Efficient Analysis of Annotation Colocalization Accounting for Genomic Contexts

Askar Gafurov[1] Tomáš Vinař[2] Paul Medvedev[3,4,5] Broňa Brejová[1]

[1]Department of Computer Science, Faculty of Mathematics, Physics and Informatics, Comenius University in Bratislava, Slovakia
[2]Department of Applied Informatics, Faculty of Mathematics, Physics and Informatics, Comenius University in Bratislava, Slovakia
[3]Department of Computer Science and Engineering, The Pennsylvania State University, USA
[4]Huck Institutes of the Life Sciences, The Pennsylvania State University, USA
[5]Department of Biochemistry and Molecular Biology, The Pennsylvania State University, USA

Abstract

An annotation is a set of genomic intervals sharing a particular function or property. Examples include genes, conserved elements, and epigenetic modifications. A common task is to compare two annotations to determine if one is enriched or depleted in the regions covered by the other. We study the problem of assigning statistical significance to such a comparison based on a null model representing two random unrelated annotations. Previous approaches to this problem remain too slow or inaccurate.

To incorporate more background information into such analyses and avoid biased results, we propose a new null model based on a Markov chain which differentiates among several genomic contexts. These contexts can capture various confounding factors, such as GC content or sequencing gaps. We then develop a new algorithm for estimating p-values by computing the exact expectation and variance of the test statistics and then estimating the p-value using a normal approximation. Compared to the previous algorithm by Gafurov et al., the new algorithm provides three advances: (1) the running time is improved from quadratic to linear or quasi-linear, (2) the algorithm can handle two different test statistics, and (3) the algorithm can handle both simple and context-dependent Markov chain null models.

We demonstrate the efficiency and accuracy of our algorithm on synthetic and real data sets, including the recent human telomere-to-telomere assembly. In particular, our algorithm computed p-values for 450 pairs of human genome annotations using 24 threads in under three hours. The use of genomic contexts to correct for GC-bias also resulted in the reversal of some previously published findings.

Availability: The software is freely available at https://github.com/fmfi-compbio/mcdp2 under the MIT licence. All data for reproducibility are available at https://github.com/fmfi-compbio/mcdp2-reproducibility

1 Introduction

Recent years have brought rapid growth in the number of different assays that can extract genome-scale functional information. This has led to growing collections of genome annotations; for example in the UCSC Genome browser, the GRCh38 human genome assembly currently features 136 different annotation tracks, many of which have multiple subtracks. In this work, we provide new models and algorithms for annotation colocalization analysis, where the goal is to determine if one input annotation is significantly colocated with regions covered by another annotation. Such analyses may hint at possible connections between biological processes governing individual annotations (e.g. histone modification H3K4me3 binding sites are colocated with promoter regions, and H3K4me3 indeed plays a role in gene transcription regulation [10]).

Mathematically, we view a genome annotation as a set of non-overlapping chromosomal intervals. Given two annotations, query Q and reference R, we consider two widely-used colocalization statistics. The *overlap statistic* is the number of intervals in R that intersect with at least one interval in Q. The *shared bases statistic* is the number of positions in the genome covered by both R and Q. However, even randomly generated annotations will share bases or have overlapping intervals by chance. In order to ascertain statistical significance of the observed statistic, its p-value needs to be computed under a suitable null hypothesis. Until very recently, all the methods [3, 20, 6, 11, 18, 19, 5] were limited by having a null hypothesis that either does not properly model the data or its p-value computation does not scale to annotations of human-sized genomes (see Section 1.1 for details).

Recently, Gafurov et al. [5] proposed an alternative null hypothesis in which the annotation is produced by a two-state Markov chain. The algorithm, called MCDP, was a substantial improvement in time and memory over previous approaches. However, it is quadratic in the number of reference intervals and still takes several hours for a human exon reference annotation. It thus remains time-prohibitive to compare many pairs of annotations against each other.

Another limitation of MCDP as well as other approaches is that two unrelated annotations may appear to be colocalized because they are each colocalized with another genomic feature [12]. For example, two annotations may appear colocalized simply due to their prevalence in high-GC regions, even though they are not related. More generally, different regions of the genome can be thought of as providing different background to the null model, and we think of these various backgrounds as partitioning the genome into *contexts*. Accounting for contexts in calculating p-values is important to limit false associations, yet this capability is limited or absent in existing tools including MCDP (see Section 1.1).

In this paper, we propose a model and an algorithm to overcome these scalability and accuracy barriers. Our first contribution is a new algorithm, called MCDP2, for estimating p-values, which is linear in the number of reference intervals. Using simulated data, we show that our algorithm is also orders-of-magnitude faster then MCDP in practice. To demonstrate the scalability of our algorithm, we considered 10 reference annotations, corresponding to different types of repeats in the human genome, and 45 query annotations, corresponding to epigenetic modifications in different cell lines. Our algorithm computed p-values for all 450 pairs using 24 threads in under 2 hours for the number of overlaps and 3 hours for the number of shared bases.

Our second contribution expands the modeling capability of the Markov chain null hypothesis so that it takes into account genomic context and thus captures various confounding factors influencing annotation colocalization. Unlike previous approaches [11], our model is able to handle annotation intervals that span class boundaries in a natural way. We demonstrate the importance of modeling the genomic context by re-analyzing colocalization of copy number deletions with various gene classes [22] and find that adding a genome context in fact reverses some of the previous conclusions. In one striking example, the set of all exons appears enriched for overlap with copy number losses but enrichment turns into depletion after taking into account gaps and GC content. We also compare the colocalization of epigenetic marks with subtelomeric repeats on the new human telomere-telomere assembly [8], using a genome context to compare enrichment between two classes of repeats.

1.1 Related work

Several null hypotheses for colocalization statistics have been considered prior to the Markov chain model [5]. Some lend themselves to fast and simple statistical tests (e.g. Fisher's exact test) but do not capture relevant properties of the data. For example, one can assume that all positions in the query annotation are chosen uniformly at random [3, 20]. However, this does not capture either the integrity of intervals or their length distribution. A more faithful option is the permutational null hypothesis, which reshuffles the query intervals while maintaining their lengths [6, 11, 18]; it was also called the gold null hypothesis [5]. Computing the exact p-values for the overlap statistic in this model

turns out to be NP-hard [5], and the only known efficient algorithms are either inaccurate or impractical for human-sized annotations [19]. Sampling approaches can be used, but their accuracy is directly proportional to the number of samples, making it difficult to estimate small p-values. With these limitations, it was impossible to compute small p-values for human-sized genomes while having a null hypothesis that is faithful to the data.

Accounting for genomic contexts has also been considered but most previous approaches [3, 6, 16, 20] are only able to account for contexts which are completely inadmissible to annotations (e.g. assembly gaps, which are unassembled regions of the genome). A notable exception is GAT [11], which splits a genome into multiple contexts and analyzes colocalization in each context independently. However, this approach does not satisfactorily handle intervals that span context boundaries, which become prevalent when the contexts are short.

2 Methods

In this section, we define our context-aware Markov chain null model ($\mathcal{H}_0^{\text{context}}$) and provide an overview of our algorithm for efficient estimation of p-values under this model. We first present our results on a single chromosome; an extension to multiple chromosomes is discussed in Section 2.5. We will denote the chromosome length as L, and we number its bases $0, \ldots, L-1$. An *annotation* is a set of intervals contained in $[0, L)$ so that each two intervals are disjoint and separated by at least one base. By $|Q|$ we denote the number of intervals in annotation Q. We will represent an annotation either as an ordered list of half-open intervals with integer coordinates $Q = ([b_1, e_1), \ldots, [b_{|Q|}, e_{|Q|}))$ where $0 \leq b_1 < e_1 < b_2 < e_2 < \cdots < b_{|Q|} < e_{|Q|} \leq L$ or as a binary sequence $Q = (Q_0, Q_1, \ldots, Q_{L-1})$, where Q_i is 1 if position i is covered by one of the intervals and 0 otherwise.

Let R and Q be two annotations, denoted as the reference and the query, respectively. In this setting, a test statistic is a function that measures the extent to which R and Q are colocalized. We will consider two concrete test statistics in this paper. One is the number of overlaps $K(R, Q)$, which is defined as the number of intervals in R that overlap some interval in Q. The other is the number of shared bases $B(R, Q)$, which is defined as the number of bases in the genome covered by both R and Q. Let C be the distribution of the query annotation under the null hypothesis of the query being generated independently of the reference annotation. Given some test statistic $A(R, Q)$, we are interested to compute the p-value measuring the statistical significance of enrichment of Q with respect to R, that is, probability $\Pr_{Q' \sim C}[A(R, Q') \geq A(R, Q)]$.

Our algorithm is based on the observation that the distribution of the test statistics is in most realistic scenarios well approximated by the normal distribution (see Sections 3 and 4). Therefore, instead of computing the full probability mass function (PMF), we compute only its exact mean and variance and use them as the parameters of the normal distribution. This means that we calculate the p-value by first computing the *Z-score*, which is the number of standard deviations that $A(R, Q)$ is above the expected value, under the null. Formally,

$$Z_A(R, Q) = \frac{A(R, Q) - \mathrm{E}_{Q' \sim C}[A(R, Q')]}{\sqrt{\mathrm{Var}_{Q' \sim C}[A(R, Q')]}}$$

Under the assumption that the statistic is normally distributed, the desired p-value is then simply $1 - \Phi(Z_A(R, Q))$, where Φ is the cumulative distribution function of the standard normal distribution. Analogously, the p-value for the statistical significance of depletion, defined as $\Pr_{Q' \sim C}[A(R, Q') \leq A(R, Q)]$, is computed as $\Phi(Z_A(R, Q))$.

In Section 2.1, we describe our context-aware Markov chain model for generating random annotations and then use it to formally define the $\mathcal{H}_0^{\text{context}}$ null model in Section 2.2. In Sections 2.3 and 2.4, we outline our algorithm for computing the mean and variance of the overlap and shared bases test statistics, and we also extend it to a more general class of test statistics. Finally, we describe how our model naturally extends to multiple chromosomes (Section 2.5).

2.1 A generative model

An annotation of a chromosome of length L can be generated by running a two-state Markov chain for L steps. The state at step i indicates whether the annotation includes position i on the chromosome. The lengths of the generated intervals and of the gaps between them are known to be geometrically distributed in this model, and the transition probabilities of the Markov chain dictate the expected values of these two distributions [13]. The Markov chain generative model makes many properties easy to derive and fast to compute [5], and so we build upon it in this work.

We want to use such a generative model to test if a given query annotation Q behaves as if it was "randomly shuffled" on the chromosome. To this end, we set the parameters of the Markov chain so that the expected interval

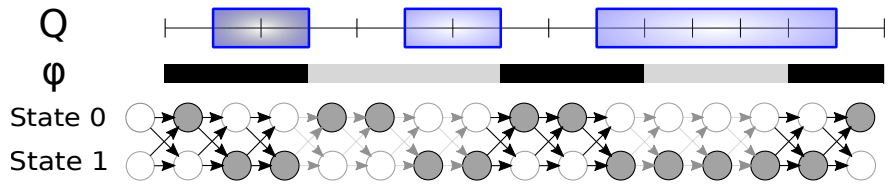

Figure 1: An example of a query annotation $Q = \{[1,3),[5,7),[9,14)\}$ shown as a set of framed boxes and the corresponding sequence of states of the context-aware Markov chain that induces the annotation (shown as filled circles). Genome context ϕ is shown as black and gray bars with colors corresponding to two distinct class labels; the same colors are also used on transition arrows between successive states of the Markov chain, as the transition probabilities depend on the genome context.

lengths and gaps between them match what is observed in the query Q. However, this does not allow to incorporate background knowledge of the chromosome; i.e., some regions of the genome may be *a priori* more likely to generate an interval.

We therefore introduce the notion of genome contexts. Given a finite set of class labels Λ, a *genome context* is a mapping $\phi : \{0,\ldots,L-1\} \longrightarrow \Lambda$ of each position on the genome onto a class label (e.g. $\Lambda = \{\text{gap}, \text{non-gap}\}$). This mapping partitions the genome into several segments with the same class assigned. We will refer to the positions where the class differs from the class label at the previous position as to *class boundaries*. We assume throughout the paper that a context is represented as a sequence of class boundary positions with the corresponding class labels, sorted in an increasing order by positions.

Our generative model allows each context class to have its own Markov chain, i.e. its own distribution of interval lengths and gaps. An annotation is then generated by iterating over the genome positions from left to right, and at each position i transitioning to the next state of the Markov chain according to the transition probabilities of the class at position i (see Figure 1). A similar model was proposed by Burge and Karlin [1] for gene finding; their hidden Markov model uses different transition and emission probabilities based on GC content in the current window of the genome. Formally, our model is defined as follows.

Definition 1. *A* context-aware Markov chain *is a pair* $(\phi, \mathbf{T})$*, where* ϕ *is a genome context and* $\mathbf{T} : \Lambda \to \mathbb{R}^{2\times 2}$ *is a mapping that provides a transition probability matrix for each context class. The context-aware Markov chain* $(\phi, \mathbf{T})$ *generates a sequence of states* $(s_{-1}, s_0, \ldots, s_{L-1}) \in \{0,1\}^{L+1}$ *with probability*

$$\Pr[(S_{-1}, S_0, \ldots, S_{L-1}) = (s_{-1}, s_0, \ldots, s_{L-1})] = \vec{\pi}_{s_{-1}} \cdot \prod_{i=0}^{L-1} \mathbf{T}(\phi(i))_{s_{i-1},s_i},$$

where

- $\mathbf{T}(\phi(i))_{s,s'}$ *is the probability of transition from state* s *to state* s' *in context class* $\phi(i)$*,*
- $\vec{\pi}_s$ *is the probability of state* s *in the stationary distribution of the Markov chain with transition probabilities* $\mathbf{T}(\phi(0))$*. Specifically,*

$$\vec{\pi} = \left(\frac{\mathbf{T}(\phi(0))_{1,0}}{1 - \mathbf{T}(\phi(0))_{0,0} + \mathbf{T}(\phi(0))_{1,0}} \quad \frac{1 - \mathbf{T}(\phi(0))_{0,0}}{1 - \mathbf{T}(\phi(0))_{0,0} + \mathbf{T}(\phi(0))_{1,0}} \right)$$

Note that we are indexing vectors and matrices starting from 0 in order to make the formulas more readable. The produced binary sequence of states $(s_0, \ldots, s_{L-1})$ can be viewed as an annotation of a genome of size L. State s_{-1} is added to the start for notational convenience, as we will often refer to the state preceding the start of an interval. The distribution of the random vector of generated states $(S_{-1}, S_0, \ldots, S_{L-1})$ will be denoted as $C(\phi, \mathbf{T})$. We will use the same notation to denote the distribution of the induced annotation.

2.2 The context-aware Markov chain null model

The generative model of the previous section serves a basis for our null model, which we call $\mathcal{H}_0^{\text{context}}$. Given a context ϕ and a query annotation $Q = (Q_0, \ldots, Q_{L-1})$, we first need to find the transition probabilities $\mathbf{T}_Q$ that would

maximize the probability of the context-aware Markov chain $(\phi, \mathbf{T}_Q)$ generating Q. This is achieved through the standard approach of training Markov chains by counting transition frequencies [4]. In particular, for each class, we count the number of times each possible state transition occurs in Q. Formally,

$$c_{\lambda,s,s'} = 1 + \sum_{i=1}^{L-1} \mathbb{1}((\phi(i) = \lambda) \text{ and } (s = Q_{i-1}) \text{ and } (s' = Q_i)),$$

where $\mathbb{1}$ is the indicator function which evaluates to 1 if the logical expression inside is true and 0 otherwise. A pseudocount 1 is added to avoid zero probabilities [4]. The transition matrix is then defined from these counts as

$$\mathbf{T}_Q(\lambda) = \begin{pmatrix} \frac{c_{\lambda,0,0}}{c_{\lambda,0,0} + c_{\lambda,0,1}} & \frac{c_{\lambda,0,1}}{c_{\lambda,0,0} + c_{\lambda,0,1}} \\ \frac{c_{\lambda,1,0}}{c_{\lambda,1,0} + c_{\lambda,1,1}} & \frac{c_{\lambda,1,1}}{c_{\lambda,1,0} + c_{\lambda,1,1}} \end{pmatrix}.$$

The mapping $\mathbf{T}_Q$ is computable in time $O(|Q| + c)$ and space $O(|\Lambda|)$, where c is the number of class boundaries. We can now formally define the context-aware Markov chain null hypothesis.

Definition 2. *The context-aware Markov chain null hypothesis ($\mathcal{H}_0^{context}$) for query annotation Q and context $\phi : \{0, \ldots, L-1\} \to \Lambda$ is that the query annotation Q is generated by the context-aware Markov chain $(\phi, \mathbf{T}_Q)$.*

Note that under context ϕ with a single class, the context-aware Markov chain null hypothesis $\mathcal{H}_0^{\text{context}}$ reduces to the Markov chain null hypothesis by Gafurov et al. [5], with a small difference that the initial state distribution $\vec{\pi}$ is set to the stationary distribution at position -1 instead of always starting in state 0 at position 0.

When there is just one context class, $\mathcal{H}_0^{\text{context}}$ can be viewed as an approximation to the permutational null, i.e. shuffling the query intervals around in a random fashion [5]. In the case of multiple classes, $\mathcal{H}_0^{\text{context}}$ can be thought of as an approximation to shuffling the query intervals around separately within each class. However, $\mathcal{H}_0^{\text{context}}$ also transparently handles intervals spanning one or even multiple class boundaries.

2.3 Computing the mean and variance of the overlap and shared bases test statistics

Here, we state our main algorithmic result: fast computation of the expected value and variance of $K(R, Q)$ and $B(R, Q)$ statistics under $\mathcal{H}_0^{\text{context}}$. These are then used to compute the p-values using the normal approximation.

Theorem 1. *Let R and Q be two annotations and let ϕ be a genome context with c class boundaries. Let A be either the number of overlaps test statistic (i.e. K) or the number of shared bases test statistic (i.e. B). It is possible to compute mean $\mathrm{E}_{Q' \sim C(\phi, \mathbf{T}_Q)}[A(R, Q')]$ and variance $\mathrm{Var}_{Q' \sim C(\phi, \mathbf{T}_Q)}[A(R, Q')]$ in space $O(|R| + c)$ and in time*

- *$O(|R| + |Q| + c)$ when A is the overlap test statistics,*
- *$O(|Q| + (|R| + c) \log t)$ when A is the shared bases test statistics, where t is the length of the longest stretch of positions within a single reference interval with the same context class in R.*

Under the assumption that the test statistic is approximately normally distributed, this algorithm can be used to obtain the full probability mass function of its distribution under the null, i.e. values $\Pr_{Q' \sim C(\phi, \mathbf{T}_Q)}[A(R, Q') = x]$ for all values of x. Note that MCDP, the previous algorithm for this problem, only works with one context, only works for the overlap statistic K, and runs in $O(|R|^2 + |Q|)$ time [5]. However, it makes no assumption about normality.

In the next section, we provide a high-level description of the algorithm for computing expectation and variance, leaving the details, including additional notation and a detailed proof of correctness, for the Appendix.

2.4 Mean and variance of any separable statistic

To reuse parts of the algorithm for both K and B, we give a more general algorithm to compute the expectation and variance for a family of test statistics we call *separable* (not to be confused with *separable statistics* used by Y. Medvedev [14]). A statistics is separable if it can be expressed as a sum of contributions from each reference interval and each contribution depends only on the part of the query annotation inside this reference interval. For example, the contribution of each reference interval in the overlap statistics K is 1 or 0, depending on whether there

is an overlap with some query interval. In the shared bases statistics, the contribution is the number of bases that a particular reference interval shares with the query intervals. The formal definition is given in the Appendix.

Thanks to linearity of expectation, the expected value of any separable statistic can be computed for every reference interval separately and then summed together. The simplest case is the shared bases statistic B, which can be expressed as the sum of indicator variables for each base covered by R, and in the case of single-class null model, the expectation can be computed simply as the number of bases covered by R multiplied by the stationary probability of state 1 of the Markov chain. Context-aware models complicate the situation, as each base of the genome has its unique marginal distribution over states, depending on the sequence of class labels preceding it.

Computing variance is more complicated, as the values of the statistic in individual intervals of R are dependent, and therefore the overall variance is not a simple sum of individual variances. However, in a sequence of Markov chain states $(S_0, \ldots, S_{L-1})$, states S_i and S_j are conditionally independent given $S_k = s$ for $i \leq k \leq j$. Therefore, if random variable X is a function of $S_i, \ldots, S_{k-1}$ and random variable Y is a function of $S_k, \ldots, S_{j-1}$, then $\mathrm{Var}[X + Y \mid S_k = s] = \mathrm{Var}[X \mid S_k = s] + \mathrm{Var}[Y \mid S_k = s]$. Our algorithm computes conditional variance in individual intervals of R conditioning on states at both interval boundaries, and then combines them using this formula. In order to remove conditioning on the boundary states, we use the law of total variance, which for binary variable S_k can be written as

$$\begin{aligned}\mathrm{Var}(X) = &\sum_{s=0}^{1} \mathrm{Var}[X \mid S_k = s] \Pr[S_k = s] + \\ &+ \sum_{s=0}^{1} E[X \mid S_k = s]^2 \Pr[S_k = s](1 - \Pr[S_k = s]) - \\ &- 2 \prod_{s=0}^{1} E[X \mid S_k = s]^2 \Pr[S_k = s]\end{aligned}$$

This is formally stated as Lemma 1 in the Appendix. The key data structure in our algorithm is a $O(1)$-sized vector called a *two-sided plumbus* defined below. It contains the quantities that we need to compute for every interval of R, conditioning on states at interval boundaries. In the definition, function v expresses the contribution of a reference interval to the separable test statistics.

Definition 3. *Let $(\phi, \mathbf{T})$ be a context-aware Markov chain with state sequence $S_{-1}, S_0, \ldots S_{|L|-1}$, let $[i, j)$ be a subinterval of $[0, L)$, and let v be a function on a binary sequence of length $j - i$. We define the* two-sided plumbus *for interval $[i, j)$ as the collection of values*

$$\begin{aligned}\mu_v(i, j \mid x, y) &:= \mathrm{E}[v(S_i, \ldots, S_{j-1}) \mid S_{i-1} = x, S_{j-1} = y] \\ \sigma_v^2(i, j \mid x, y) &:= \mathrm{Var}[v(S_i, \ldots, S_{j-1}) \mid S_{i-1} = x, S_{j-1} = y] \\ \Psi(x \overset{i-1,j-1}{\rightarrow} y) &:= \Pr[S_{j-1} = y \mid S_{i-1} = x]\end{aligned}$$

for all combinations of (x, y) in $\{0, 1\}^2$.

The two-sided plumbuses computed for individual intervals of R and gaps between them are then combined to plumbuses for successively longer intervals, until we cover the whole chromosome and obtain the overall variance and expected value of the statistic of interest (see Figure 2).

In the algorithm, we compute $\Pr[S_j = y \mid S_i = x]$ and $\Pr[S_i = S_{i+1} = \cdots = S_j = 0]$ in constant time, provided that interval $[i, j]$ is labeled by the same context class. This leads to a linear-time algorithm for $K(R, Q)$ statistics. For $B(R, Q)$ statistics, we split an interval of R into subintervals of size 1, compute plumbuses for them, and combine them in a similar manner, as we combine plumbuses in the overall algorithm. However, within a single context class, corresponding plumbus depends only on the interval length, and therefore we can compute plumbuses for interval sizes which are powers of two and combine them to obtain a plumbus for any interval length within a single context in logarithmic time. Further details of the algorithm and its proof are given in the Appendix.

2.5 Multiple chromosomes

Both our model and our algorithm can be extended to genomes with multiple chromosomes in a straightforward way. We assume that the query annotation is generated independently for each chromosome. The training of the context-

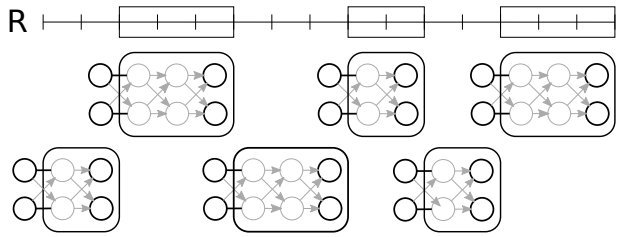

Figure 2: An example of a reference annotation R (shown as framed boxes) and the corresponding two-sided plumbuses. The plumbuses in the first row correspond to the reference intervals, and the plumbuses in the second row correspond to the gaps between the intervals. We highlight in black the boundary states on which we condition the values in each plumbus. Note that the conditional means μ_v and variances σ_v^2 in the gap plumbuses are constant zeroes, since gaps do not contribute to the total test statistic.

aware Markov chain is accomplished simply by counting transition frequencies on all chromosomes. The test statistic for the whole genome is defined as the sum of test statistic values for the individual chromosomes. This, in turn, allows to compute the mean and variance of the total statistic by summing the means and variances, respectively, for the individual chromosomes. Note that this simple computation works for the variance thanks to the chromosome independence assumption. Therefore, the time and space complexity of our algorithm remains the same for the case of multiple chromosomes.

3 Experiments

The normal distribution yields an accurate p-value approximation. Our MCDP2 algorithm computes the exact expectation and variance of the null distribution and uses them to approximate the null distribution by the normal distribution. Here, we first compare the accuracy of this approximation for the $K(R, Q)$ statistic with the exact distribution computed by the previous MCDP algorithm [5]. The comparison was performed on synthetic data sets with genome length $L = 10^8$ bp, query annotations with 50 000 randomly generated intervals of length 500 bp each and reference annotations with up to 20 000 intervals of length 500 bp each. To understand the influence of the number of reference intervals on the accuracy, we vary $|R|$ from 200 to 20000.

Figure 3 shows that the exact PMF in general agrees well with the normal approximation. The approximation approach allows to estimate even very low p-values accurately with the growing number of reference intervals. However, for $|R| = 200$ the differences in the extreme tail of the distribution lead to overly conservative p-values. Therefore, for small values of $|R|$ we recommend the use of the exact MCDP algorithm, which is not time-prohibitive. The new MCDP2 tool includes a reimplementation of the exact computation of the PMF and its extension to multiple context classes, and we use it in these experiments under the label MCDP*.

We have conducted similar experiments for the number of shared bases statistic $B(R, Q)$, which is more complicated because the original MCDP algorithm worked only for the $K(R, Q)$ statistic. After adapting it for the $B(R, Q)$ statistic, it works in time quadratic in the total length of reference intervals, restricting our comparison to small reference annotations. Details of the experiments can be found in the Appendix S2. We observe similar patterns as for the $B(K, R)$ statistics (Appendix, Figure S2). Namely, the central part of the PMF is well approximated already for small annotations with 40 intervals, and the approximation of the tail is also improving with the growing size of the reference annotation.

MCDP2 is fast and memory efficient. The speed of our algorithm enables us to apply our tools to large-scale comparisons, such as the data from a recent study of ENCODE epigenetic modification enrichment for different repeat types in the human genome [8], employing the Telomere-to-Telomere (T2T) human genome assembly [15]. We use a context with two classes, one corresponding to all repeats and one to the rest of the genome, leading to over 4 million class boundaries. We use one of the 10 repeat types as the reference and one of the 45 available combinations of an epigenetic modification and a cell line as the query. Using 24 CPU threads, MCDP2 computed p-values for all 450 pairs in approx. 2 hours (wall clock) for the number of overlaps and approx. 3 hours for the number of shared

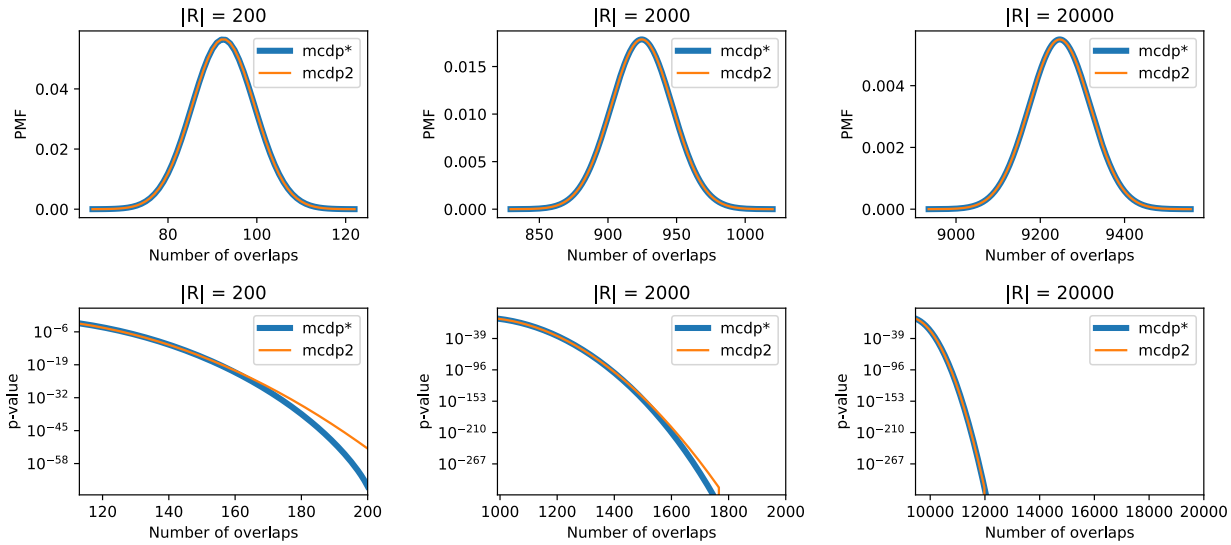

Figure 3: The comparison of the exact probability mass functions (PMFs) for the number of overlaps K statistic (MCDP*) with its normal approximation (MCDP2) on synthetic data sets. Each column represents a different number of reference intervals ($|R| \in \{200, 2000, 20000\}$). The bottom plots show differences for the extreme tail of the distribution. The curves represent the exact and approximated p-value for different values of the statistic, starting from the position with Z-score +3. The lines in the second and the third plots eventually go below the minimum positive value representable as a double-precision floating point number (i.e. 10^{-308}).

bases, using at most 4.2 Gb (2.3 Gb) memory per comparison for overlaps (shared bases) statistic. The running times for each repeat type are shown in Table 1.

We compare the running time of MCDP2 to the quadratic-time MCDP* algorithm on the synthetic data sets used for Figure 3, with 20 pairs of R and Q generated for each setting. Note that MCDP* is faster than the original MCDP implementation [5], thanks to more extensive use of `numpy` library and reimplementation of part of the algorithm in C++. Figure 4 shows the result of the comparison. For the overlaps statistic, the MCDP* needs more than 1 000 seconds for 20 000 reference intervals, while our new approach MCDP2 only takes approximately 8 seconds on the same inputs. Computation for the number of shared bases is slightly slower (23 seconds for $|R| = 20\,000$), which is consistent with its quasi-linear time complexity (in contrast to purely linear for the number of overlaps).

Table 1: Data set sizes and average running times for comparison of repeat types (R) with ENCODE epigenetic modifications (Q), using all repeats as context. Averages are computed across 45 different query annotations, each representing a specific epigenetic modification in a specific cell line. Note that the running time grows with $|R| + c$, and in this experiment, c is large even for inputs with small $|R|$.

		time (s)	
Repeat type	$\|R\|$	$K(R, Q)$	$B(R, Q)$
RNA	11 139	220	160
Other	8 835	220	157
Unknown	11 229	226	159
Satellite	47 041	229	202
Low-complexity	102 521	244	223
DNA	505 896	343	588
LTR	660 823	387	769
Simple	708 565	398	637
LINE	1 440 792	578	1 477
SINE	1 672 984	640	1 637

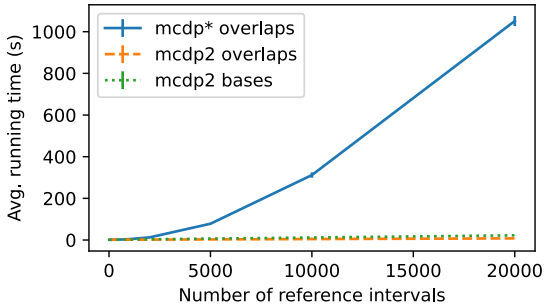

Figure 4: Average running time on synthetic data for both MCDP* and MCDP2 on the overlaps statistics and MCDP2 on the shared bases statistics. The vertical bars represent the standard deviation over 20 samples. The calculations were performed on a single thread on Intel(R) Xeon(R) Gold 6248R CPU.

Genome contexts enable more detailed analysis of colocalization of copy number loss with different gene groups. To illustrate the power of our context-aware null model, we have reanalyzed the colocalization of exons of different gene groups with copy number loss regions, originally performed by Zarrei et al. [22]. Figure 5 shows the Z-scores for both K and B test statistics and for three types of contexts. The first context function only uses a single class. The second context function creates two classes by masking regions that are assembly gaps; this is motivated by the fact that both copy number losses and exons are annotated exclusively outside the gaps and, therefore, may appear colocated even if they were independent. The influence of biases caused by gaps was studied by Domanska et al. [2]. The third context function uses six classes: one for gaps and the other five for discretization of the GC content in 1 kbp windows; this is motivated by the fact that GC content is known to be a significant confounding factor in many genomic analyses [1, 7].

Figure 5 illustrates the importance of having a class in the context dedicated to gaps. In one jarring scenario, the set of all exons is enriched for overlaps with copy number losses (enrichment was also observed by Zarrei et al.), but after accounting for gaps, the exons become depleted. More generally, we find that across all studied gene groups, the Z-score decreases when the gaps are taken into account. This is expected as neither exons nor copy number losses occur in gaps, and thus ignoring gaps in the analysis may create spurious enrichments or lower the degree of observed depletion compared with analysis that takes gaps into account.

The GC-aware context also proves crucial for an accurate analysis. For example, the depletion of all exons for overlaps with copy number losses becomes much more pronounced in the GC-aware context than in the gap-aware context. A more detailed analysis (Appendix S3 and Appendix Table S1) shows that the overlaps are depleted in all GC contexts but the most pronounced depletion is at the lowest GC level. In another striking example, genes with no known phenotype appear enriched for overlap with losses (also in agreement with Zarrei et al.) when using the gap-aware context, but enrichment turns into slight depletion after taking GC content into account.

Other observations from Figure 5 are generally consistent with biological expectations. Protein coding genes are only slightly depleted in the single-class context but become significantly depleted when using gap-aware or GC-aware contexts. This depletion is consistent with the expectation that protein-coding exons are mostly evolutionary conserved. Interestingly, the set of non-coding genes is strongly enriched for copy number losses under all three context functions, and the enrichment was also observed by Zarrei et al. Other categories of genes shown in Figure 5 are generally depleted under all models.

Differential analysis of non-telomeric and telomeric TAR elements. Completion of the previously inaccessible parts of the human genome [15] has allowed Gershman et al. [8] to study telomere-associated repeats (TARs) and their colocalization with epigenetic modifications. While TARs located in subtelomeric regions are presumed to be important for telomere length regulation, TAR copies have also been dispersed to other parts of the genome. Differences between these two groups may further clarify mechanisms and functions of TARs in subtelomeric regions. While Gershman et al. observe differences in enrichment of some epigenetic marks, they do not assign statistical significance to their findings [8, Figure S21].

We adapted our context-aware Markov chain model to perform such differential analysis of enrichment between two annotations. In general, consider two references $R_1 \subseteq R_2$. In our case, R_1 are non-telomeric TARs, R_2 are all

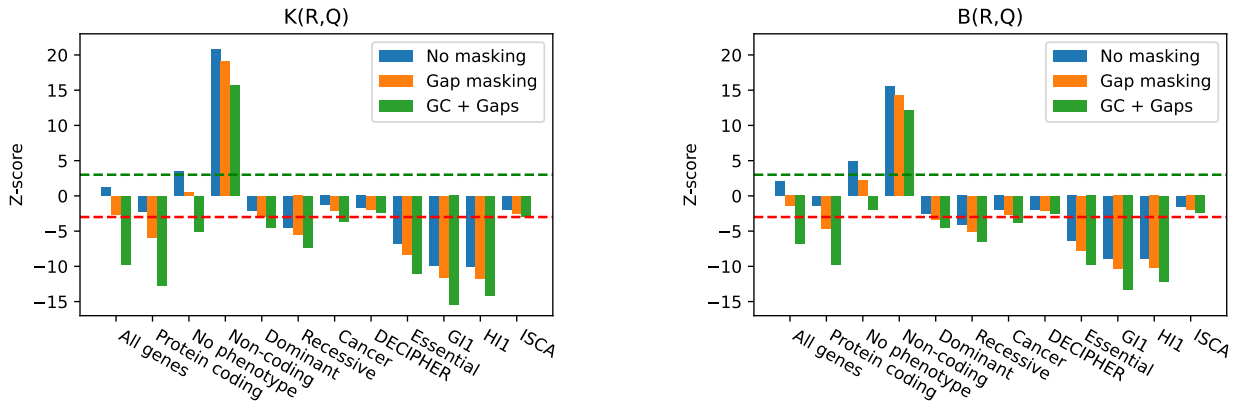

Figure 5: Z-scores for colocalization of exons of various gene groups (R, x-axis) with copy number losses (Q) under three different null models: single-class context, gap-aware context, and GC–aware context. The top panel shows the overlap test statistic K and the bottom panel shows the shared bases test statistic B. The green and red dashed lines stand for Z-score +3 and -3 respectively; these scores correspond (in the normal distribution) to p-value 0.00135 for enrichment and depletion, respectively.

TARs, and Q are regions with a particular epigenetic mark. One could compare the enrichment p-value of Q in R_1 with the enrichment p-value of Q in R_2; however, this is not statistically sound, as p-values should not generally be compared with each other [9, 21].

Instead, we create a context ϕ_{rel} with two class labels $\{\text{outside}, \text{inside}\}$, where positions covered by R_2 are labeled "inside" and all other positions are labeled "outside." We then use a test statistic to measure the significance of enrichment of Q in reference R_1 with context ϕ_{rel}. This context ensures that within R_1 the null model uses the parameters estimated from intervals of Q that overlap R_2, thus comparing colocalization of Q in R_1 relative to colocalization of Q in the whole R_2. Note that the query intervals can occur also outside of R_2, and their properties are summarized in the parameters of the Markov chain for the "outside" class. These outside areas then influence the distribution of the test statistic under the null only by influencing the initial state distribution at the start of each interval of R_1.

Figure 6 shows the result of such analysis applied to colocalization of epigenetic marks in non-telomeric TARs compared to all TARs. Similarly to Gershman et al. [8] we observe relative enrichment of activating marks *H3K27ac* and *H3K4me3* in non-telomeric TARs using both K and B statistics. We can also see enrichment of *CTCF*, which is significant only under the shared bases statistic, perhaps due to the small number of intervals in R_1. Gershman et al. were not able to observe relative enrichment for *CTCF* on non-telomeric TARs, although they do observe that *CTCF* is strongly enriched in both TAR classes compared to the background. This highlights usefulness of our context model in scenarios requiring relative analysis of two reference annotations.

4 Discussion

In this paper, we have provided a novel model for annotation colocalization analysis, which uses genomic contexts to capture confounding factors that may lead to false significance results. Taking advantage of the Markovian properties of our model, we have provided a general framework to compute the exact mean and variance of a broad class of colocalization test statistics (which we named *separable*). Using this framework, we were able to obtain linear and quasi-linear algorithms to compute the Z-scores for the number of overlaps and the number of shared bases, respectively, which are two frequently used statistics. We have then proposed to convert the exact Z-score to approximate the p-values using the normal distribution.

Our algorithm computes a Z-score in $O(|Q|+|R|+c)$ time for the overlap number statistic and in $O(|Q|+(|R|+c)\log t)$ time for the shared bases statistic, where $|Q|$ and $|R|$ are the number of intervals in the query and reference, respectively, c is the number of context class switches along the genome, and t is an upper bound on the reference interval length. This is in contrast to the previous best algorithm, which did not account for genome contexts and took $O(|R|^2 + |Q|)$ time to compute the probability mass function of the p-values.

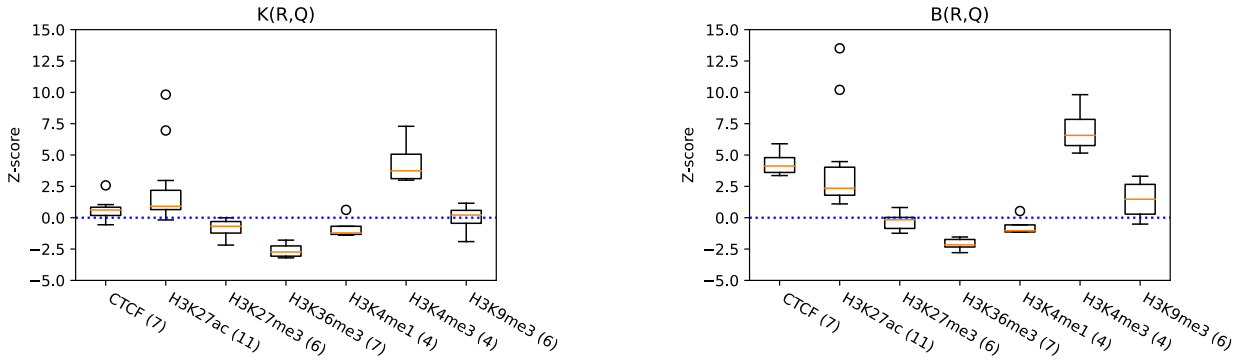

Figure 6: Z-scores for relative enrichment of telomere-associated repeats (TARs) located further than 20 kbp from chromosome ends (R_1) with epigenetic modifications (Q) in comparison to all TARs (R_2), using both the number of overlaps statistic K (top) and the number of shared bases statistics B (bottom). Each box summarizes Z-scores of available cell lines for one epigenetic modification in the T2T ENCODE data set. The numbers in the parentheses denote the number of different cell lines available for each modification.

In our experiments, we have demonstrated that our algorithm is sufficiently fast to allow large-scale studies comparing many pairs of annotations with large reference sets and frequent context class boundaries. We have reanalyzed data sets from two large-scale studies [8, 22], and thanks to our new context-aware model, we were able to further illuminate the nature of colocalizations discovered in these works, in some cases reversing previously published findings.

We have experimentally shown that the normal approximation of the distribution of the number of overlaps under $\mathcal{H}_0^{\text{context}}$ yields accurate p-values, and the approximation gets tighter with increasing number of reference annotation intervals (Figure 3). This behaviour intuitively follows from the representation of a separable statistic as a sum of contributions for individual reference intervals. If those contributions were independent, their sum would converge to a normal distribution with a growing number of reference intervals under the classical central limit theorem. Though the contributions of individual intervals are dependent in our case, the fact that the dependencies stem from a Markov chain makes it possible that the sum converges under some extensions of the central limit theorem. In future, we hope to find sufficient conditions for convergence of the true distribution of a separable statistic under $\mathcal{H}_0^{\text{context}}$ to the normal distribution. Additionally, we would like to explore the possibility of providing lower and upper bounds on the precision of the p-value estimation, possibly by applying the Stein's method [17].

On a more practical side, in our future research, we would like to address the possibility of using quantitative rather than qualitative contexts, with numeric values such as GC content, epigenetic mark density, sequence conservation etc. Some work in this direction has already been done, particularly in HyperBrowser [18]. In MCDP2 this could be achieved for example by parameterizing the weights of the underlying Markov chains with the context value at each position. The challenge would be to keep the running time efficient for large genomes.

Another challenge is to provide statistical significance for statistics comparing colocalization of query Q with respect to two different references R_1 and R_2, such as $B(R_1, Q)/B(R_2, Q)$. This may in some situations be preferable to our approach of comparing such colocalization through contexts, which we used for the analysis of TAR elements.

Funding. This material is based upon work supported by the National Science Foundation under Grant No. DBI-2138585. Research reported in this publication was supported by the National Institute Of General Medical Sciences of the National Institutes of Health under Award Number R01GM146462. The content is solely the responsibility of the authors and does not necessarily represent the official views of the National Institutes of Health. This work was also supported by a grant from the European Union Horizon 2020 research and innovation program No. [872539] (PANGAIA); and grants from the Slovak Research and Development Agency [APVV-22-0144] and the Scientific Grant Agency VEGA [1/0463/20, 1/0538/22].

References

[1] Chris Burge and Samuel Karlin. "Prediction of complete gene structures in human genomic DNA". In: *Journal of Molecular Biology* 268.1 (1997), pp. 78–94.

[2] Diana Domanska, Chakravarthi Kanduri, Boris Simovski, and Geir Kjetil Sandve. "Mind the gaps: overlooking inaccessible regions confounds statistical testing in genome analysis". In: *BMC Bioinformatics* 19.1 (Dec. 2018), p. 481. DOI: 10.1186/s12859-018-2438-1.

[3] Mikhail G Dozmorov, Lukas R Cara, Cory B Giles, and Jonathan D Wren. "GenomeRunner web server: regulatory similarity and differences define the functional impact of SNP sets". In: *Bioinformatics* 32.15 (2016), pp. 2256–2263.

[4] Richard Durbin, Sean R Eddy, Anders Krogh, and Graeme Mitchison. *Biological sequence analysis: probabilistic models of proteins and nucleic acids*. Cambridge University Press, 1998.

[5] Askar Gafurov, Broňa Brejová, and Paul Medvedev. "Markov chains improve the significance computation of overlapping genome annotations". In: *Bioinformatics* 38.Supplement_1 (2022), pp. i203–i211.

[6] Bernat Gel, Anna Diez-Villanueva, Eduard Serra, et al. "regioneR: an R/Bioconductor package for the association analysis of genomic regions based on permutation tests". In: *Bioinformatics* 32.2 (2016), pp. 289–291.

[7] Sahar Gelfman and Gil Ast. "When epigenetics meets alternative splicing: the roles of DNA methylation and GC architecture". In: *Epigenomics* 5.4 (2013), pp. 351–353.

[8] Ariel Gershman, Michael E.G. Sauria, Xavi Guitart, et al. "Epigenetic patterns in a complete human genome". In: *Science* 376.6588 (2022). ISSN: 10959203. DOI: 10.1126/science.abj5089.

[9] Steven Goodman. "A dirty dozen: twelve p-value misconceptions". In: *Seminars in hematology*. Vol. 45. 3. Elsevier. 2008, pp. 135–140.

[10] Matthew G. Guenther, Stuart S. Levine, Laurie A. Boyer, Rudolf Jaenisch, and Richard A. Young. "A Chromatin Landmark and Transcription Initiation at Most Promoters in Human Cells". In: *Cell* 130.1 (2007), pp. 77–88. DOI: 10.1016/j.cell.2007.05.042.

[11] Andreas Heger, Caleb Webber, Martin Goodson, Chris P Ponting, and Gerton Lunter. "GAT: a simulation framework for testing the association of genomic intervals". In: *Bioinformatics* 29.16 (2013), pp. 2046–2048.

[12] Chakravarthi Kanduri, Christoph Bock, Sveinung Gundersen, Eivind Hovig, and Geir Kjetil Sandve. "Colocalization analyses of genomic elements: Approaches, recommendations and challenges". In: *Bioinformatics* 35.9 (2019), pp. 1615–1624. ISSN: 14602059. DOI: 10.1093/bioinformatics/bty835.

[13] Daphne Koller and Nir Friedman. *Probabilistic graphical models: principles and techniques*. MIT press, 2009.

[14] Yurii Medvedev. "Separable statistics in a polynomial scheme. I". In: *Theory of Probability & Its Applications* 22.1 (1977), pp. 1–15.

[15] Sergey Nurk, Sergey Koren, Arang Rhie, et al. "The complete sequence of a human genome". In: *Science* 376.6588 (Apr. 2022), pp. 44–53. DOI: 10.1126/science.abj6987.

[16] Aaron R Quinlan and Ira M Hall. "BEDTools: a flexible suite of utilities for comparing genomic features". In: *Bioinformatics* 26.6 (2010), pp. 841–842.

[17] Nathan Ross. "Fundamentals of Stein's method". In: *Probability Surveys* 8.1 (2011), pp. 210–293. ISSN: 15495787. DOI: 10.1214/11-PS182.

[18] Geir K. Sandve, Sveinung Gundersen, Halfdan Rydbeck, et al. "The Genomic HyperBrowser: Inferential genomics at the sequence level". In: *Genome Biology* 11.12 (2010). ISSN: 14747596. DOI: 10.1186/gb-2010-11-12-r121.

[19] Shahab Sarmashghi and Vineet Bafna. "Computing the Statistical Significance of Overlap between Genome Annotations with ISTAT". In: *Cell Systems* 8.6 (2019), pp. 523–529. ISSN: 24054720. DOI: 10.1016/j.cels.2019.05.006. URL: https://doi.org/10.1016/j.cels.2019.05.006.

[20] Nathan C. Sheffield and Christoph Bock. "LOLA: Enrichment analysis for genomic region sets and regulatory elements in R and Bioconductor". In: *Bioinformatics* 32.4 (2016), pp. 587–589. DOI: 10.1093/bioinformatics/btv612.

[21] Gail M Sullivan and Richard Feinn. "Using effect size—or why the P value is not enough". In: *Journal of graduate medical education* 4.3 (2012), pp. 279–282.

[22] Mehdi Zarrei, Jeffrey R. MacDonald, Daniele Merico, and Stephen W. Scherer. "A copy number variation map of the human genome". In: *Nature Reviews Genetics* 16.3 (2015), pp. 172–183. ISSN: 14710064. DOI: 10.1038/nrg3871.

Appendix: Efficient Analysis of Annotation Colocalization Accounting for Genomic Contexts

S1 Algorithm details

In this section, we provide additional details of our algorithm for computing mean and variance of separable statistics K and B under the $\mathcal{H}_0^{\text{context}}$. We start with a formal definition of separable statistics.

Definition 4. *Let $R = ([b_1, e_1), \ldots, [b_{|R|}, e_{|R|}))$ be a reference annotation and let $Q = (Q_0, \ldots, Q_{|L-1|})$ by a query annotation expressed as a binary sequence. A test statistic A is called* separable *if there exists a function A_v accepting an arbitrary number of binary arguments such that we can express A as*

$$A(R, Q) = \sum_{i=1}^{|R|} A_v(Q_{b_i}, \ldots, Q_{e_i-1}),$$

We will refer to A_v as to the interval function *of A and use it to define a separable test statistic.*

Note that both the number of overlaps and the number of shared bases are separable statistics with interval functions $K_v(s_1, \ldots, s_k) = 1 - \prod_{i=1}^{k}(1 - s_i)$ and $B_v(s_1, \ldots, s_k) = \sum_{i=1}^{k} s_i$, respectively.

Next, we introduce notation expressing conditional state probabilities in the context-aware Markov chain with state sequence $S_{-1}, S_0, \ldots S_{|L|-1}$. Let $x, y \in \{0, 1\}$ and let $-1 \leq i \leq j < L$. We define

$$\Psi(x \overset{i,j}{\rightarrow} y) := \Pr[S_j = y \mid S_i = x]$$

$$\Psi_0(x \overset{i,j}{\rightarrow} y) := \begin{cases} \Pr[S_j = y, S_{i+1} = S_{i+2} = \ldots = S_j = 0 \mid S_i = x] & \text{for } i < j \\ 1 & \text{if } i = j \wedge x = y \\ 0 & \text{otherwise} \end{cases}$$

Note that Ψ was already introduced in Definition 3 in the main text, but is repeated here for completeness. In addition to two-sided plumbuses from Definition 3, we will also compute one-sided plumbuses, which condition the mean and variance only on the state at the left side of the interval, as defined below.

Definition 5. *Let $(\phi, \mathbf{T})$ be a context-aware Markov chain with state sequence $S_{-1}, S_0, \ldots S_{|L|-1}$, let $[i, j)$ be a subinterval of $[0, L)$, and let v be a function on a binary sequence of length $j - i$. We define the* left-sided plumbus *for $[i, j)$ as the collection of values*

$$\mu_v(i, j \mid x) = \mathrm{E}[v(S_i, \ldots, S_{j-1}) \mid S_{i-1} = x]$$
$$\sigma_v^2(i, j \mid x) = \mathrm{Var}[v(S_i, \ldots, S_{j-1}) \mid S_{i-1} = x]$$
$$\Psi(x \overset{i-1,j-1}{\rightarrow} y)$$

for all combinations of (x, y) in $\{0, 1\}^2$.

The next lemma expresses the law of total expectation and the law of total variance (also known as Adam's law and Eve's law) in the form useful for our context, where we work with binary state variables. The lemma will be used repeatedly to compute and combine plumbuses.

Lemma 1. *Given a random variable Y and a discrete random variable X with support $\{0, 1\}$, the following holds:*

$$\begin{aligned}
\mathrm{E}[Y] &= \sum_{x=0}^{1} \mathrm{E}[Y \mid X = x] \cdot \Pr[X = x] \\
\mathrm{Var}[Y] &= \sum_{x=0}^{1} \mathrm{Var}[Y \mid X = x] \cdot \Pr[X = x] + \\
&+ \sum_{x=0}^{1} \mathrm{E}^2[Y \mid X = x] \cdot \Pr[X = x] \cdot (1 - \Pr[X = x]) - \\
&- 2 \cdot \prod_{x=0}^{1} \mathrm{E}[Y \mid X = x] \cdot \Pr[X = x].
\end{aligned}$$

Proof. The first equation can be obtained from the law of total expectation:

$$\mathrm{E}[Y] = \mathrm{E}_X[\mathrm{E}_Y[Y \mid X]] = \sum_{x=0}^{1} \mathrm{E}[Y \mid X = x] \cdot \Pr[X = x].$$

The second equation can be obtained from the law of total variance:

$$\begin{aligned}
\mathrm{Var}[Y] =& \mathrm{E}_X[\mathrm{Var}_Y[Y \mid X]] + \mathrm{Var}_X[\mathrm{E}_Y[Y \mid X]] = \\
&= \sum_{x=0}^{1} \mathrm{Var}_Y[Y \mid X = x] \cdot \Pr[X = x] + \mathrm{E}_X[\mathrm{E}_Y^2[Y \mid X]] - \mathrm{E}_X^2[\mathrm{E}_Y[Y \mid X]] = \\
&= \sum_{x=0}^{1} \mathrm{Var}[Y \mid X = x] \cdot \Pr[X = x] + \sum_{x=0}^{1} \mathrm{E}^2[Y \mid X = x] \cdot \Pr[X = x] - \left(\sum_{x=0}^{1} \mathrm{E}[Y \mid X = x] \cdot \Pr[X = x] \right)^2 = \\
&= \sum_{x=0}^{1} \mathrm{Var}[Y \mid X = x] \cdot \Pr[X = x] + \sum_{x=0}^{1} \mathrm{E}^2[Y \mid X = x] \cdot (\Pr[X = x] - \Pr[X = x]^2) - 2 \cdot \prod_{x=0}^{1} \mathrm{E}[Y \mid X = x] \cdot \Pr[X = x].
\end{aligned}$$

□

The following series of lemmas explains crucial parts of the algorithm.

Lemma 2. *Given a context-aware Markov chain $(\phi, \mathbf{T})$, values $\Psi(x \xrightarrow{i,j} y)$ and $\Psi_0(x \xrightarrow{i,j} y)$ can be computed in $O(c)$ time and $O(1)$ additional space, where c is the number of class boundaries between positions i and j, assuming that the class boundary positions are given.*

Proof. Both of those values can be represented using matrix multiplication:

$$\begin{aligned}
\Psi(x \xrightarrow{i,j} y) &:= \left(\begin{pmatrix} 1 - x & x \end{pmatrix} \cdot \prod_{k=i+1}^{j} \mathbf{T}(\phi(k)) \right)_y \\
\Psi_0(x \xrightarrow{i,j} y) &:= \left(\begin{pmatrix} 1 - x & x \end{pmatrix} \cdot \prod_{k=i+1}^{j} M(\mathbf{T}(\phi(k))) \right)_y ,
\end{aligned}$$

where M is a transformation that replaces the last column with zeros, i.e. $M\left(\begin{pmatrix} t_{00} & t_{01} \\ t_{10} & t_{11} \end{pmatrix}\right) = \begin{pmatrix} t_{00} & 0 \\ t_{10} & 0 \end{pmatrix}$. Note that $\begin{pmatrix} 1 - x & x \end{pmatrix} = \begin{pmatrix} 1 & 0 \end{pmatrix}$ for $x = 0$ and $\begin{pmatrix} 1 - x & x \end{pmatrix} = \begin{pmatrix} 0 & 1 \end{pmatrix}$ for $x = 1$.

Assume the class boundary positions are $t_1, \ldots, t_c$. We also add $t_0 := i$ and $t_{c+1} := j$ in order to unify the notation. Now, we can group the multiplied transition matrices into a product of exponentiations:

$$\Psi(x \overset{i,j}{\to} y) := \left(\begin{pmatrix} 1-x & x \end{pmatrix} \cdot \prod_{k=1}^{c+1} \mathbf{T}(\phi(t_k - 1))^{t_k - t_{k-1}} \right)_y$$

$$\Psi_0(x \overset{i,j}{\to} y) := \left(\begin{pmatrix} 1-x & x \end{pmatrix} \cdot \prod_{k=1}^{c+1} M(\mathbf{T}(\phi(t_k - 1)))^{t_k - t_{k-1}} \right)_y,$$

Matrices $\mathbf{T}(\phi(k)) = \begin{pmatrix} t_{00} & t_{01} \\ t_{10} & t_{11} \end{pmatrix}$ and $M(\mathbf{T}(\phi(k))) = \begin{pmatrix} t_{00} & 0 \\ t_{10} & 0 \end{pmatrix}$ can be exponentiated to an arbitrary positive integer power a in time $O(1)$ using their diagionalizations [5]:

$$\mathbf{T}(\phi(k))^a = \begin{pmatrix} 1 & \frac{-t_{01}}{t_{10}} \\ 1 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 0 & (t_{00} - t_{10})^a \end{pmatrix} \begin{pmatrix} \frac{t_{10}}{t_{01}+t_{10}} & \frac{t_{01}}{t_{01}+t_{10}} \\ \frac{-t_{10}}{t_{01}+t_{10}} & \frac{t_{10}}{t_{01}+t_{10}} \end{pmatrix}$$

$$M(\mathbf{T}(\phi(k)))^a = \begin{pmatrix} {t_{00}}^a & 0 \\ t_{10}{t_{00}}^{a-1} & 0 \end{pmatrix},$$

which concludes the proof of time and space complexity of the algorithm. □

Lemma 3. *(Conversion from two-sided to a left-sided plumbus) Given a two-sided plumbus, it is possible to compute the left-sided plumbus for the same function on the same interval in time $O(1)$.*

Proof. Let the collection of values $\mu_v(i, j \mid x, y)$, $\sigma_v^2(i, j \mid x, y)$ and $\Psi(x \overset{i-1,j-1}{\to} y)$ for all combinations of (x, y) in $\{0, 1\}^2$ be a two-sided plumbus for function v on interval $[i, j)$.

Transition probabilities in the corresponding left-sided plumbus are the same as in the two-sided plumbus, so we only need to compute values $\mu_v(i, j \mid x)$ and $\sigma_v^2(i, j \mid x)$ for all values of x in $\{0, 1\}$. We can compute them in constant time by marginalizing variable y using values contained in the original two-sided plumbus (using Lemma 1):

$$\mu_v(i, j \mid x) = \sum_{y=0}^{1} \mu_v(i, j \mid x, y) \cdot \Psi(x \overset{i-1,j-1}{\to} y)$$

$$\begin{aligned} \sigma_v^2(i, j \mid x) = &\sum_{y=0}^{1} \sigma_v^2(i, j \mid x, y) \cdot \Psi(x \overset{i-1,j-1}{\to} y) + \\ &+ \sum_{y=0}^{1} \mu_v(i, j \mid x, y)^2 \cdot \Psi(x \overset{i-1,j-1}{\to} y) \cdot (1 - \Psi(x \overset{i-1,j-1}{\to} y)) - \\ &- 2 \cdot \prod_{y=0}^{1} \mu_v(i, j \mid x, y) \cdot \Psi(x \overset{i-1,j-1}{\to} y) \end{aligned}$$

□

Lemma 4. *(Merging lemma for plumbuses) Given two two-sided plumbuses for functions f and g on intervals $[i, j)$ and $[j, k)$, respectively, it is possible to compute the two-sided plumbus for function $h(S_i, \ldots, S_{k-1}) = f(S_i, \ldots, S_{j-1}) + g(S_j, \ldots, S_{k-1})$ on interval $[i, k)$ in time $O(1)$.*

Proof. We start by computing $\zeta(x, z, y) = \Pr[S_{j-1} = z \mid S_{i-1} = x, S_{k-1} = y] = \dfrac{\Psi(x \overset{i-1,j-1}{\to} z) \cdot \Psi(z \overset{j-1,k-1}{\to} y)}{\Psi(x \overset{i-1,k-1}{\to} y)}$. The required values of the two-sided plumbus for h on $[i, k)$ can be then computed in $O(1)$ operations from the provided

values using Lemma 1 as follows:

$$\Psi(x \overset{i-1,k-1}{\rightarrow} y) = \sum_{z=0}^{1} \Psi(x \overset{i-1,j-1}{\rightarrow} z) \cdot \Psi(z \overset{j-1,k-1}{\rightarrow} y)$$

$$\mu_h(i,k \mid x,y) = \sum_{z=0}^{1} (\mu_f(i,j \mid x,z) + \mu_g(j,k \mid z,y)) \cdot \zeta(x,z,y)$$

$$\begin{aligned}\sigma_h^2(i,k \mid x,y) = &\sum_{z=0}^{1} (\sigma_f^2(i,j \mid x,z) + \sigma_g^2(j,k \mid z,y)) \cdot \zeta(x,z,y) + \\ &+ \sum_{z=0}^{1} (\mu_f(i,j \mid x,z) + \mu_g(j,k \mid z,y))^2 \cdot \zeta(x,z,y) \cdot (1 - \zeta(x,z,y)) - \\ &- 2 \cdot \prod_{z=0}^{1} (\mu_f(i,j \mid x,z) + \mu_g(j,k \mid z,y)) \cdot \zeta(x,z,y).\end{aligned}$$

Note that this derivation uses the fact that thanks to Markov property, we have

$$\mu_f(i,j \mid x,z) = \mathrm{E}[f(S_i,\ldots,S_{j-1}) \mid S_{i-1} = x, S_{j-1} = z] = \mathrm{E}[f(S_i,\ldots,S_{j-1}) \mid S_{i-1} = x, S_{j-1} = z, S_{k-1} = y],$$
$$\mu_g(j,k \mid z,y) = \mathrm{E}[g(S_j,\ldots,S_{k-1}) \mid S_{j-1} = z, S_{k-1} = y] = \mathrm{E}[g(S_j,\ldots,S_{k-1}) \mid S_{i-1} = x, S_{j-1} = z, S_{k-1} = y],$$

and similarly for the variances. □

Lemma 5. *(Folding lemma for plumbuses) Given a two-sided plumbus for function f on interval $[i,j)$ and a left-sided plumbus for function g on interval $[j,k)$, it is possible to compute the left-sided plumbus for function $h(S_i,\ldots,S_{k-1}) = f(S_i,\ldots,S_{j-1}) + g(S_j,\ldots,S_{k-1})$ on interval $[i,k)$ in time $O(1)$.*

Proof. Let $\zeta(x,z) := \Pr[S_{j-1} = z \mid S_{i-1} = x] = \Psi(x \overset{i-1,j-1}{\rightarrow} z)$. Note that this value is provided by the two-sided plumbus for function f on interval $[i,j)$. The required values of the left-sided plumbus for h on $[i,k)$ can be then computed using $O(1)$ operations from the provided values as follows (using Lemma 1):

$$\Psi(x \overset{i-1,k-1}{\rightarrow} y) = \sum_{z=0}^{1} \Psi(x \overset{i-1,j-1}{\rightarrow} z) \cdot \Psi(z \overset{j-1,k-1}{\rightarrow} y)$$

$$\mu_h(i,k \mid x) = \sum_{z=0}^{1} (\mu_f(i,j \mid x,z) + \mu_g(j,k \mid z)) \cdot \zeta(x,z)$$

$$\begin{aligned}\sigma_h^2(i,k \mid x) = &\sum_{z=0}^{1} (\sigma_f^2(i,j \mid x,z) + \sigma_g^2(j,k \mid z)) \cdot \zeta(x,z) + \\ &+ \sum_{z=0}^{1} (\mu_f(i,j \mid x,z) + \mu_g(j,k \mid z))^2 \cdot \zeta(x,z) \cdot (1 - \zeta(x,z)) - \\ &- 2 \cdot \prod_{z=0}^{1} (\mu_f(i,j \mid x,z) + \mu_g(j,k \mid z)) \cdot \zeta(x,z),\end{aligned}$$

□

The next two theorems show the details of efficient plumbus computation for the two statistics $K(R,Q)$ and $B(R,Q)$ considered in this paper, as required in Theorem 4.

Theorem 2. *For given reference annotation R and context-aware Markov chain $(\phi, \mathbf{T})$, the two-sided plumbuses for interval function v of the number of overlaps statistic K on all intervals in R are computable in time $O(|R| + c)$ and additional space $O(|R|)$.*

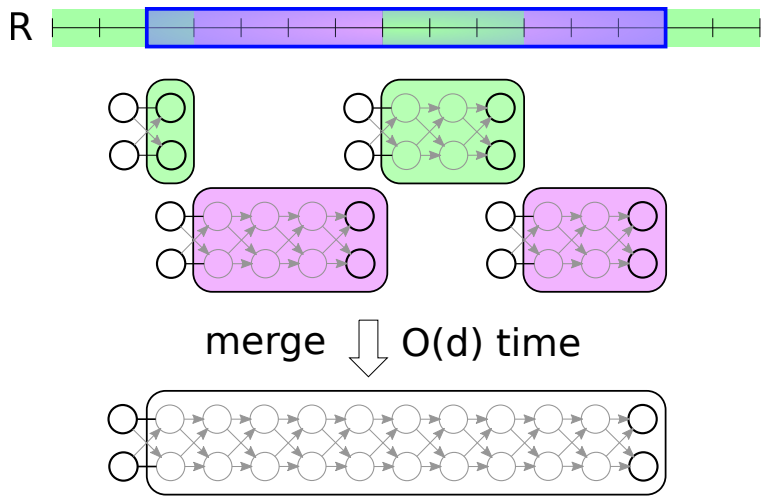

Figure S1: Example of decomposition of a single reference interval into single-class two-sided plumbuses. Note that the individual single-class two-sided plumbuses for the number of shared bases statistics $B(\cdot,\cdot)$ can be computed in time logarithmic to the length of their corresponding genomic interval and then merged into a single plumbus for the whole reference interval in time linear w.r.t. their count using the merging lemma (see the proof of Theorem 3).

Proof. Given interval $[b_j, e_j)$ from R, random variable $v(S_{b_j}, \ldots, S_{e_j-1})$ is an indicator variable, so its conditional mean and variance can be computed as $\mu_v(b_j, e_j \mid x, y) = 1 - q(b_j, e_j \mid x, y)$ and $\sigma_v^2(b_j, e_j \mid x, y) = q(b_j, e_j \mid x, y) \cdot (1 - q(b_j, e_j \mid x, y))$, respectively, where $q(b_j, e_j \mid x, y) := \Pr[v(S_{b_j}, \ldots, S_{e_j-1}) = 0 \mid S_{b_j-1} = x, S_{e_j} = y]$. Value $q(b_j, e_j \mid x, y)$ can be computed as follows:

$$
\begin{aligned}
q(b_j, e_j \mid x, y) &= \Pr[S_{b_j} = S_{b_j+1} = \ldots = S_{e_j-1} = 0 \mid S_{b_j-1} = x, S_{e_j-1} = y] = \\
&= \frac{\Pr[S_{b_j} = S_{b_j+1} = \ldots = S_{e_j-1} = 0, S_{e_j-1} = y \mid S_{b_j-1} = x]}{\Pr[S_{e_j-1} = y \mid S_{b_j-1} = x]} = \\
&= \frac{\Psi_0(x \overset{b_j-1, e_j-1}{\rightarrow} y)}{\Psi(x \overset{b_j-1, e_j-1}{\rightarrow} y)}.
\end{aligned}
$$

Assuming the class boundary positions are available sorted, and the reference intervals are sorted as well, it is then possible to compute values $q(b_j, e_j \mid x, y)$ for all combinations of the input arguments in time $O(|R| + c)$ and space $O(|R|)$, using fast computation of Ψ and Ψ_0 values provided by Lemma 2. Since this computation already includes the computation of transition probabilities $\Psi(x \overset{b_j-1, e_j-1}{\rightarrow} y)$ for each combination of values x, y, j, all the values required for the plumbuses are computable in the required time and space. □

Theorem 3. *For a given reference annotation R and context-aware Markov chain $(\phi, \mathbf{T})$, the two-sided plumbuses for interval function v of the number of shared bases statistic B on all intervals in R are computable in time $O((|R|+c) \cdot \log t)$ and additional space $O(|R|)$, where t is the length of the longest stretch of positions within a single reference interval with the same context class.*

Proof. Let $t_1, \ldots, t_d$ be the class boundaries inside the j-th reference interval $[b_j, e_j)$. Let us also define $t_0 := b_j$ and $t_{d+1} := e_j$ in order to unify the computations. We will split interval $[b_j, e_j)$ into $d+1$ intervals $[t_0, t_1), [t_1, t_2), \ldots, [t_d, t_{d+1})$ with a single context class within each individual interval. Once we compute the two-sided plumbuses for each interval $[t_i, t_{i+1})$ (see Figure S1), we can merge them using the merging lemma (Lemma 4) into a two-sided plumbus on the whole j-th reference interval in time $O(d)$.

Let us now describe computation of the two-sided plumbus for a single-class interval $[t_i, t_{i+1})$ with class label λ. We can split the interval into $t_{i+1} - t_i$ intervals of length 1 $[t_i, t_i + 1), [t_i + 1, t_i + 2), \ldots, [t_{i+1} - 1, t_{i+1})$. The plumbuses

on all those intervals are the same and can be computed in $O(1)$ time:

$$\begin{aligned}\Psi(x \overset{t_i+k, t_i+k+1}{\rightarrow} y) =& \mathbf{T}(\lambda)_{x,y}\\ \mu_v(t_i+k, t_i+k+1 \mid x, y) =& y\\ \sigma_v^2(t_i+k, t_i+k+1 \mid x, y) =& 0.\end{aligned}$$

Similarly, two-sided plumbus for any subinterval of $[t_i, t_{i+1})$ of length k depends only on the length, not on its specific position. We can therefore use the merging operation from Lemma 4 to compute two sided plumbuses for intervals of lengths that are powers of two and to combine these to the two-sided plumbus for the whole interval $[t_i, t_{i+1})$ in time $O(\log(t_{i+1} - t_i))$ and additional space $O(1)$.

Thus, we can compute two-sided plumbus on j-th reference interval in time $O((d+1) \cdot \log t)$ and additional space $O(1)$. Applying this to all reference intervals, we would obtain the desired time and space complexity of the algorithm. □

We can use our lemmas to give a general algorithm for computing mean and variance of any separable statistic.

Theorem 4. *For any separable statistic A, reference annotation R and context-aware Markov chain $(\phi, \mathbf{T})$, it is possible to compute mean $\mathrm{E}_{Q' \sim C(\phi,\mathbf{T})}[A(R, Q')]$ and variance $\mathrm{Var}_{Q' \sim C(\phi,\mathbf{T})}[A(R, Q')]$ in time $O(|R| + c + d)$ and space $O(|R| + c + e)$, where c is the number of class boundaries in context ϕ, and $O(d)$ and $O(e)$ are the time and space complexity of computing two-sided plumbuses for the separable statistic A on each interval in the reference annotation R.*

Proof. We are provided two-side plumbuses for all reference intervals. To handle the gaps between these intervals, we will use zero interval function $z(s_1, \ldots, s_k) = 0$. Using this function, we can obtain auxiliary two-sided plumbuses for all gaps in the reference annotation R including the potential gap between the start of the genome and the first reference interval. This works in total time $O(|R| + c)$ and space $O(|R|)$, since conditional means and variances are all equal to zero, and we only need to compute the transition probabilities using Lemma 2.

Then, we can convert the two-sided plumbus of the right-most reference interval into a left-sided plumbus in time $O(1)$ (Lemma 3). Starting from that left-sided plumbus, we can then fold all interval and auxiliary plumbuses in time $O(|R|)$ (Lemma 5). This would leave us with a left-sided plumbus for the test statistic A on interval $[0, e_{|R|})$.

We can then compute the final unconditional mean and variance of the test statistic A by marginalizing the conditions in the total left-sided plumbus:

$$\begin{aligned}\mathrm{E}_{Q' \sim C(\phi,\mathbf{T})}[A(R, Q')] &= \sum_{x=0}^{1} \mu_A(0, e_{|R|} \mid x) \cdot \vec{\pi}_x\\ \mathrm{Var}_{Q' \sim C(\phi,\mathbf{T})}[A(R, Q')] &= \sum_{x=0}^{1} \sigma_A^2(0, e_{|R|} \mid x) \cdot \vec{\pi}_x + \sum_{x=0}^{1} \mu_A(0, e_{|R|} \mid x)^2 \cdot \vec{\pi}_x \cdot (1 - \vec{\pi}_x) - 2 \cdot \prod_{x=0}^{1} \mu_A(0, e_{|R|} \mid x) \cdot \vec{\pi}_x,\end{aligned}$$

where $\vec{\pi}$ is the initial state distribution of context-aware Markov chain $(\phi, \mathbf{T})$.

Adding up the total time and space complexity with the time and space requirements to compute the plumbuses for the reference intervals, we obtain the time and space complexities as given in the theorem. □

Finally, we are able to prove Theorem 1 from the main text.

Proof of Theorem 1. The statement of theorem follows immediately by plugging in the times in Theorems 2 and 3 into Theorem 4. □

S2 The normal distribution yields an accurate p-value approximation for the number of shared bases statistic B

We have conducted experiments on synthetic data to assess the accuracy of our normal approximation of the p-value for the number of shared bases B statistic in comparison with the full PMF computed by an exact algorithm. To address the fact that the original MCDP algorithm does not work for this statistic, we split each reference interval into a sequence

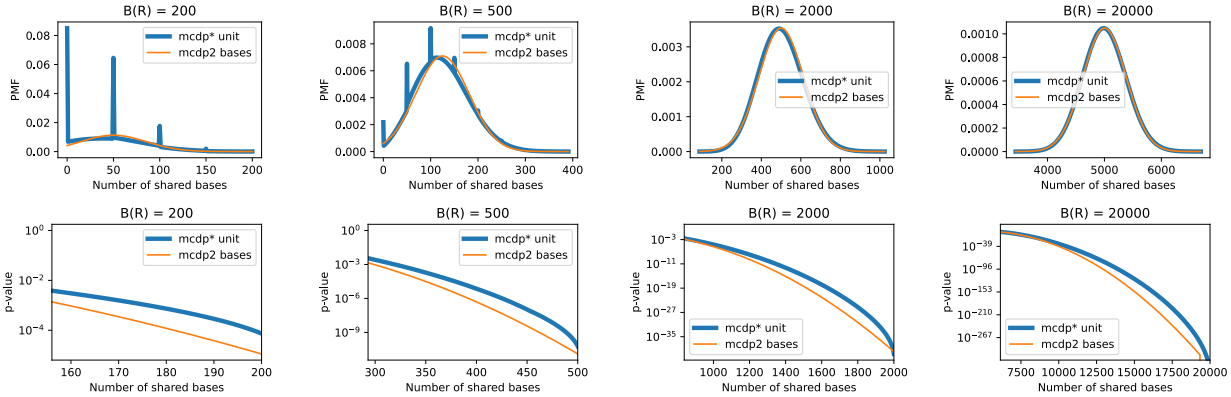

Figure S2: The comparison of the exact probability mass functions (PMFs) for the number of shared bases statistic B with its normal approximation on synthetic data sets. Each column represents a different total number of bases in reference intervals ($B(R) \in \{200, 500, 2000, 20000\}$). The exact PMF (labeled "mcdp* unit") was computed by splitting all reference intervals into intervals of size one and then applying the exact MCDP* algorithm for the number of overlaps statistics K. The normal approximation (labeled "mcdp2 bases") is computed using our new MCDP2 algorithm for the number of shared bases statistic B. The bottom plots show differences for the extreme tail of the distribution. The curves represent the exact and approximated p-value for different values of the statistic, starting from the position with Z-score +3.

of adjacent intervals of size one, producing a modified reference annotation R'. For example, reference annotation $R = \{[2,5),[7,9)\}$ would yield $R' = \{[2,3),[3,4),[4,5),[7,8),[8,9)\}$. It is easy to see that $B(R,Q) = B(R',Q) = K(R',Q)$, and we can therefore compute the exact PMF for the $B(R,Q)$ by computing the PMF for the $K(R',Q)$ statistic by the MCDP* algorithm. The original MCDP algorithm assumes that adjacent intervals are separated by at least one base, which is not true for R', but the algorithms can be easily adapted also for adjacent intervals, which was done in MCDP*. A major limitation is that the running time of MCDP is quadratic with respect to the number of reference intervals $|R|$. In our case, this turns into a quadratic dependence on the total number of *bases* in reference intervals, which we denote $B(R)$. This limits our experiments to reference annotations with up to tens of thousands of bases.

The comparison was performed on synthetic data sets similar to those used for the number of overlaps K, adjusting for the limitation on the total number of bases in the reference. The synthetic data sets were generated with genome length $L = 10^5$ bp, query annotations with 500 randomly generated intervals of length 50 bp each. The reference annotations were generated with up to 400 intervals of length 50 bp each. To understand the influence of the total number of reference bases $B(R)$ on the accuracy, we vary $B(R)$ from 200 to 20 000, i.e. from 4 to 400 reference intervals.

Figure S2 shows the rapid convergence of the exact PMF to its normal approximation. The cases with four and ten reference intervals (the first and second columns of the figure) show prominent peaks in the exact PMF, corresponding to probabilities of hitting zero, one, two or more complete reference intervals. The periodicity of the peaks is equal to the length of the reference intervals (50 bp). Nonetheless, forty intervals (the third column) are in this case enough for the PMF to be very close to normal around the center of the distribution. However, even with 400 intervals the PMFs still differ significantly in the extreme tails. We expect that approximation of even these extreme values will be improved with larger reference annotations which could not be tested due to computational costs of the full PMF computation. The largest annotation tested here contains only 400 intervals; in practice we usually work with much larger annotations.

S3 Details of overlaps between all exons and copy number losses

This section provides a more detailed analysis of overlaps between all exons (reference annotation R) and copy number losses (query annotation Q) in the model with six context class labels. The MCDP analysis without contexts as well as the original analysis by Zarrei et al. [22] indicates enrichment, adding gaps turns the MCDP2 result into a slight depletion and adding five GC content class labels plus gaps leads to an even more pronounced depletion (Figure 5,

label "All genes"). Detailed statistics are shown in Table S1. The five GC labels were chosen so that each corresponds to roughly the same amount of genomic sequence. The number of exon bases is similar in the first four GC classes, but it is much higher in the class with the highest GC. Copy number losses are also most frequent in high GC regions, but the difference is less pronounced. For each context class label, we also calculated the raw fold increase defined as the ratio between the observed and expected number of bases shared between both R and Q, where the expected number of bases is computed under a simplified model with independent bases. For example, for a context class with 1 000 000 bases in total, 300 000 reference bases and 200 000 query bases, the expected number of shared bases is equal to $1\,000\,000 \cdot \frac{300\,000}{1\,000\,000} \cdot \frac{200\,000}{1\,000\,000} = 60\,000$. If the observed number of shared bases in that context class is 100 000, then the raw fold increase is equal to $\frac{100\,000}{60\,000} \approx 1.67$. Our results show that even after taking into account different density of losses and exons in different GC contexts, the overlaps are depleted in all GC contexts, with the strongest 0.75-fold depletion in the lowest GC level and much weaker 0.92-0.97-fold depletion in the highest three GC content levels.

Table S1: Detailed statistics for reference annotation R consisting of exons of all genes and query annotation Q consisting of copy number losses in the model with six context classes. For each context class, we show the number of bases from the whole genome, R, Q and $R \cap Q$ as well as the raw fold increase. We also show what percentage of bases in a particular set is in each context class.

Class label (GC level)	1	2	3	4	5	gap
All bases	586,333,909	578,608,837	583,483,606	570,416,860	578,813,841	239,504,211
All bases %	18.69%	18.44%	18.60%	18.18%	18.45%	7.63%
R (exons) bp	11,412,329	10,236,623	10,313,452	11,577,984	33,390,840	3
R (exons) % bp	14.83%	13.31%	13.41%	15.05%	43.40%	0.00%
Q (losses) bp	38,521,828	38,854,411	39,613,087	42,156,278	56,042,336	2
Q (losses) % bp	17.90%	18.06%	18.41%	19.59%	26.04%	0.00%
R and Q overlap bp	561,989	576,403	646,773	829,132	2,988,902	0
R and Q overlap % bp	10.03%	10.29%	11.54%	14.80%	53.34%	0.00%
Raw fold increase	0.75	0.84	0.92	0.97	0.92	0.00