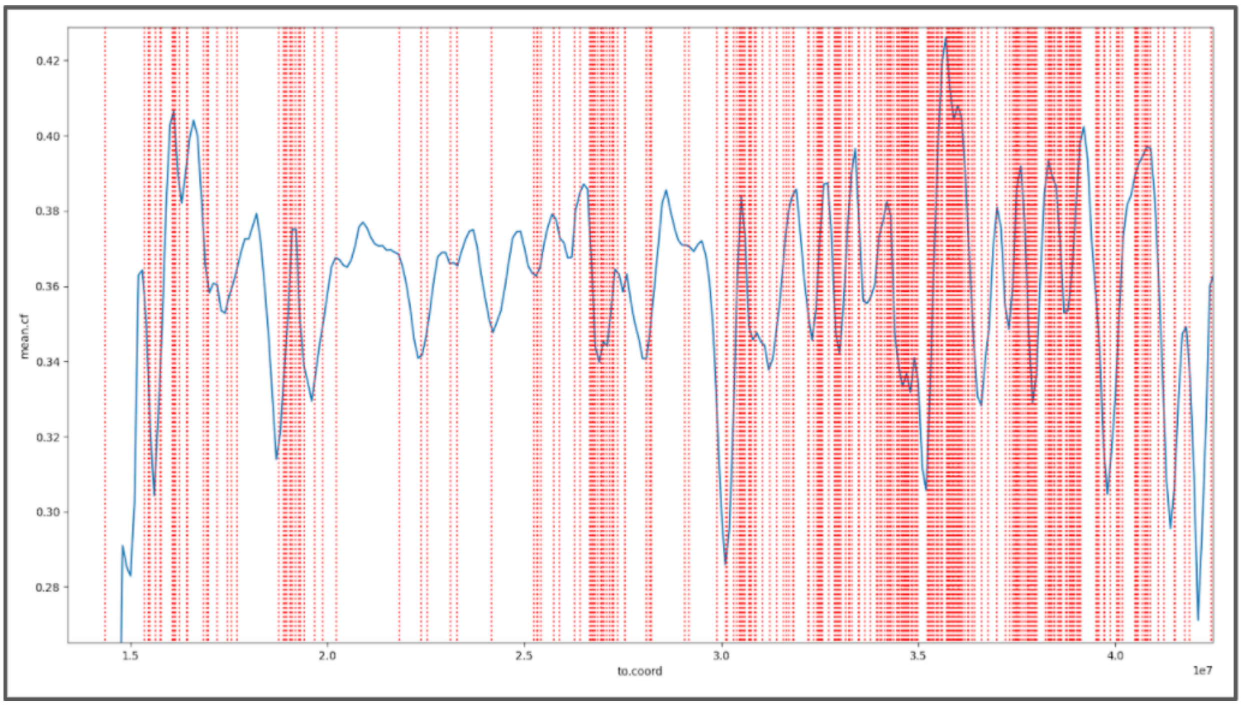


Figure 5: Fragment of chromosome 21. Signal curve received with TopDom plotted against centers of confident CTCF peaks (-log10(pv) > 10).

At first, we planned to implement CTCF chip-seq data into our approach, for TAD boundaries correction. However data downloaded from ENCODE database turned out to be very noisy and inconsistent in regard to our task, even when only very confident peaks (with -log10(pv) > 10) were considered. En example is presented in Figure 5.

4 Discussion

4.1 WA1 team's results

The other student team from Warsaw - WA1, achieved better results in terms of its similarity to reference data. Our results are also closer to theirs, than to reference set. We are not sure however, how did they specifically obtained them. All we now from their GitHub repository is that they also used TopDom algorithm. Whether it was modified or pure version is unclear. In case of the first option, it would be interesting to know their full method and compare each step in order to identify features responsible for differences in resulting TAD sets.

4.2 Reference data

Achieving results similar to the reference set of TADs with our algorithm turned out to be a difficult task. The results of overlapscore varies greatly (usually in range from 0.40 to 0.93) between direction of comparison, as it is not a symmetrical measure. The MoC measure implemented specially because of its symmetry returns even lower values (always less than 0.50). The reasons for this seem to be the differences both in TADs sizes and abundance between our results and reference datasets. It must be due to the shape of signal curve, we based our method on. TADs such short as those produced by arrowhead cannot be obtained with our approach, even with all minimas considered as TAD boundaries Figure 4. En interesting way of comparing results we did not have time to apply would be to merge neighbouring TADs firstly and then using the measure functions. Such approach would mitigate the differences in both TAD number and its size. Additional metric that would be worth considering when number of TAD is significantly different is the number of breakpoints common for two approaches. This way we concentrate on compatibility of TAD boundaries rather than the size.

Moreover additional fine tuning of parameters of specific functions could improve results of our approach. Unfortunately we were not able to conduct that. Another future possible improvement to our method is implementing hierarchical TAD detection. From Table 1 it is obvious that we can report depending on a setup different number of TADs. Further inspection of that problem offers interesting opportunity for improvement.

References

[Rao+14] S. S. Rao et al. "A 3D map of the human genome at kilobase resolution reveals principles of chromatin looping". In: *Cell* 159.7 (2014), pp. 1665–1680.

[Zui+14] Jessica Zuin et al. "Cohesin and CTCF differentially affect chromatin architecture and gene expression in human cells". In: *Proceedings of the National Academy of Sciences* 111.3 (2014), pp. 996–1001. ISSN: 0027-8424. DOI: 10.1073/pnas.1317788111. eprint: https://www.pnas.org/content/111/3/996.full.pdf. URL: https://www.pnas.org/content/111/3/996.

[DH15] Job Dekker and Edith Heard. “Structural and functional diversity of Topologically Associating Domains”. In: *FEBS Letters* 589.20, Part A (2015). 3D Genome structure., pp. 2877–2884. ISSN: 0014-5793. DOI: `https://doi.org/10.1016/j.febslet.2015.08.044`. URL: `https://www.sciencedirect.com/science/article/pii/S0014579315008145`.

[Shi+16] Hanjun Shin et al. “TopDom: an efficient and deterministic method for identifying topological domains in genomes”. In: *Nucleic Acids Research* 44 (2016), e70–e70.

[DB17] R. Dali and M. Blanchette. “A critical assessment of topologically associating domain prediction tools”. In: *Nucleic Acids Res* 45.6 (Apr. 2017), pp. 2994–3005.

[Zuf+18] Marie Zufferey et al. “Comparison of computational methods for the identification of topologically associating domains”. In: *Genome Biology* 19 (2018).

[Vir+20] Pauli Virtanen et al. “SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python”. In: *Nature Methods* 17 (2020), pp. 261–272. DOI: `10.1038/s41592-019-0686-2`.