SpacerPlacer: Ancestral reconstruction of CRISPR arrays reveals the evolutionary dynamics of spacer deletions

Axel Fehrenbach[1,2,*], **Alexander Mitrofanov**[3,*], **Omer S. Alkhnbashi**[4,5], **Rolf Backofen**[3,6,✉], **and Franz Baumdicker**[1,2,✉]

[1]Cluster of Excellence "Controlling Microbes to Infections", Mathematical and Computational Population Genetics, University of Tübingen, Tübingen, Germany
[2]Institute for Bioinformatics and Medical Informatics (IBMI), University of Tübingen, Tübingen, Germany
[3]Bioinformatics group, Department of Computer Science, University of Freiburg, Freiburg, Germany
[4]Information and Computer Science Department, King Fahd University of Petroleum and Minerals (KFUPM), Dhahran 31261, Saudi Arabia
[5]Interdisciplinary Research Center for Intelligent Secure Systems (IRC-ISS), King Fahd University of Petroleum and Minerals (KFUPM), Dhahran 31261, Saudi Arabia
[6]Signalling Research Centres BIOSS and CIBSS, University of Freiburg, Freiburg, Germany
[*]shared first authorship

Bacteria employ CRISPR-Cas systems for defense by integrating invader-derived sequences, termed spacers, into the CRISPR array, which constitutes an immunity memory. While spacer deletions occur randomly across the array, newly acquired spacers are predominantly integrated at the leader end. Consequently, spacer arrays can be used to derive the chronology of spacer insertions.

Reconstruction of ancestral spacer acquisitions and deletions could help unravel the coevolution of phages and bacteria, the evolutionary dynamics in microbiomes, or to track pathogens. However, standard reconstruction methods produce misleading results by overlooking insertion order and joint deletions of spacers.

Here, we present SpacerPlacer, a maximum likelihood-based ancestral reconstruction approach for CRISPR array evolution. We used SpacerPlacer to reconstruct and investigate ancestral deletion events of 4565 CRISPR arrays, revealing that spacer deletions occur 374 times more frequently than mutations and are regularly deleted jointly, with an average of 2.7 spacers.

Surprisingly, we observed a decrease of the spacer deletion frequency towards *both* ends of the reconstructed arrays. While the resulting trailer-end conservation is commonly observed, a reduced deletion frequency is now also detectable towards the variable leader end. Finally, our results point to the hypothesis that frequent loss of recently acquired spacers may provide a selective advantage.

CRISPR-Cas | bacterial genomics | spacer array | ancestral reconstruction | spacer gain and loss | likelihood functions | microbiomes | CRISPR typing

Correspondence: *franz.baumdicker@uni-tuebingen.de, backofen@cs.uni-freiburg.de*

Introduction

Bacteria and Archaea are frequently exposed to phage and plasmid invasions. As a countermeasure, an enormous number of immune mechanisms have evolved in prokaryotes [1, 2]. A very prominent system is the adaptive defense system CRISPR-Cas, Clustered Regularly Interspaced Repeats (CRISPR) with CRISPR-associated genes (Cas), which provides immunity against phages, plasmids, and other mobile genetic elements [3, 4]. A variety of CRISPR-Cas systems have been found and categorized into two classes with a multitude of types and subtypes [5, 6]. While the mechanisms of interference are highly subtype specific and depend on the specific set of Cas proteins involved, many of the Cas proteins responsible for adaptation (e.g. Cas1 and Cas2) are widespread and more conserved. Nevertheless, subtype-specific differences in adaptation exist.

CRISPR spacer array evolution The adaptation process, i.e. the acquisition of spacers, which are small snippets of DNA that correspond to foreign genetic elements, is an essential part of most CRISPR systems. Spacer sequences are inserted into an array separated by repeats [4, 7]. The matching foreign sequences, called protospacers, are subsequently recognized and targeted by the CRISPR system. Insertion of new spacers is almost always facilitated at one end of a CRISPR array [7, 8]. Thus, the order of spacers in a CRISPR array provides a timeline of the phages and plasmids encountered, identified by the corresponding protospacer sequences.

As part of the genome, acquired CRISPR spacers are inherited across generations, but can also be deleted [9]. While in adaptation, e.g. factors like Cas1, Cas2 and the leader sequence are well studied, the source for deletions is not yet fully understood. Hypotheses include that spacer deletion occurs by misalignment between repeats during replication [10, 11], or that spacer deletion occurs with the addition of a new spacer [12].

On the positive side, spacer acquisition and deletion are frequent and thus constantly modify the CRISPR arrays within the population, resulting in observable short-term dynamics formed by the co-evolution with foreign genetic elements. The ordered timeline of spacer acquisitions not only constrains the population histories but also links to the targeted co-evolving phages and plasmids. Finally, the arrays are relatively short in length and can be reliably identified due to their repeat structure [13–16].

In principle, CRISPR arrays could thus be particularly useful for investigating evolutionary relationships at high resolution, tracking and typing strains in epidemiological outbreaks, understanding the dynamic ecology of microbiomes, and analyzing biotechnological and pharmaceutical biocultures [16–23]. However, a reconstruction tool, as presented here, is needed to correctly determine ancestral spacer insertion and deletion events.

CRISPR spacer array evolution challenges traditional phylogenetic reconstruction methods A tool for reconstructing bacterial evolution from CRISPR arrays faces the problem that the number of spacers in an array is small, and that polarized acquisitions and blockwise deletions generate dependencies between different evolutionary spacer events. This makes it difficult to use standard phylogenetic methods for DNA sequences, which neglect the specifics of CRISPR evolution, resulting in avoidable errors.

While a variety of visualization tools for CRISPR spacer arrays have been developed [24–27], reconstruction methods that take into account the specifics of CRISPR array evolution are still rare. An exception is the CRISPR Comparison Toolkit (CCTK) introduced by Collins and Whitaker [27], which contains a maximum parsimony approach to reconstruct ancestral spacer arrays by first aligning the arrays using the Needleman-Wunsch algorithm, and then inferring ancestral arrays using empirically determined parsimony costs. However, to go beyond ancestral reconstruction, it is crucial to employ explicit evolutionary models. These models incorporate interpretable parameters, allowing for a more comprehensive analysis of the corresponding molecular mechanisms and evolutionary processes.

***SpacerPlacer* allows to investigate spacer array evolution** Here, we introduce *SpacerPlacer*, to our knowledge, the first parametric likelihood-based approach tailored for the reconstruction of ancestral spacer insertions and deletions. *SpacerPlacer* reconstructs ancestral states while considering the unique features of CRISPR arrays and their polarized insertion order through an evolutionary model. The advantage of using a probabilistic approach is that we can compare different evolutionary models for CRISPR spacer arrays and investigate the corresponding molecular mechanisms of spacer acquisitions and deletions, as well as selection pressures, in more detail. One major benefit of this model-based approach is that it - in addition to the reconstruction - allows to estimate the underlying evolutionary parameters. This enables us to infer the distribution of the number of jointly lost adjacent spacers, detect potential differences between CRISPR-Cas types, and test model assumptions using likelihood ratio tests.

We applied *SpacerPlacer* on all suitable high-confidence CRISPR arrays from the CRISPRCasdb database [28]. Despite the variability of subtypes, our analysis did not find significant differences in deletion rates and lengths between the subtypes considered here, which points to a Cas-independent process for deletion. We find that the median deletion rate per spacer is 374 times higher than the mutation rate per site. Since spacers and their repeats are fairly short [11, 29], we conclude that the evolution of CRISPR arrays by acquisitions and deletions is much faster than the mutation of the spacer sequences themselves.

We then investigate the reconstructed spacer deletions in more detail. To this end, we performed a likelihood ratio test and found that spacers are indeed regularly jointly deleted. Using our probabilistic framework, we estimate the mean length of these joint deletions as 2.7 adjacent spacers.

We also investigated how the distribution of the deletion frequency changes with respect to the position within the array. We find that deletion frequencies are decreased towards both boundaries of the array. This boundary effect is consistent with the conjecture that spacer deletions are mainly caused by repeat misalignment.

Moreover, by comparing empirical and simulated deletion frequencies, we find indications that either selection increases the deletion frequency of the first spacer, or mutations in the last repeat reduce the deletion frequency of the last spacer, or both.

Methods

Spacer array data preparation. To run *SpacerPlacer* we require a group G of completely sequenced related CRISPR arrays with consecutively ordered spacers and a respective phylogenetic tree $\mathcal{T}$.

Ideally, this tree should be provided by the user, but *SpacerPlacer* provides a heuristic method to infer a suitable tree $\mathcal{T}$ from the spacer arrays itself.

Each unique spacer DNA sequence in G is labeled with an individual natural number. A simple example of a group G is shown in Figure 4 a). Obviously, the arrays in G should share some spacers but not all be identical. So, we restricted the analysis to such groups.

Database. CRISPRCasdb is a freely accessible database that contains the CRISPR array predictions of "CRISPRCas-Finder" [30] on whole genome assemblies collected from GenBank. For the construction of the spacer database, we extracted spacer sequences from CRISPRCasdb [28] in July 2022. We only consider predictions of the highest evidence level (4), resulting in a total of 7352 repeats and 32624 arrays.

Clustering spacers based on sequence homology. After extraction, we clustered the spacers into sets that we treat as identical, and have likely undergone only mutations. Each spacer in the CRISPR arrays was relabeled according to its respective cluster label. The relabeled CRISPR arrays serve as input for *SpacerPlacer*.

Clustering was performed individually for each identified consensus repeat. The procedure encompassed several stages: Initially, all CRISPR arrays containing the specific consensus repeat were selected. Subsequently, spacers associated with these arrays were aggregated. Hierarchical clustering was then applied to these spacers using the following algorithm: Initially, each spacer was treated as an independent cluster. This was followed by amalgamating sequences with their reverse complementary counterparts into single clusters. In subsequent iterations, the clusters were progressively merged if they contained spacer pairs that differed by a single editing operation, as determined by the Levenshtein distance metric, which includes insertions, deletions, and substitutions. The clustering process was deemed complete when no additional merges were feasible.

Identification of sets of closely related CRISPR arrays. We further subdivided the pool of closely related arrays with the highest confidence level according to their repeat sequence.

For all repeats, we checked if there is overlap with reverse complement repeats and reverse complemented spacer arrays. CRISPR arrays were assigned their Cas type according to the closest Cas-cassette found within the genome. Arrays with no close (< 10000 bp distance) Cas-cassette were removed, unless only one Cas-cassette was found within the genome. Groups were further subdivided by genus and Cas-type.
Finally, the groups were further subdivided by pairwise overlap of the spacers. Each spacer array has to have at least one spacer overlap with any of the other arrays in a group to be included. We excluded groups that had less than three arrays or lacked spacer diversity between arrays.
For further details and a breakdown of the number of groups and arrays within each step see Supplementary Note 4A.
The final data set with core gene trees fed into *SpacerPlacer* contains 518 groups with 5934 arrays in total. Of these, 333 groups (4565 arrays) contain at least one reconstructed deletion. We refer to these groups as the "CRISPRCasdb dataset". A breakdown of the Cas types and genera found in the CRISPRCasdb dataset is shown in Supplementary Figure S12 and S13, respectively.

Tree estimation. The quality of the underlying phylogenetic tree seriously affects both the ancestral spacer reconstruction and the parameter estimates. In general, we suggest using whole genome trees, if available, to reduce the dependency between tree estimation and array reconstruction, especially if rate estimation is desired.
For each group of spacer arrays, we reconstructed a phylogenetic tree, independent of the arrays, based on the single-copy core genes of the corresponding strain. We used Bakta to annotate genomes [31], the pangenome analysis tool panX [32], and IQ-TREE [33] to reconstruct trees. Throughout this work, we used these core genome trees unless otherwise stated.
We did not find evidence indicating that our median estimates were affected by potential biases introduced by using core genome trees for CRISPR arrays where horizontal transfer might have occurred. Furthermore, we removed strains with multiple arrays from our analysis to avoid the inability of core genome-based trees to differentiate between them.
As an alternative for such cases, where no plausible independent tree is available, *SpacerPlacer* offers the option to use a basic UPGMA approach to infer trees directly from CRISPR arrays. For this, we developed a distance function for spacer arrays that takes into account the specifics of CRISPR array evolution, including blockwise deletions and acquisition order. The details of the distance function can be found in Supplementary Note 2.
When strains with multiple spacer arrays are of particular interest, comparing the spacer-array tree with the core phylogeny can identify whether the different spacer arrays are artifacts of the CRISPR array detection process, such as a mistakenly split spacer array, or related coexisting CRISPR arrays [34].
However, we suggest to closely investigate the plausibility of these CRISPR array-based trees, especially if the number or diversity of spacer arrays is low. Also, the calculated per-generation deletion rates are only meaningful when the underlying tree does not depend on the arrays and has an independent time scaling.

(Re-)orientation of spacer arrays. The orientation of spacer arrays was chosen according to a comparison between the likelihood of the "forward" reconstruction of the group, i.e. the orientation supplied by CRISPRCasdb, and the likelihood of the "reverse" reconstruction, where the MSAA of the group was reversed. If no clear distinction, i.e. a large margin between likelihoods was found, the orientation provided by CRISPRCasdb was used.

Simulations. The simulation of spacer acquisition and deletion events along a simulated tree was done similarly to previous approaches [35, 36].
We start at the root of the corresponding phylogeny, with spacer arrays with a random length that is distributed according to the stationary length distribution of the respective evolutionary model. The array is then evolved along the provided tree by sampling insertions and deletion events according to an independent or block deletion model with polarized insertions. The underlying trees were simulated according to the coalescent, if not indicated otherwise. More details can be found in Supplementary Note 4B.

Estimation of block deletion length in experimental data. To test the assumption of geometrically distributed lengths of block deletions we performed a goodness of fit test (Kolmogorov-Smirnov) on the empirical distribution inferred by Lopez-Sanchez et al. [37]. They selected against the individual spacer 258 and experimentally derived the neighboring spacers deleted jointly with this spacer (Figure 3 in [37]). Since the time span of the experiment was very short and only a few (2) concurrent spacer deletions occurred, we assume the deletions of and around spacer 258 to be the result of single events. So we conditioned on a guaranteed deletion of spacer 258 to estimate the expected length of deletions.

Results

A new probabilistic model for CRISPR array evolution.

Underlying probabilistic model. We will introduce different models, all Markov processes along a given (estimated) phylogenetic tree $\mathcal{T}$ (note that we adapt the naming conventions of previous studies [35, 36]). Besides using a provided phylogeny, our approach allows tree estimation solely based on the spacer arrays themselves in cases where there is no independent estimate of the phylogeny available (see Supplementary Note 2).
The state space in our models corresponds to the alignment of all (observed) spacers. At each branch, the spacer array is described by a presence-absence pattern, where each spacer corresponds to a position in the input alignment.
Spacer insertion and deletion events occur along the branches of $\mathcal{T}$ according to a Poisson point process. Thus, each of the models is characterized by rate-of-occurrence parameters.

The main differences between the models are the possible positions along the spacer array where new spacers can be inserted and the specific characteristics of deletion events, as described in the following.

Ordered or unordered insertions. Despite variations in the adaptation proteins Cas1 and Cas2 [5, 6, 11, 38], the polarized insertion remains nearly universal across Cas types, and we leverage this fact to improve the reconstruction in *SpacerPlacer*. Thus, we mainly consider the *ordered insertion model* corresponding to the predominant mode of insertion [7], where a spacer can be inserted only at the leader end. While the ordered insertion model is clearly the more realistic model, we also introduce a simpler *unordered insertion model* to speed up and simplify parts of the guide reconstruction. In this model, new spacers can also be inserted between existing spacers. The different events of both models are shown for an example array in Figure 1. Furthermore, for all models, we assume that all acquired spacers are unique. However, *SpacerPlacer* detects and resolves duplicated and reinserted spacers by alternative means discussed later. For this *SpacerPlacer* renames multiple occurrences of the same spacer and thus guarantees a unique naming of spacers.

Independent or blockwise deletions. In contrast to insertions, so far no active molecular mechanism modulating spacer deletions has been identified. Replication misalignment or strand/replication slippage have been proposed as the main underlying molecular mechanisms that cause spacer deletions between repeats [11]. Misalignment is believed to occur when two repeats erroneously align within a single replication fork [39]. Consequently, this process is constrained to repeats that are relatively close in proximity, which aligns well with CRISPR arrays, where spacers between repeats tend to be rather short. Furthermore, this would also allow for deletions of several adjacent spacers, which we term block deletions.

Thus, to investigate how many spacers are typically affected by a single deletion event, we contrast two different deletion models. In the *independent deletion model (IDM)* each individual spacer s_i (of some array $s = (s_1, s_2, \ldots, s_n)$ of length n) can only be independently deleted with rate ρ_{I}. In the *block deletion model (BDM)*, deletion events also arise independently at any spacer s_i at rate ρ_{B}. However, a consecutive block of L spacers $s_i \ldots s_{i+L-1}$ is jointly deleted, where L is geometrically distributed with mean α. If L exceeds the boundary of the spacer array, the event is ignored. Note that, due to its monotonically decreasing probability mass function, longer deletions become increasingly unlikely and singular deletions remain the most likely. Further, note that setting $\alpha = 1$ yields the independent deletion model, which allows us to use the framework of likelihood ratio tests to distinguish between the IDM and the BDM. Both models are illustrated in Figure 1.

An alternative result of replication misalignment is spacer duplication or rearrangement [40, 41], which has also been observed for spacer-repeat units in CRISPR arrays [11, 39]. Here, we do not directly incorporate such more complex events into the evolutionary models. Instead, we detect and resolve them in a separate step of the reconstruction pipeline.

Comparison to other models of spacer array evolution. Some stochastic models of CRISPR evolution have been specifically designed for spacer array evolution [35, 36], while others explored the coevolutionary dynamics between phages and bacteria [29, 42, 43].

Kupczok and Bollback compared models of spacer deletions that assume independent deletion of single spacers and polarized insertions with models with internal insertions, as well as a model in which fragments of the spacer array could be deleted [35]. In contrast to the block deletions with geometrically distributed length considered here, fragment deletions cover, regardless of its length, any possible fragment of the array with the same probability. Based on the data available at the time, Kupczok and Bollback found that independent deletions, as well as fragment deletions are plausible models. The block deletion model proposed here has two advantages: a) the observed deletion pattern observed in CRISPRCasdb and experiments [37] are in line with block deletions of a geometrically distributed length and b) the memorylessness of the geometric distribution allows efficient likelihood computation for reconstruction and parameter estimation. In the past, the independent deletion model has been used to construct a deletion rate estimator that is based only on the variability at the trailer-end of spacer arrays [36], but did not incorporate the other parts of the array nor ancestral reconstructions.

The length of spacer arrays has also been the subject of multiple modeling approaches [44, 45]. The equilibrium between spacer deletions and acquisitions might lead to an average spacer array length, which is subject to selection [44]. In our setting with no fitness effect of spacers, the number of spacers in an array is Poisson distributed and depends on the ratio of acquisition and per spacer deletion rate.

Collins and Whitaker proposed an alternative to parametric models by introducing a reconstruction tool based on maximum parsimony principles [27]. To accommodate all possible types of events in CRISPR array evolution, they defined empirically determined scores for acquisitions at the leader end, duplications, insertions within the array, deletions (of arbitrary length), and independent acquisitions.

In contrast, we rely on explicit evolutionary models that can offer insight beyond ancestral reconstruction through parameters that correspond to molecular mechanisms and evolutionary processes. We use a dual approach where the underlying molecular mechanisms of CRISPR arrays influenced the modeling choices, but the reconstruction-based parameter estimates also allows investigating the evolutionary (co-)dynamics of CRISPR arrays. For example, only our new framework allows us to introduce the likelihood ratio test that identifies the mode of deletions.

An overview of the models used in related studies, different steps of our pipeline, including our LRT workflow, is shown in Figure 2.

Ancestral CRISPR Spacer Array Reconstruction.

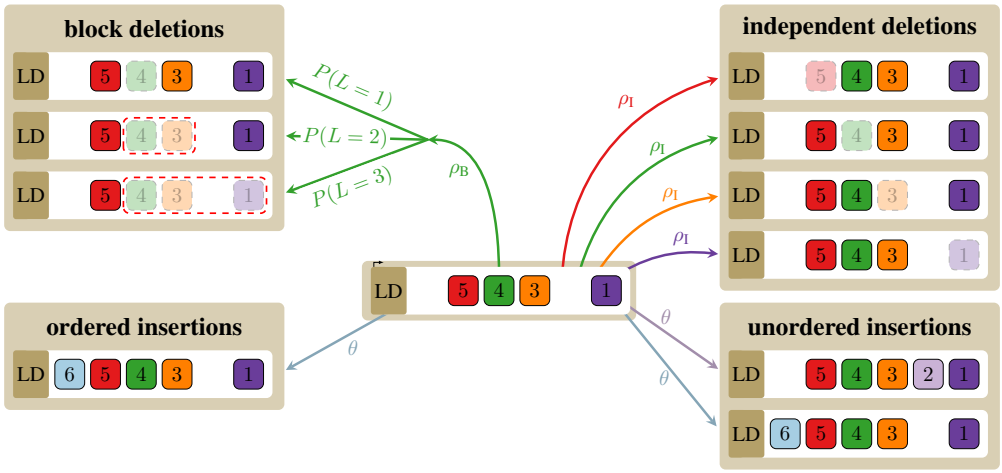


Fig. 1. Different deletion and insertion models. The figure shows how aligned spacer arrays evolve in different models used within *SpacerPlacer*. Each colored and numbered box represents a different spacer. Unordered insertions can occur at any unoccupied position within the array (bottom right). Within the ordered insertion model (bottom left), spacers can only be acquired directly after the leader (indicated by LD) in front of the first existing spacer. In the independent deletion model (top right), each spacer can be deleted individually with rate ρ_I. Block deletions (top left) occur at each (occupied) position at rate ρ_B and $L \geq 1$ spacers are deleted jointly (including the spacer hit by the event), where L is geometrically distributed with mean α. For simplicity, we show only one block deletion event. The color of the arrows indicates at which spacer position an event was initiated. The most realistic model includes ordered insertions and block deletions (left side), while models with independent deletions and unordered insertions (right side) serve as a reference and speed up parts of the reconstruction pipeline.

Overview of the pipeline. *SpacerPlacer* is the first parametric reconstruction and visualization tool for CRISPR spacer array evolution. Based on a set of overlapping spacer arrays and a provided phylogeny, *SpacerPlacer* proceeds in multiple steps to reconstruct ancestral spacer acquisitions, deletions, and more complex events, such as rearrangements. An overview of the reconstruction pipeline is shown in Figure 3. First, a spacer array alignment based on the spacer labels is constructed using a MAFFT script tweaked for this purpose. This alignment of the spacer array reveals duplicated spacers as well as candidates for more complex events. Next, the alignment of the spacer array is used to create a partial spacer insertion order (PSIO) that contains all available relative acquisition-time information.

The dependence between different spacer sites introduced by polarized insertion and block deletions makes computing the joint likelihoods for the reconstructions infeasible. Therefore, we first calculate preliminary ancestral states at the internal nodes of the provided phylogeny based on the less complex independent deletion model with unordered insertions. In the next step, the guide reconstruction is refined so that the reconstructed insertions obey the partial order. This indirect inclusion of the insertion polarization is more efficient than calculations in a complex parametric model. Finally, the reconstructed events of putative duplicated spacers are categorized as duplications, rearrangements, reinsertions, or similar events.

On the basis of such reconstructed insertion and deletion events, *SpacerPlacer* will also estimate the parameters of an IDM and the more complex BDM. Subsequently *SpacerPlacer* identifies the most likely model in a likelihood ratio test (Figure 2) for the data set considered.

SpacerPlacer is available at `https://github.com/fbaumdicker/SpacerPlacer`.

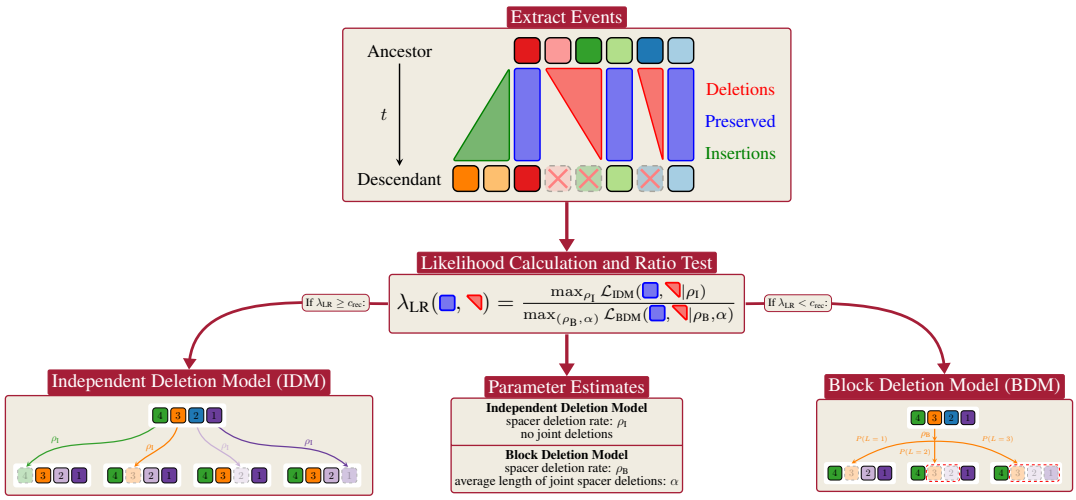
Multiple spacer array alignment The order of the spacers in different arrays and even within a single spacer array can contradict each other under the assumptions of polarized insertions. For instance, if a spacer occurs twice in array A and only once in array B, it is unclear which of the duplicated spacers in array A corresponds to the spacer in array B. Similarly, if spacer 1 follows spacer 2 in array A but spacer 2 follows spacer 1 in array B, this is clearly not realizable through a single acquisition.

To identify and group such candidate spacers, *SpacerPlacer* performs a Multiple Spacer Array Alignment (MSAA) on each group of spacer arrays. We then identify the most plausible matching spacers according to their positions in the MSAA.

Instead of using the sequences of the spacers, we align groups of spacer arrays solely based on the spacer labels with the multiple sequence alignment tool MAFFT [46]. MAFFT can align sequences of arbitrary characters instead of nucleotide and protein sequences. *SpacerPlacer* tweaks the MAFFT parameters and uses low gap penalties and a very high mismatch penalty, to ensure that MAFFT does not produce mismatches in the spacer alignment, favoring gaps instead. More details about this process can be found in Supplementary Note 3.

Partial Spacer Insertion Order graph construction The partial order within the spacer arrays encodes the information that is generated from polarized insertions [7], one of the most consistent properties of CRISPR arrays. This has been empirically confirmed for many Cas types, except for type II-A systems, where ectopic insertions have also been found [47–49].

Each spacer array constitutes a timeline of acquired spacers, i.e., an insertion order for the acquisitions of this array. Thus, by combining the order information of all arrays in a group,



Task	SpacerPlacer	Kupczok et al. [35]	Collins et al. [27]
Phylogeny computation	external or blockwise UPGMA	rooted NJ	external or most parsimonious
Guide Reconstruction	unordered acq. individual del.	—	custom acq. costs custom del. costs
Refined	partially ordered acq. individual del.		
Parameter estimation	ordered acq.* individual del. blockwise del.	ordered acq. individual del. fragment del.	—
Deletion models for LRT	individual vs blockwise	—	—

Fig. 2. Parametric models and LRT workflow in SpacerPlacer Using a parametric model allows *SpacerPlacer* to combine ancestral reconstruction with rate estimation and likelihood ratio tests. After reconstructing the ancestral states with *SpacerPlacer* we extract the deletion events and preserved spacers, as well as the lengths of the tree branches. These events are then used to generate maximum likelihood estimates for the parameters of both deletion models the IDM and the BDM. *SpacerPlacer* also uses a likelihood ratio test to identify whether the IDM can be rejected in favor of the BDM. The table shows a comparison of the spacer deletion and insertion models used here and in related publications [27, 35]. Naturally, the most parsimony approach by Collins and Whitaker [27] does not include rates for different events but custom cost functions for ordered and ectopic acquisitions, duplications, deletions depending on their position and length.
*The parameter estimation and the likelihood ratio testing in *SpacerPlacer* rely solely on reconstructed deletions, thus the acquisition rate can only be estimated indirectly.

we can extract additional timeline information. For example, when given two arrays, A (1-2-4-6) and B (2-3-5-6). Here, array A implies that spacer 1 was acquired after spacer 2 and array B implies spacer 2 after spacer 3. Together, we can conclude that spacer 1 was acquired after spacer 3. Note that when considering multiple spacer arrays, not all spacers need to have an order relation with each other. In the previous example, the arrays do not determine whether spacer 4 was inserted before or after spacer 5. Thus, combining the order information of multiple arrays leads only to a *partial* spacer insertion order.

The PSIO can be visualized as a graph (Figure 4). In this graph, each arrow indicates a "younger than" or equivalently, "was acquired after" relation. We construct the PSIO by thinning out redundant edges from the graph based on all single array insertion orders from the MSAA. The resulting graph is a directed graph without loops as the previous MSAA step already resolved potential conflicts.

Guide reconstruction As a first step of the ancestral spacer array reconstruction we calculate a preliminary *guide* reconstruction.

As the unordered independent deletion model on spacer presence-absence profiles is a *general time reversible (GTR)* model we can use an effective dynamic programming algorithm [50] for our ancestral guide reconstruction. As multiple independent acquisitions of the same spacer are rare, we use parameters that encourage the model to prioritize multiple deletions over multiple insertions in the guide reconstruction.

The algorithm scales linearly with the number of arrays and linearly with the number of sites, in our case, the number of spacers in the MSAA [50]. Our implementation is based on *TreeTime* [51] and outlined in more detail in Supplementary Note 3.

The guide reconstruction cannot take advantage of all available information as it ignores the PSIO. Consequently, it can still contradict polarized acquisition, as illustrated in Figure 4. In particular, the guide reconstruction will still underestimate the number of deletion events. However, it is a suitable starting point for the subsequent refinement steps that address these

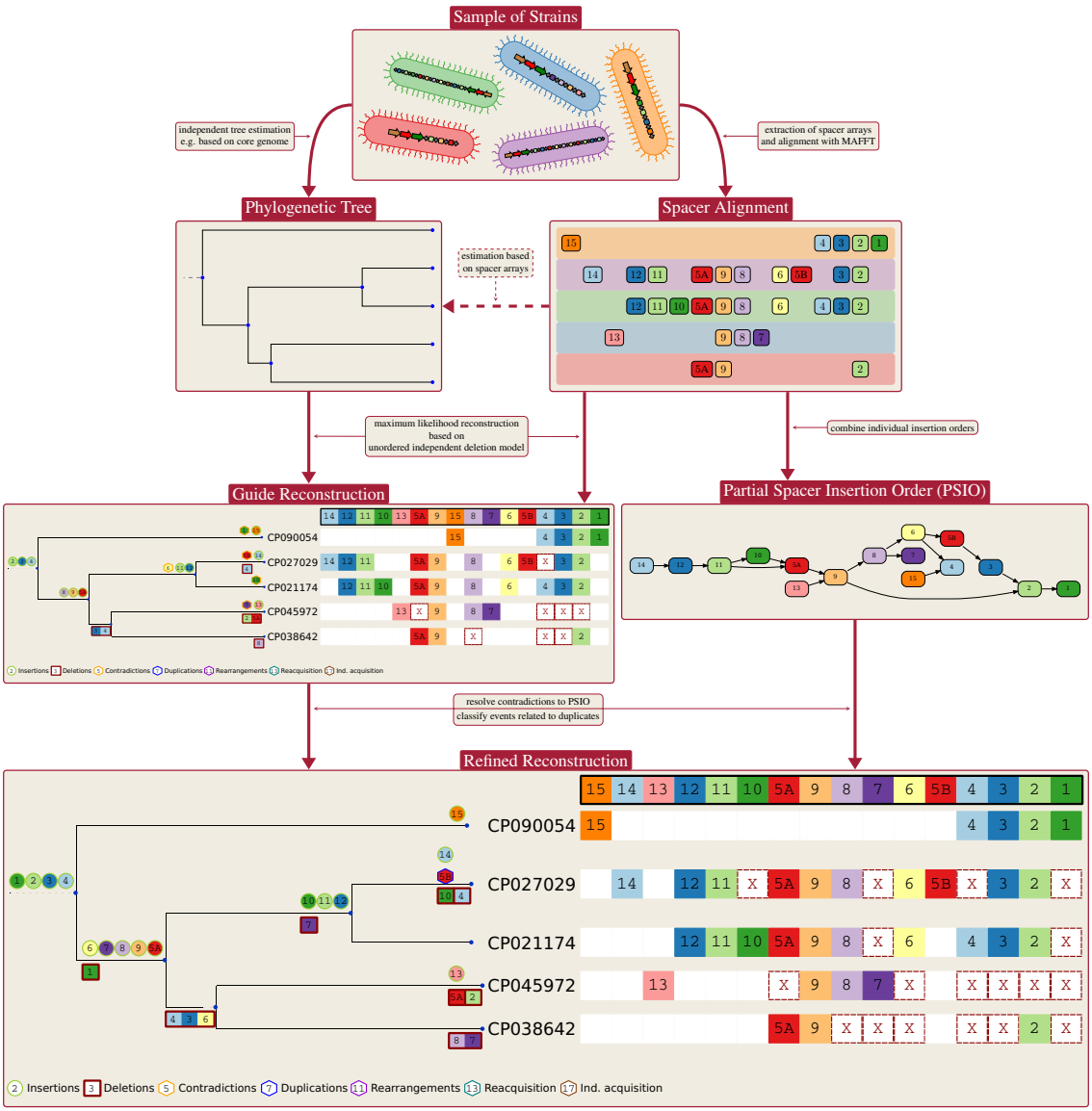


Fig. 3. Workflow of *SpacerPlacer*: ancestral reconstruction. To prepare for the *SpacerPlacer* analysis the first step is to extract the CRISPR arrays from their strain genomes and reconstruct a phylogenetic tree, e.g. based on genome alignments of the strains. *SpacerPlacer* then relabels and aligns the CRISPR spacers. The spacer alignment optionally allows us to reconstruct an alternative phylogenetic tree. Then *SpacerPlacer* builds the partial spacer insertion order (PSIO) and performs a guide reconstruction along the provided tree. This is followed by refining the reconstruction using the PSIO, resulting in a finished reconstruction of ancestral acquisition, deletion and more complex events of the considered group. The group portrayed here are CRISPR arrays of Cas type I-B from *Listeria monocytogenes* strains extracted from CRISPRCasdb. The tree shown is estimated based on the CRISPR arrays with *SpacerPlacer*.

issues.

Iterative PSIO-based refinement The guide reconstruction does not need to be consistent with the PSIO and can contradict polarized acquisition. Therefore, we iteratively refine the guide reconstruction using three straightforward steps:

1. Identify any contradictions between the insertions estimated by the ancestral reconstruction and the PSIO. If no contradictions are found, the reconstruction is complete. Otherwise, continue with step 2.

2. For each reconstructed spacer acquisition event that is in conflict with the PSIO, determine the parent node in the tree. Regardless of the most likely state calculated at this node in the guide reconstruction, fix the spacer to be present at this node; see Figure 4.

   This is done for all spacer positions in the MSAA and all PSIO-contradicting events along the tree.

3. Compute the maximum joint likelihood reconstruction with the added fixed ancestral states that help to resolve the contradictions. Note that the newly reconstructed insertion events may not be exactly at the fixed nodes, but can be reconstructed even closer to the root.

   Then go to step 1. The fixed ancestral states added are retained through all repetitions of this process.

For more details about this process see Supplementary Note 3. The PSIO is particularly advantageous for the reconstruction of older spacers, as more of the younger spacers can provide information about the insertion time. This contributes to a more reliable reconstruction of spacer deletions toward the trailer end of the array, as observed in Figure 7 1b) and 2b).

Classification of spacer duplication candidates The computed MSAA allows us to identify spacer candidates that are likely the result of spacer duplication, rearrangement, reinsertions, or independent acquisition events, since they are present in multiple columns of the alignment. Each identified candidate is relabeled such that each spacer/column in the MSAA is unique. In Figure 4, spacer 3 has a set of two duplication candidates $\{3\text{A}, 3\text{B}\}$.

All of these event types share the characteristic of not conforming to the rules of polarized acquisition. As a result, they have the potential to cause misleading conflicts within the PSIO, e.g. when both versions of a duplicated spacer are included in the PSIO. To overcome this problem, we adopt the following approach:

Since each duplication candidate found in the MSAA is uniquely labeled, they are treated as independent spacers in the guide reconstruction. In the refinement step, only the candidate spacer that was acquired the earliest in the guide reconstruction remains in the PSIO and can produce conflicts with the PSIO in the guide reconstructions. Any conflicts to the PSIO arising from other duplication candidates are ignored at this stage. In Figure 4, 3B is found to be the "oldest" candidate and therefore remains in the PSIO, while 3A (dashed) is not considered to be part of the PSIO.

After refinement, candidates (excluding the "oldest" candidate), or rather their insertion events, are classified according to the reconstructed evolutionary events of all candidates. We distinguish between rearrangements, reacquisitions, duplications, independent acquisitions, and other more complex types of duplicate events. In Figure 4, the insertion of 3A on s_0 is classified as a duplication event, since 3B is already present at the root and 3A is in the middle of the array.

For a detailed description and illustration of the classification procedure and the different classes see Supplementary Note 3. However, more than 96% of the spacers are in line with polarized insertions and deletions (Table 2), justifying this classification step outside the scope of the model-based reconstructions.

Parameter estimation. We estimate the spacer deletion rate ρ and the mean block length of spacer deletions α by numerically maximizing the likelihood given by the independent deletion model or blockwise deletion model for the *reconstructed* spacer deletions.

For a group of spacer arrays, let $\mathcal{T}$ be the corresponding phylogenetic tree and N_b the number of spacers present at the parent node of branch $b \in \mathcal{T}$. Also, let K_b describe the set of reconstructed spacers that are deleted along branch b such that $|K_b|$ is the total number of spacers deleted at branch b. Moreover, let $\mathcal{K} := \{K_b \,|\, b \in \mathcal{T}\}$ and $\mathcal{N} := \{N_b \,|\, b \in \mathcal{T}\}$ be the collection of all these sets for all branches in the tree.

We can compute the likelihood $L_{\text{B}}(b, N_b, K_b, \rho, \alpha)$ of deletions K_b to occur along branch b, while retaining $N_b - |K_b|$ spacers, under the block deletion model with deletion rate ρ and mean block length α. Then the total likelihood of a group under the BDM is given by

$$\mathcal{L}_{\text{B}}(\mathcal{T}, \mathcal{N}, \mathcal{K}, \rho, \alpha) := \prod_{b \in \mathcal{T}} L_{\text{B}}(b, N_b, K_b, \rho, \alpha). \qquad \textbf{(1)}$$

To compute the likelihood under the IDM we simply set $\alpha = 1$, i.e. $\mathcal{L}_{\text{I}}(\mathcal{T}, \mathcal{N}, \mathcal{K}, \rho) := \mathcal{L}_{\text{B}}(\mathcal{T}, \mathcal{N}, \mathcal{K}, \rho, 1)$.

The (analytic) computation of the likelihood function of the block deletion model L_{B} is not trivial and described in detail in Supplementary Note 1. We give only a short description here.

The main issue to solve is that a connected component of consecutive spacers that have been lost along a branch of the phylogeny can arise from a combination of multiple block deletion events of different lengths. *SpacerPlacer* provides two ways to compute the combined likelihood for each of these connected components. On the one hand, we provide a simplified approach that approximates the likelihood, ignoring the exact time and sequence of possible combinations of different block deletion events along a branch.

On the other hand, in a more precise approach, we take into account the order and timing of the events. This gives rise to several ordinary differential equations that can be solved analytically, but solving them is computationally expensive. We give an illustration of the differences between the simplified and precise likelihood functions in Supplementary Figure S1.

Surprisingly, we found that the simplified likelihood function performs slightly better in simulations (Supplementary Figure S7) compared to the precise likelihood function (Supplementary Figure S6).

As the results are based on the branch lengths of the phylogenetic tree, the estimated deletion rate $\hat{\rho}_{\text{B}}$ is proportional to the time scale of this tree. In the core genome-based trees used here, the branch length is scaled as the expected number of substitutions per site. In this scaling, an estimated spacer deletion rate of 374 indicates that a spacer is deleted 374 times more frequently than a single site mutates.

Bias correction for unobserved extinct spacers Spacers that existed at some time in the past can get deleted such that these spacers are not observed in any of the sampled spacer arrays. Consequently, deletion events related to such spacers cannot be reconstructed. This results in an underestimation of the deletion rate ρ in both the IDM and the BDM. Therefore, we correct the likelihood for such unobserved spacers by conditioning on maintaining the spacers in at least one of the samples that are descendants of the individual where the spacer has been acquired, similar to an approach introduced by Joseph Felsenstein in a different setting [52]. The effects of unobserved spacers on parameter estimates are significant, but our bias correction can effectively address this issue (Supplementary Figure S8).

Let us consider the IDM case to illustrate this correction: Let $L_{\text{I}}^{\text{unobs}}(b, \mathcal{T}, \rho_{\text{I}})$ be the probability that a spacer that is acquired

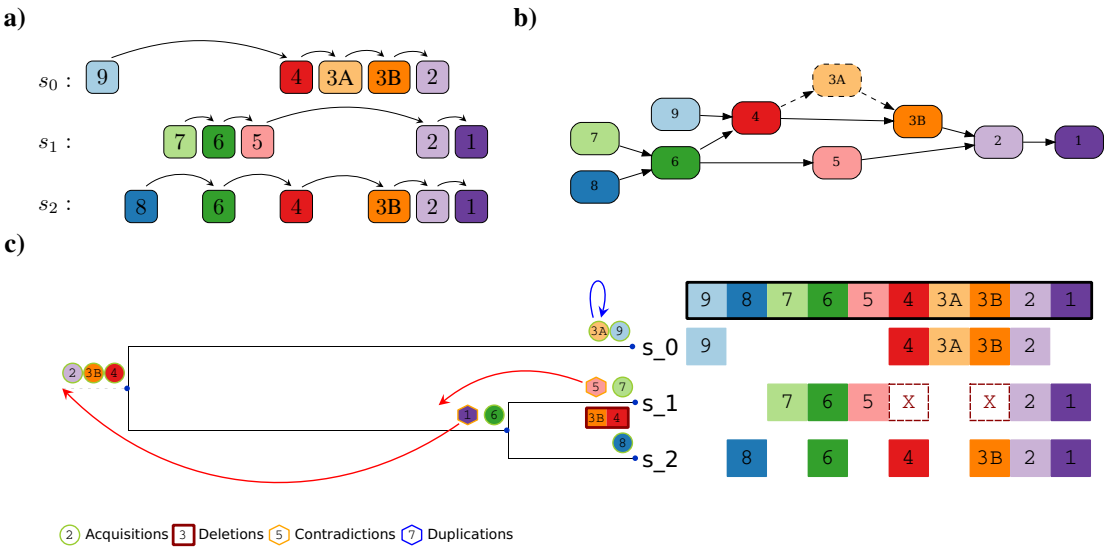


Fig. 4. Partial Spacer Insertion Order based refinement of the guide reconstruction. a) Relabeled multiple spacer alignment of a group with 3 arrays s_0, s_1 and s_2, together with their induced orders. Only one step relationships are indicated. Spacers 3A and 3B are relabeled duplicates of each other. **b)** Partial Spacer Insertion Order (PSIO) as induced by the spacer array group in a). The spacer 3A is categorized as a potential duplication event occurring between 3A and 3B. Thus, it is removed from the PSIO for the refined reconstruction (dashed), and contradictions due to 3A are ignored. **c)** Illustration of the guide reconstruction and the PSIO-based refinement process. The acquisition event of 5 contradicts the acquisition of 6 in the parent branch. Similarly, the insertion of 1 is a contradiction to 2 (or 3, 4). Both contradictions are solved in the refinement process by fixing the spacer to be present at the parent node, as indicated by the red arrows. Spacer 3A is classified as a duplication event, since 3A is located in the middle of the array and spacer 3B is already present on this branch, indicated by the blue arrow.

at branch b and subsequently deleted at rate ρ_I along the branches of the subtree below b will be absent in all samples. The corrected likelihood, conditioned on observing the g_b spacers acquired at branch b in the sample, is then given by

$$\mathcal{L}_{\mathrm{I}}(\mathcal{T},\mathcal{N},\mathcal{K},\rho_{\mathrm{I}}) \cdot \prod_{b\in\mathcal{T}} \left(1 - L_{\mathrm{I}}^{\mathrm{unobs}}(b,\mathcal{T},\rho_{\mathrm{I}})\right)^{-g_b}.$$

SpacerPlacer computes this likelihood and derives a bias-corrected estimate $\hat{\rho}_{\mathrm{I}}^{\mathrm{corr}}$ of the deletion rate. For a more precise description of the computation of $L_{\mathrm{I}}^{\mathrm{unobs}}$ see Supplementary Note 1D.

For the BDM the exact computation of this unobserved likelihood $L_{\mathrm{B}}^{\mathrm{unobs}}$ is hard due to the possibility of arbitrarily long deletions originating from unobserved spacers. Thus, we approximate the effect of the conditioned likelihood in the BDM using the correction factor from the IDM.

Bias correction for the mean spacer deletion block length

The maximum likelihood estimator of a geometrically distributed random variable is known to be biased, especially for a low number of observations.

Therefore, in addition to the IDM-based bias correction for unobserved spacers, we use a standard bias correction for the estimated mean spacer deletion block length α after the numerical estimation step; see Supplementary Note 1D. The improvement of parameter estimates by the combination of both bias corrections is evident when comparing Figure 5 and Supplementary Figure S7.

Likelihood ratio test. Based on the likelihoods computed in the previous section, we formulate a maximum likelihood ratio test to investigate within *SpacerPlacer* whether spacers are (likely) deleted in blocks or individually.

Here, a likelihood ratio test can be used, as the independent deletion model is nested in the block deletion model (setting $\alpha = 1$). We formulate the hypotheses which correspond to the independent and block deletion model

$$H_0: \quad \rho_{\mathrm{B}} > 0, \alpha = 1 \quad (\text{i.e. } \rho_{\mathrm{I}} > 0), \quad (\mathrm{IDM}) \qquad \textbf{(2)}$$

$$H_1: \quad \rho_{\mathrm{B}} > 0, \alpha \geq 1, \quad (\mathrm{BDM}) \qquad \textbf{(3)}$$

where the test statistic is given by

$$\lambda_{\mathrm{LR}} := \frac{\sup_{\rho_{\mathrm{I}}>0} \prod_{b\in\mathcal{T}} L_{\mathrm{B}}(b, N_b, K_b, \rho_{\mathrm{I}}, 1)}{\sup_{\rho_{\mathrm{B}}>0,\alpha>1} \prod_{b\in\mathcal{T}} L_{\mathrm{B}}(b, N_b, K_b, \rho_{\mathrm{B}}, \alpha)}. \qquad \textbf{(4)}$$

If λ_{LR} is significantly small we dismiss H_0 on $\mathcal{T}$ and accept H_1, i.e. the block deletion model fits the data better and is more likely. Otherwise, we cannot dismiss H_0, i.e. both models explain the data reasonably well.

See Supplementary Note 1 for a more detailed description of the likelihood calculation and the likelihood ratio test.

Analyzing the patterns of spacer array evolution across numerous bacterial genomes.

Parametric models enable accurate estimation of the rates and lengths of spacer deletions. We evaluated the performance of *SpacerPlacer* using simulated data. *SpacerPlacer* performs well in estimating the (median) deletion rates in simulations and counteracts the bias of likelihood models towards too few events by incorporating

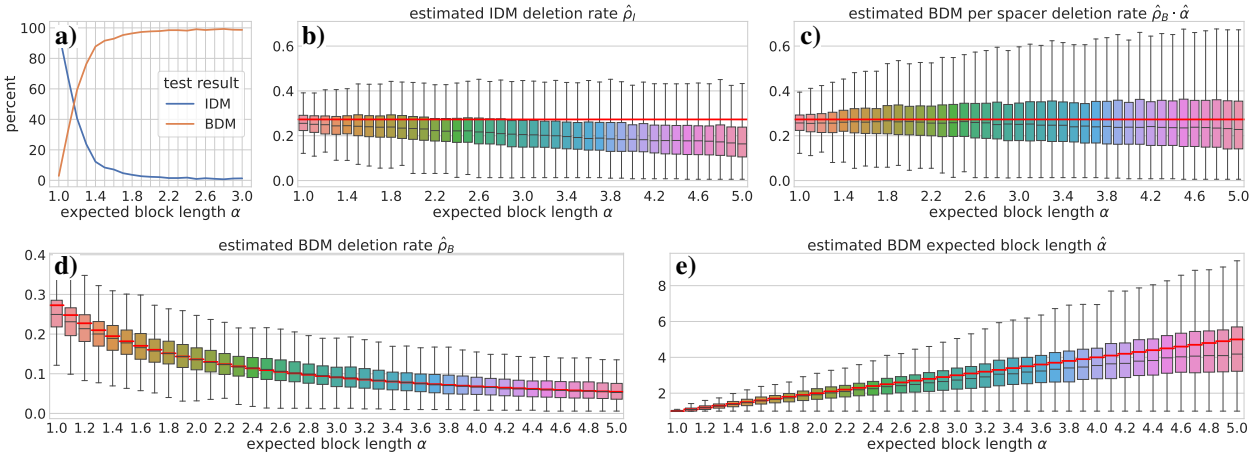


Fig. 5. Parameter estimation and LRT-performance on simulated data. We evaluated *SpacerPlacer* on simulated spacer arrays. The simulations are based on random coalescent trees for different parameters (ρ_B, α), where the deletion initiation rate ρ_B in the BDM was always chosen such that the per spacer deletion rate $\rho_B \cdot \alpha$ remains constant for different choices of α. True parameters are shown as red lines. *SpacerPlacer* was then used to reconstruct the ancestral spacer arrays and estimate the corresponding parameters (using the simulated coalescent trees and including both bias corrections), shown as boxplots here.
a) The percentages of likelihood ratio tests (LRTs) that indicate a BDM (orange line) or do not reject the IDM (blue line) are shown. For $\alpha > 1.6$ the likelihood ratio test in *SpacerPlacer* can correctly recognize the BDM used in the simulation (ground truth). b) Estimates of the spacer deletion rate ρ_I based on the IDM. One can clearly see that the accuracy of the estimated per spacer deletion rate (difference to red line) drops with the expected block length used in the simulations. c)Estimates based on the BDM for $\rho_B \cdot \alpha$. As expected, parameter estimation based on the BDM performs better than estimation based on the IDM. Furthermore, the performance is strong for all values of α, only decreasing for high values of α. d)+e) Estimates of ρ_B (bottom left) and α (bottom right) separately. As their product is fixed, the underlying ρ_B decreases with growing α. Both parameters are well estimated in the median. There is a trade-off between ρ_B and α since both parameters are estimated simultaneously.

the information from the PSIO, indicating a well-performing reconstruction algorithm (Figure 5 (b, $\alpha = 1$), (d), and (e)).

The likelihood ratio test reliably distinguishes independent individual spacer deletions compared to joint deletion of adjacent spacers, as soon as the average length of jointly deleted blocks of spacers exceeds 1.6 in the simulations (Figure 5(a)).

In simulations where spacer deletions occur in blocks, assuming an IDM model results in an underestimation of the per spacer deletion rate (Figure 5(b)). Thus, it is important to incorporate block deletions into the parametric model to correctly estimate the frequency and length of spacer deletions.

When the BDM is set as the underlying model, *SpacerPlacer* is able to accurately estimate the per spacer deletion rate and the parameters ρ_B, α individually (Figure 5(d), and (e)). The estimation quality only degrades for very high α when deletion events that collapse (almost) the entire array occur more frequently.

If the effective rate of deletion per spacer by taking the product from ρ_B and α is considered (Figure 6(c), the IDM and BDM estimates are more comparable. Here, the tendency to underestimate the deletion rate when using the IDM becomes clear (see Figure 5(b)).

The estimation of parameters in the BDM is non-canonical due to the dependencies created by jointly lost neighboring spacers (Supplementary Note 1). In addition, we need to use bias corrections for both ρ_B, α. BDM-based estimates without bias corrections tend to underestimate ρ_B and α, as can be seen in Supplementary Figure S7 and S8.

Due to the typically short length of spacer arrays, the variance of the estimated rates and block lengths is high for individual groups, suggesting that the estimated parameters are more reliable in well-curated, large groups and when averaging over multiple CRISPR array groups.

Blockwise deletions of neighboring spacers are frequent. In the past, various models have been proposed to explain the distribution of joint deletions. Until now, however, statistical tests have not been able to determine the most likely scenario, as the deletion patterns observed in different CRISPR array datasets are consistent with different models for spacer deletions. In particular, both the IDM and BDM contain losses of adjacent spacers that can arise from combinations of multiple overlapping deletions; only their frequency and distribution change.

To determine the mode of spacer deletions in CRISPR arrays, we applied the likelihood ratio test described in the previous section to the data sets extracted from CRISPRCasdb and reconstructed the ancestral CRISPR arrays to investigate the properties of spacer array evolution.

We found that in 76 % of the CRISPR array groups, the likelihood ratio test rejected the null hypothesis, i.e. the independent deletion model, in favor of the BDM. For a breakdown of the test results with respect to Cas type see Supplementary Figure S11. Although the IDM cannot be rejected in the remaining cases, this indicates that, overall, the block deletion model describes the natural process of spacer deletions significantly better.

For the expected block length parameter α the median of all estimated parameters based on the CRISPRCasdb datasets

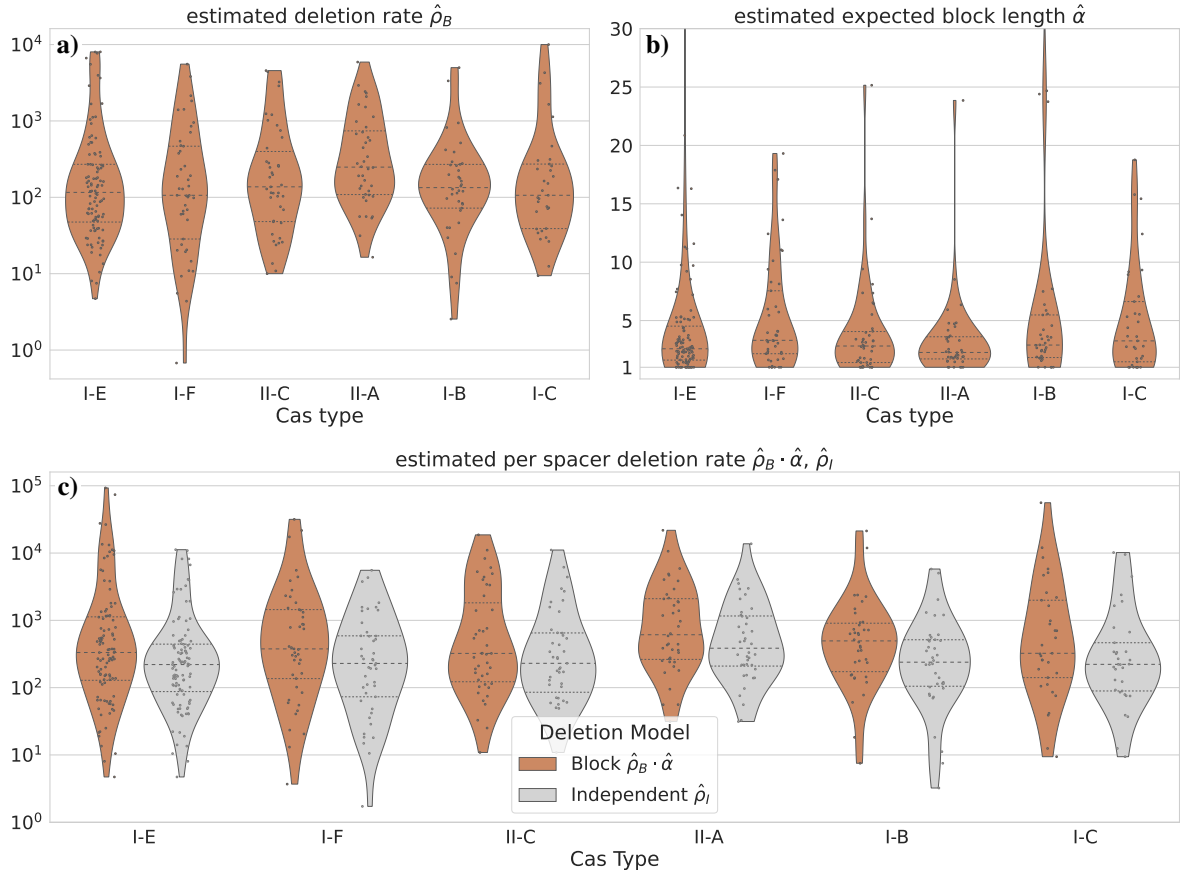


Fig. 6. Parameter estimates on CRISPRCasdb data. The estimated parameters for (a) the deletion initiation rate, and (b) the expected length of deletion events in a block deletion model, as well as (c) the resulting per spacer deletion rate for different Cas types are shown. The maximum block deletion length shown in the plot is limited to 30 for readability. There are no significant differences between the Cas types. The variation of individual estimates is high, but the median estimates of the per-spacer deletion rate are consistently similar. For comparison, we also show the estimates if the underlying model is the less accurate IDM (light gray) in (c). Although IDM-based estimates correlate with BDM-based results, the same bias as in the simulations can be observed.

is $\alpha \approx 2.73$. This is consistent with our statistical analysis of the block deletion lengths observed by Lopez-Sanchez et al. [37]. The length of the spacer blocks jointly deleted in their experiment fits a geometric distribution (p-value $= 2.76 \cdot 10^{-5}$) and has a mean deletion length of $\alpha = 3.32$. Therefore, in all subsequent analyzes, we will only present the results derived from the more accurate BDM, unless otherwise indicated.

Similar spacer deletion patterns across different cas types. The estimates of the spacer deletion rate are comparable for different Cas types, indicating that, at least for the investigated Cas types, no cas-dependent molecular mechanism influences the deletion of spacers. The estimated parameters for different Cas types are shown in Figure 6.

Spacer deletions are frequent. Note that the estimated deletion rate will depend on the scaling of the underlying trees. Here, we can estimate the evolutionary turnover speed of CRISPR arrays because the core genome trees used for the estimates are time-scaled based on the average per site mutation rate.
In the BDM, we estimate the deletion initiation rate ρ_B and the mean block length α of deletions. From these we can calculate the average deletion rate per spacer $\rho_B \cdot \alpha$. The median of the spacer deletion rate per spacer in the BDM is 374, indicating that a spacer is deleted at a rate 374 times higher than the rate at which individual nucleotide mutations occur. In particular, since spacers (and their repeats) are generally smaller than 100 bp in length, the deletion of spacer sites occurs at a faster rate than nucleotide mutations within the array. The loss of immunity by spacer deletion is therefore the greater risk for the host compared to mutations within the spacer.
Using the relationship between the average deletion rate per

	α	ρ_B	ρ_I	$\alpha \cdot \rho_B$
mean	4.58	601	895	2602
median	2.73	137	230	374

Table 1. Mean and median parameter estimates in the independent and block deletion model. We estimated the parameters of the independent and block deletion model on CRISPRCasdb data across all groups with at least one reconstructed deletion. Left: Estimates for the BDM. In the BDM deletions are initiated at a rate ρ_B and cover a geometrically distributed number of spacers with mean α. Middle gray: Estimate of the deletion rate ρ_I of the IDM. In this model, all initiated deletions have length 1. Right: $\alpha \cdot \rho_B$ is the approximate deletion rate per spacer in the BDM and roughly corresponds to ρ_I in the IDM.

spacer, the insertion rate, and the average length of the array, we can also give an ad hoc estimate of the median insertion rate $\theta \approx 6522$. We find that the variance between the estimates of θ of each group is large (Supplementary Figure S5).
This indirect rough estimate of the average insertion rate should be considered with extreme caution, especially for single groups. As the frequency of spacer acquisitions depends on environmental fluctuations such as co-evolution with phages, it remains unclear how an inferred average acquisition rate can be interpreted.
In contrast, spacer deletions can take place without the involvement of any foreign elements, which suggests that they can be better explained by a constant rate.
However, together the estimates indicate that spacer turnover is significantly faster compared to mutations.

Spacer duplications and rearrangements are rare. Spacers that can be explained solely by insertions and deletions were by far the most common case in the CRISPRCasdb datasets, but we also identified roughly 3% of all spacers that form loops in the PSIO and are only explainable by more complex event types, as shown in Supplementary Figure S2. Of these complex events, 86% are simple duplications, 5% are rearrangements, and only 1% new insertions of the same spacer (Table 2). This aligns with the finding of Kupczok et al. that recombination between CRISPR arrays is rare [53].

Distribution of deletions along CRISPR arrays aligns with deletions caused by replication misalignments between repeats. To investigate the distribution of deletions according to their position in the spacer array, we compared *SpacerPlacer* reconstructions for two different scenarios, Figure 7. We either assumed that spacer losses occur evenly distributed along the array without the need to pair different repeats or, in the second scenario, that spacers at the edges of the array are less often part of block deletions according to the lower number of neighbor spacers and repeats that could mismatch with them.
To get a clear picture independent of the length of individual arrays, we considered the relative deletion position, i.e. the fraction between the position of the deleted spacer and the total array length at the ancestral parent node above the deletion; see Figure 7. To avoid confusion due to counterintuitive effects of binning, we only show groups with an average array length between 15 and 25 in Figure 7. The distributions are similar across different average array lengths (Supplementary Figure S9).
In both simulation scenarios, the accuracy of ancestral reconstruction towards the leader end, where new spacers are acquired, is lower than towards the trailer end of the array (Figure 7 (1) and (2)). Therefore, we expect that reconstruction at the leader ends for CRISPRCasdb data are similarly impacted (Figure 7 (3)).
The frequency of reconstructed spacer deletions based on CRISPRCasdb data shows a fall-off towards both boundaries of the array with a roughly uniform distribution in the middle of the array. This is different from what we would expect from a distribution that is directly dependent on the length of the array, such as the fragment deletion model proposed by [35] or models that favor deletions on one side of the array [27, 29]. The uniform distribution in the middle combined with the drop towards both ends of the array is consistent with a geometric distribution of deletion lengths combined with current models of a pairwise repeat-dependent deletion process [11].
Concordantly, a uniform deletion rate along the array was also found to be a representative model for spacer deletions in a related simulation study by Han et al. who considered CRISPR array spacer diversity in a co-evolutionary model between bacteria and phages [42].

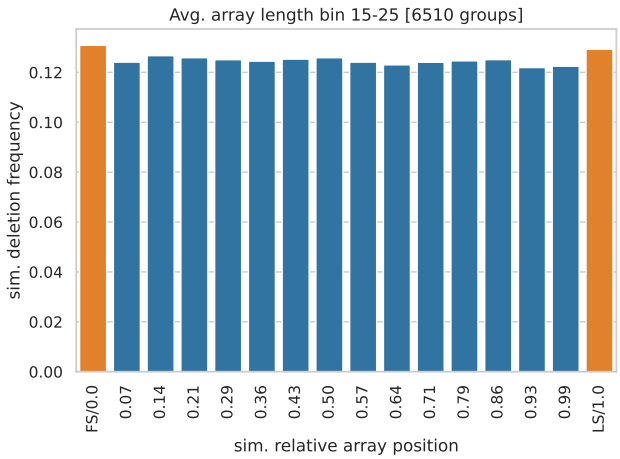
No evidence of age-dependent selection along the spacer array, except possibly at the first and last positions. If selection pressures would differ along the array, e.g. if more anciently acquired spacers would provide a higher or lower fitness benefit than more recently acquired spacers, one would expect that the observed deletion rate along the array correlates with this selective advantage. The absence of a clear sign of such a selective gradient indicates that the selective pressures to maintain a particular beneficial spacer and the fitness costs to keep a useless spacer are either constant over time or small. For spacers that are close to the first or last position, we expect a strong influence of the underlying repeat-dependent deletion process on the decrease of deletion frequencies. However, when looking at the absolute positions, the CRISPRCasdb data show more reconstructed deletions of the first spacer than of the last spacer, Figures 7 (3) and 8. For the first spacers, the task of inferring the selection strength is particularly challenging, as the bias created by the reconstruction can obscure our analysis. Thus, we simulated spacer arrays to account for this reconstruction effect. The higher number of reconstructed deletions of the first spacers in the CRISPRCasdb data is contrary to the trend seen in neutral symmetric simulations, where fewer deletions of the first spacers than of the last spacers can be reconstructed (Figure 8). This suggests that the spacer deletion rate is higher towards the leader end. In the discussion, we describe potential selective benefits and alternative explanations that could cause this difference.

Discussion

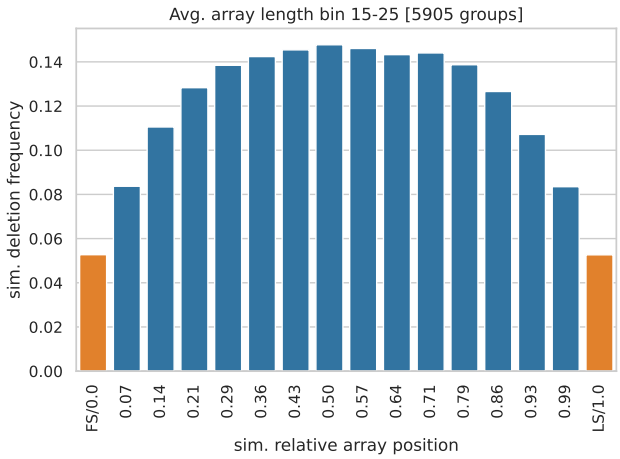
Inferring the dynamics of spacer array evolution is challenging, as arrays are typically short and provide very limited information. Furthermore, the acquisition of new spacers correlates with environmental conditions, such as the abundance of phages. In contrast, spacer deletions are less dependent on the environment and thus are more likely to follow a molecular clock. Fortunately, the order of spacers provides additional information and repeated acquisitions of the same spacer are extremely rare. These two properties form the basis for the reconstruction approach of *SpacerPlacer* to first reconstruct the most likely acquisition point(s) and the resulting deletions along the tree independently for each spacer and then refine these according to the PSIO.
The parametric model-based approach allows to investigate the likelihood of evolutionary forces at play and paves the

Simulated deletions along the array

1a) array boundary does not affect deletion block length

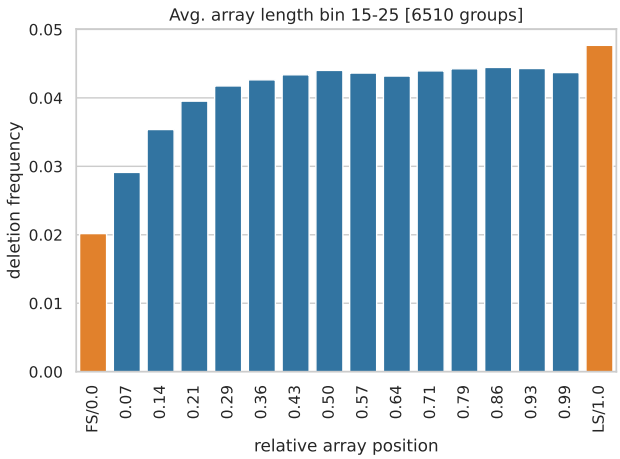


2a) boundary affects deletion block length

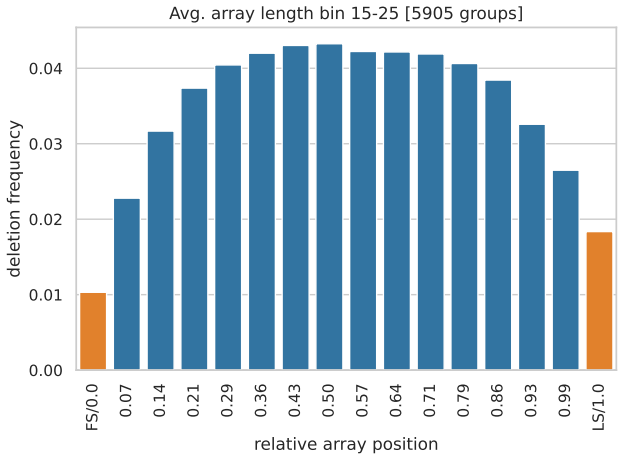


Reconstructed deletions along the array

1b) reconstructed from simulations shown in 1a



2b) reconstructed from simulations shown in 2a



3) reconstructed from CRISPRCasdb data

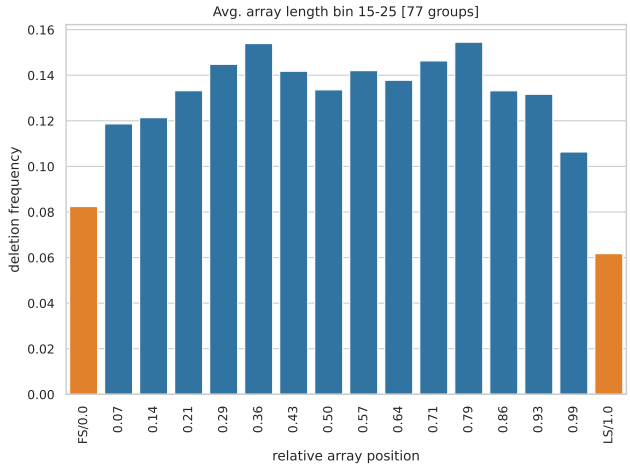


Fig. 7. Deletion frequency with respect to the position along the array. The bars display the proportion of branches in the tree where a deletions occurs for different relative positions along the array as indicated by the x-axis. **1a)** shows the deletion frequencies for simulations under an artificial block deletion model, where deletions of internal spacers and spacers at the ends of the array occur at the same frequency. In contrast, **2a)** shows the deletion frequencies for simulations under the more natural block deletion model where block deletions cannot exceed the array boundary as nearby repeats need to align. Here, deletions are naturally less likely to occur towards both ends. **1b)** and **2b)** show the corresponding values based on the events reconstructed by *SpacerPlacer*. In both cases *SpacerPlacer* is able to reconstruct the deletion distribution of the respective models, except near the leader end of the array. This result is not surprising, as errors in the reconstruction of recently acquired spacers are naturally less likely to be corrected by the PSIO. In reconstructions based on CRISPRCasdb data, **3)**, the deletion frequencies show a distribution similar to the deletion model with a boundary effect (2a and 2b). But there are more reconstructed deletions at the first spacer position than at the last spacer position, contrasting the pattern we would expect based on the neutral simulations. For each plot, we bin all deletion events according to their relative position compared to the current length of the array at the parent node of each branch. Each bar in the figures corresponds to a bin, with the label on the x-axis indicating the upper limit of the bin. The first spacer (FS/0.0) and the last spacer (LS/1.0) are placed in their own categories (shown in orange).

total nb. of spacers	24090	
PSIO-compatible spacers	23285	0.967 (prop. of spacers)
duplication candidates	805	0.033 (prop. of spacers)
complex event type	count	proportion of candidates
duplications	695	0.863
rearrangements	43	0.053
reacquisitions	8	0.010
independent insertions	7	0.009
other events	52	0.065

Table 2. Number and proportions of duplication events. The total number of spacers is divided into spacers that are compatible with polarised insertions and deletions only, and spacers that are duplication candidates according to the MSAA and the PSIO. Duplication candidates are further classified into reconstructed complex event types.

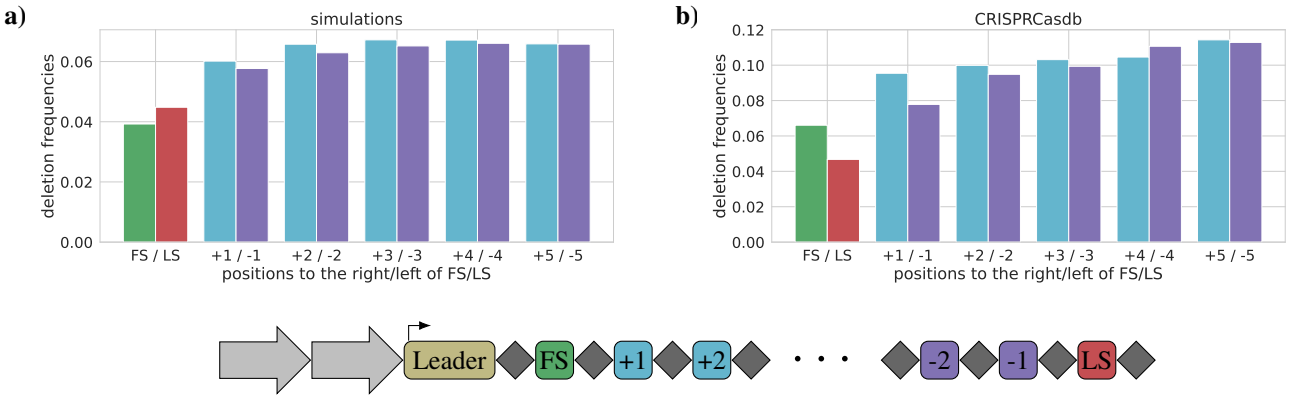


Fig. 8. Deletion frequencies with respect to the proximity to the array boundary. We show the deletion frequencies for absolute positions next to the first and last spacer. **a)** shows the reconstructed deletion frequencies based on simulations with array boundary effect. Note, that the first spacer (FS) appears to be deleted less often than the last spacer (LS), although the simulation is perfectly symmetric. But, just as in Figure 7, the reconstruction is more likely to underestimate the frequency of FS deletion. **b)** shows the deletion frequencies for reconstructions based on CRISPRCasdb data. Here, the pattern is reversed, and the LS is significantly less frequently deleted than the first spacer. Comparing the FS and LS deletion frequencies between **a)** and **b)** suggests that compared to a symmetric model, either the LS is deleted less frequently, the FS is deleted more frequently, or both.

way to detect molecular mechanisms and selective pressures acting on the presence and absence of spacers in the array.

Evidence of misalignment based deletions can be observed in ancestral reconstructions. The benefits of our modeling framework are best highlighted in the context of the dynamics of spacer deletions. Deletions that arise from repeated misalignment introduce three effects that can be detected in the inferred parameters and reconstructions based on the BDM.
First, if a misalignment containing multiple repeat-spacer units forms, multiple adjacent spacers can be jointly deleted in one event. This notion is supported by previous research, where blockwise deletions of multiple spacers have been experimentally observed [37].
Second, what we call the "array boundary effect" on the deletion frequency: Near both ends of the array, the deletion frequencies should be reduced because repeats at the boundary of the array have a lower number of nearby repeats with which they can align for a deletion event.
Third, deletions by repeat misalignment are, a priori, independent of the Cas genes, i.e. the Cas type.
In our model-based approach we can investigate these properties and also reveal hints at selective pressures and mechanisms deviating from the neutral repeat mismatch inspired model by comparing the patterns of reconstructed spacer deletions in neutral simulations and empirical data.

Using the partial spacer insertion order is crucial for reconstructions of spacer arrays Enforcing the chronological order of spacer insertion during the refinement step based on the PSIO counteracts the tendency of likelihood models to explain patterns by fewer (deletion) events. This allows *SpacerPlacer* to reconstruct ancestral spacer arrays with greater precision and improves rate estimates. In the CRISPRcasdb data, more than a quarter of the spacers (6.773 out of 24.090) are affected, i.e. their acquisition events are relocated during the refinement procedure, resulting in a higher number of spacer deletions.

The reconstructed spacer deletions are subsequently used by *SpacerPlacer* to estimate the maximum likelihood parameters in the more complex evolutionary model with blockwise spacer deletions. Because of the relatively low sample sizes, where many groups of spacer arrays have large spacer overlap and can be explained by a few events, the variance in the individual estimates is quite large. However, our simulations show that with large sample sizes, in terms of group size and number of groups, *SpacerPlacer* accurately predicts the median of the parameters.

Spacers deletions are frequent and often involve multiple spacers. The parameters estimated in groups of spacer arrays from CRISPRCasdb reveal two interesting characteristics of spacer array evolution.

First, the deletion rate per spacer is 374 times higher than the mutation rate per site, highlighting the potential of spacer arrays to track evolutionary events at a much shorter time scale compared to mutation-based tracking, such as MLST.

Second, blocks of jointly deleted spacers occur frequently, and their length distribution is consistent with a geometric distribution with mean 2.7 that is interrupted if the block length exceeds the spacer array.

Naturally, the actual process that determines the number of jointly lost spacers is more complex than a geometrically distributed random variable, but this simplification provides an efficient and precise computation of the likelihood function in *SpacerPlacer*. Furthermore, there is experimental evidence that the lengths of deletion blocks are geometrically distributed [37], which is in line with the current understanding of spacer deletions caused by repeat misalignment during replication [11].

Note that for a geometrically distributed block length, single-spacer deletions remain the most common event (37% for $\alpha = 2.7$), and the deletion of a long block of multiple spacers becomes increasingly rare. We inferred the distribution of the lengths of the joint deletions in our reconstructions based on CRISPRCasdb data (Supplementary Figure S14). The inferred empirical distribution is consistent with a geometric distribution for smaller lengths but, notably, does have a heavier tail. Thus, we hypothesize that the deletion process may, to a lesser extent, involve additional mechanisms, such as fragment deletions and the loss of entire CRISPR arrays.

Our results indicate that joint deletions are a common phenomenon in CRISPR array evolution and show no significant differences in their deletion rates or lengths between different Cas types. This suggests that spacer deletions are not controlled or regulated by CRISPR-specific molecular mechanisms or Cas genes but rather occur in a random fashion. Quite contrary to this view, frequent deletion of spacers and renewal of spacer composition is essential for the effectiveness of the system [44] and could, in principle, have fostered the emergence of a dynamic deletion system similar to primed adaptation. Interestingly, deletion frequencies can be subject to selection indirectly when multiple CRISPR arrays with different lengths, and therefore due to the boundary effect also different per spacer deletion rates, coexist [34, 54].

Reconstructed duplications shed light on the mechanisms of spacer deletions Our analysis of CRISPRcasdb spacer evolution showed that complex events, including duplications, rearrangements, and reacquisitions, are rare and dominated by duplication events. Only 3% of the considered spacers have been reconstructed as duplicated, and 171 out of 333 CRISPR array groups do not include a potential duplication.

The number of observed duplications is much lower than we would expect from the experiment in [37] where roughly 10% of the arrays had a duplication in addition to the deletion of spacer 258. Here, the number of reconstructed duplications is less than 3% of the reconstructed events.

One explanation could be that duplications are harder to reconstruct than deletions. We expect it to be difficult to correctly detect and classify duplication events in complex scenarios that are composed of multiple events or obscured by other deletions. Thus, the fraction of duplications, rearrangements, and ectopic insertions obtained here is a lower bound for such events.

An alternative explanation could be that duplicated spacers are lost at a higher rate than nonduplicated spacers, which would lower the abundance of duplicated spacers over time. This fits the postulate that most spacer deletions occur due to replication misalignment ("slippage") of the repeats [11, 39]. Notably, for duplicated spacers, there is an additional possibility for slippage: matching the two duplicated spacers instead of the repeats. This could lead to an increased deletion rate of duplicated spacers, which is from an evolutionary perspective a fortunate property of CRISPR arrays, as a duplicated spacer does not provide an additional benefit to the cell. To our knowledge, no experimental verification of increased deletion rates for duplicated spacers has yet been done.

Decreasing deletion frequency at both ends indicate no age dependent selection One of the main questions motivating this work was whether spacers at different positions in the spacer array are under different selection pressures.

It is important to distinguish the position along the reconstructed ancestral array and the current position among the observed sample arrays. Naturally, when looking at the deletions in the spacer arrays of a sample, these seem to appear predominantly at the distal end. However, at the time in the past when the deletions occurred, their position was closer to the leader end. This observation bias has led to the perception within the scientific community that deletions occur more frequently among shared spacers at the end of the array, rendering them less beneficial than more recently acquired spacers, which seem to be less often deleted.

Our results indicate that this impression is a misinterpretation of the accumulation of observed deletions towards the distal end in alignments of current spacer arrays. Naturally, deletions of older spacers are more often observed in such alignments, as those existed for a longer time during which deletions can occur compared to more recently acquired spacers. Here, we correct this misleading impression by instead considering the reconstructed deletion positions in ancestral spacer arrays.

We observe no clear sign of selection acting differently on deletion frequencies along the reconstructed arrays, at least for spacers that are not among the first or last positions.

While we do not at all propose that CRISPR arrays evolve neutrally, our results hint at either overall weak selection pressures to delete specific spacers or an equally strong selection along the array to maintain spacers.

Deletion rates of the first and last spacer of CRISPR arrays We found that the drop in deletion frequency towards the edge of the array is particularly pronounced for the last spacer

compared to the first spacer. This difference could be caused by the often observed excessive mutation of the last repeat [9], which renders them less likely to participate in deletion events based on misaligned repeats.

Vice versa, we can argue that the difference between the last and first spacers arises because of selective effects at the leader end. As we can only reconstruct deletions from individuals whose offspring survived, this indicates that many recently acquired spacers are not beneficial in the long run and can be deleted again quickly. Another selective advantage of deleting spacers at the leader end could arise due to the expression levels along the spacer array. Modifying the composition of spacers at the first most expressed positions might be particularly beneficial and could lead to the observed increased deletion rates at the first spacer positions.

Interestingly, our findings are in line with recent observations by López-Beltrán et al. [55]. They showed that the environmental abundance of the phages and plasmids targeted by spacers varies along the spacer array. They found that at the first positions, the targeted phages are present in the environment, whereas the targets of the spacers directly following are less abundant. This indicates a lower selection pressure to maintain recently acquired spacers. Toward the trailer end they observed that the spacers correspond to more abundant but different phages and plasmids, which fits well with the reconstructed deletion frequencies here.

Applications and current limitations Although we did not find significant differences between Cas types I-B, I-C, I-E, I-F, II-A, and II-C based on CRIPSRCasdb data here, the parametric models behind *SpacerPlacer* will allow future studies to identify potential differences in deletion patterns. For example, when comparing the dynamics of bacterial and archaeal CRISPR systems, systems with and without priming, or arrays based on plasmids and chromosomes. For this, a larger database with a stronger representation of further CRISPR types, plasmids, and archaeal genomes is necessary. *SpacerPlacer* is a rapid and versatile tool for studying CRISPR arrays and systems, and we expect it to provide valuable insights in a variety of different application scenarios.

For example, here we inferred that the deletion rate for CRISPR spacers, $\rho_B \cdot \alpha = 374$, is two orders of magnitude higher than the per-site mutation rate. This evolutionary speed and the robustness of complete spacers with regard to sequencing errors make CRISPR arrays a valuable fine-scale typing tool [18], as has already been demonstrated for multiple species [16, 19, 56, 57].

When tracking the evolution of clinically relevant strains and investigating outbreaks, it can be beneficial not only to distinguish different types but also to reconstruct their evolutionary relationship. While standard MLST typing is often insufficient to reconstruct the evolutionary history of closely related strains, CRISPR arrays are rich in phylogenetic information due to the PSIO. This concept was also discussed by Tomida et al. [13], who estimated phylogenetic trees based on a standard pairwise distance between CRISPR arrays, which provided a higher resolution than the trees generated using MLST. The currently implemented basic tree inference approach combined with the block deletion model in *SpacerPlacer* already solves some of the issues found in the CRISPR based trees of Tomida et al. (Supplementary Note 5A). We anticipate that incorporation of more advanced tree inference methods will further improve the results.

Monitoring bacterial strains and their co-evolution with phages based on sequenced CRISPR arrays is also of importance for bacteria-driven biotechnological and pharmaceutical production processes, such as bioreactors [22] or wastewater treatment plans [23]. Both studies used metagenome assembled genomes (MAGs) and extracted spacer information from them that does not necessarily include the order of spacers within the array.

Partial CRISPR arrays derived from short-read sequencing and MAGs currently limit the power of CRISPR arrays as high-resolution markers [58] and the reconstruction of population histories from CRISPR arrays, e.g. in time series of metagenomes [21–23]. If the complete spacer array is unknown, analyzing incomplete segments of CRISPR arrays can lead to inaccurate reconstructions. However, as an alternative to whole genome or long-read sequencing, complete spacer arrays can also be obtained from short reads in some scenarios [57].

If complete arrays are available, *SpacerPlacer* can help investigate the dynamics of microbiomes and the ecology of bacteria and archaea by exploring the correspondence between spacers and invading elements within the reconstructions.

For example, Lam and Ye characterize CRISPR arrays from a gut microbiome sequenced with long-read sequencing technologies [20]. Through the use of spacer alignments and spacer graphs, they analyzed the dynamics of spacer acquisition within their datasets. *SpacerPlacer* can infer the ancestral relationships for such data sets and reconstruct the corresponding spacer insertion and deletion events (Supplementary Note 5B).

In conclusion, *SpacerPlacer* is the first parametric ancestral reconstruction tool that enables in-depth investigations of CRISPR array evolution. Analyzing spacer arrays of closely related strains will allow us to shed light on the coevolution of phages and bacteria on a fine-scale, the mechanisms of spacer acquisition and deletion itself, and the dynamic ecology of microbial communities and microbiomes.

Data Availability

SpacerPlacer is available at `https://github.com/fbaumdicker/SpacerPlacer`.

ACKNOWLEDGEMENTS

We thank Shiraz Shah and Jaime Iranzo for their helpful comments and discussions. AF and FB are funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under the Priority Program - SPP 2141 - Project number Ba-5529/1-1. FB is funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under Germany's Excellence Strategy – EXC number 2064/1 – Project number 390727645, and EXC 2124 – Project number 390838134.

This work was supported by German Research Foundation (DFG) [BA 2168/23-1 and BA 2168/23-2 SPP 2141]; Much more than Defence: the Multiple Functions and Facets of CRISPR–Cas; Baden-Wuerttemberg Ministry of Science, Research and Art; University of Freiburg.

This work was supported by the BMBF-funded de.NBI Cloud within the German Network for Bioinformatics Infrastructure (de.NBI) (031A532B, 031A533A, 031A533B, 031A534A, 031A535A, 031A537A, 031A537B, 031A537C, 031A537D, 031A538A).

Bibliography

1. Hannah G. Hampton, Bridget N. J. Watson, and Peter C. Fineran. The arms race between bacteria and their phage foes. *Nature*, 577(7790):327–336, 2020. doi: 10.1038/s41586-019-1894-8.
2. Aude Bernheim and Rotem Sorek. The pan-immune system of bacteria: antiviral defence as a community resource. *Nature Reviews Microbiology*, 18(2):113–119, 2020. doi: 10.1038/s41579-019-0278-2.
3. Joy Y. Wang and Jennifer A. Doudna. CRISPR technology: A decade of genome editing is only the beginning. *Science*, 379(6629):eadd8643, 2023. doi: 10.1126/science.add8643.
4. Francisco J.M. Mojica, Chc)sar Díez-Villaseñor, Jesús García-Martínez, and Elena Soria. Intervening Sequences of Regularly Spaced Prokaryotic Repeats Derive from Foreign Genetic Elements. *Journal of Molecular Evolution*, 60(2):174–182, 2005. doi: 10.1007/s00239-004-0046-3.
5. Kira S. Makarova, Yuri I. Wolf, Omer S. Alkhnbashi, Fabrizio Costa, Shiraz A. Shah, Sita J. Saunders, Rodolphe Barrangou, Stan J. J. Brouns, Emmanuelle Charpentier, Daniel H. Haft, Philippe Horvath, Sylvain Moineau, Francisco J. M. Mojica, Rebecca M. Terns, Michael P. Terns, Malcolm F. White, Alexander F. Yakunin, Roger A. Garrett, John van der Oost, Rolf Backofen, and Eugene V. Koonin. An updated evolutionary classification of CRISPR-Cas systems. *Nature reviews. Microbiology*, 13(11):722–736, 2015. doi: 10.1038/nrmicro3569. ISBN: 1740-1534 (Electronic)\r1740-1526 (Linking).
6. Kira S. Makarova, Yuri I. Wolf, Jaime Iranzo, Sergey A. Shmakov, Omer S. Alkhnbashi, Stan J. J. Brouns, Emmanuelle Charpentier, David Cheng, Daniel H. Haft, Philippe Horvath, Sylvain Moineau, Francisco J. M. Mojica, David Scott, Shiraz A. Shah, Virginijus Siksnys, Michael P. Terns, Česlovas Venclovas, Malcolm F. White, Alexander F. Yakunin, Winston Yan, Feng Zhang, Roger A. Garrett, Rolf Backofen, John Van Der Oost, Rodolphe Barrangou, and Eugene V. Koonin. Evolutionary classification of CRISPR–Cas systems: a burst of class 2 and derived variants. *Nature Reviews Microbiology*, 18(2):67–83, 2020. doi: 10.1038/s41579-019-0299-x.
7. Rodolphe Barrangou, Christophe Fremaux, Hélène Deveau, Melissa Richards, Patrick Boyaval, Sylvain Moineau, Dennis A. Romero, and Philippe Horvath. CRISPR Provides Acquired Resistance Against Viruses in Prokaryotes. *Science*, 315(5819):1709–1712, 2007. doi: 10.1126/science.1138140.
8. Jon McGinn and Luciano A. Marraffini. Molecular mechanisms of CRISPR–Cas spacer acquisition. *Nature Reviews Microbiology*, 17(1):7–12, 2019. doi: 10.1038/s41579-018-0071-7. Number: 1 Publisher: Nature Publishing Group.
9. Philippe Horvath, Dennis A. Romero, Anne-Claire Coûté-Monvoisin, Melissa Richards, Hélène Deveau, Sylvain Moineau, Patrick Boyaval, Christophe Fremaux, and Rodolphe Barrangou. Diversity, Activity, and Evolution of CRISPR Loci in Streptococcus thermophilus. *Journal of Bacteriology*, 2008. doi: 10.1128/jb.01415-07.
10. Chitong Rao, Denny Chin, and Alexander W. Ensminger. Priming in a permissive type I-C CRISPR-Cas system reveals distinct dynamics of spacer acquisition and loss. *RNA (New York, N.Y.)*, 23(10):1525–1538, 2017. doi: 10.1261/rna.062083.117.
11. Sandra C. Garrett. Pruning and Tending Immune Memories: Spacer Dynamics in the CRISPR Array. *Frontiers in Microbiology*, 12:664299, 2021. doi: 10.3389/fmicb.2021.664299.
12. Hélène Deveau, Rodolphe Barrangou, Josiane E. Garneau, Jessica Labonté, Christophe Fremaux, Patrick Boyaval, Dennis A. Romero, Philippe Horvath, and Sylvain Moineau. Phage response to CRISPR-encoded resistance in Streptococcus thermophilus. *Journal of Bacteriology*, 190(4):1390–1400, 2008. doi: 10.1128/JB.01412-07.
13. Junko Tomida, Yuji Morita, Keigo Shibayama, Ken Kikuchi, Tomohiro Sawa, Takaaki Akaike, and Yoshiaki Kawamura. Diversity and microevolution of CRISPR loci in Helicobacter cinaedi. *PLOS ONE*, 12(10):e0186241, 2017. doi: 10.1371/journal.pone.0186241.
14. Nikki Shariat and Edward G. Dudley. CRISPRs: Molecular Signatures Used for Pathogen Subtyping. *Applied and Environmental Microbiology*, 80(2):430–439, 2014. doi: 10.1128/AEM.02790-13.
15. Tony J. Lam and Yuzhen Ye. CRISPRs for Strain Tracking and Their Application to Microbiota Transplantation Data Analysis. *The CRISPR Journal*, 2(1):41–50, 2019. doi: 10.1089/crispr.2018.0046.
16. Rodolphe Barrangou and Edward G. Dudley. CRISPR-Based Typing and Next-Generation Tracking Technologies. *Annual Review of Food Science and Technology*, 7(1):395–411, 2016. doi: 10.1146/annurev-food-022814-015729.
17. Edze R. Westra, Andrea J. Dowling, Jenny M. Broniewski, and Stineke van Houte. Evolution and Ecology of CRISPR. *Annual Review of Ecology, Evolution, and Systematics*, 47(1):307–331, 2016. doi: 10.1146/annurev-ecolsys-121415-032428.
18. P. Muzyukina and O. Soutourina. CRISPR genotyping methods: Tracing the evolution from spoligotyping to machine learning. *Biochimie*, 217:66–73, 2024. doi: 10.1016/j.biochi.2023.07.017.
19. Zahra Karimi, Ali Ahmadi, Ali Najafi, and Reza Ranjbar. Bacterial CRISPR Regions: General Features and their Potential for Epidemiological Molecular Typing Studies. *The Open Microbiology Journal*, 12:59–70, 2018. doi: 10.2174/1874285801812010059.
20. Tony J. Lam and Yuzhen Ye. Long reads reveal the diversification and dynamics of CRISPR reservoir in microbiomes. *BMC Genomics*, 20(1):567, 2019. doi: 10.1186/s12864-019-5922-8.
21. Christine L. Sun, Brian C. Thomas, Rodolphe Barrangou, and Jillian F. Banfield. Metagenomic reconstructions of bacterial CRISPR loci constrain population histories. *The ISME Journal*, 10(4):858–870, 2016. doi: 10.1038/ismej.2015.162.
22. Susana Martínez Arbas, Shaman Narayanasamy, Malte Herold, Laura A. Lebrun, Michael R. Hoopmann, Sujun Li, Tony J. Lam, Benoît J. Kunath, Nathan D. Hicks, Cindy M. Liu, Lance B. Price, Cedric C. Laczny, John D. Gillece, James M. Schupp, Paul S. Keim, Robert L. Moritz, Karoline Faust, Haixu Tang, Yuzhen Ye, Alexander Skupin, Patrick May, Emilie E. L. Muller, and Paul Wilmes. Roles of bacteriophages, plasmids and CRISPR immunity in microbial community dynamics revealed using time-series integrated meta-omics. *Nature Microbiology*, 6(1):123–135, 2021. doi: 10.1038/s41564-020-00794-8.
23. Leandro D. Guerrero, María V. Pérez, Esteban Orellana, Mariana Piuri, Cecilia Quiroga, and Leonardo Erijman. Long-run bacteria-phage coexistence dynamics under natural habitat conditions in an environmental biotechnology system. *The ISME Journal*, 15(3):636–648, 2021. doi: 10.1038/s41396-020-00802-z.
24. Moïra Dion, Simon Labrie, Shiraz Shah, and Sylvain Moineau. CRISPRStudio: A User-Friendly Software for Rapid CRISPR Array Visualization. *Viruses*, 10(11):602, 2018. doi: 10.3390/v10110602.
25. Matthew A. Nethery and Rodolphe Barrangou. CRISPR Visualizer: rapid identification and visualization of CRISPR loci via an automated high-throughput processing pipeline. *RNA Biology*, 16(4):577–584, 2019. doi: 10.1080/15476286.2018.1493332.
26. Lei Sun, Jinbiao Wang, Fu Yan, Gongming Wang, Yun Li, and Jinlin Huang. CrisprVi: a software for visualizing and analyzing CRISPR sequences of prokaryotes. *BMC Bioinformatics*, 23(S3):172, 2022. doi: 10.1186/s12859-022-04716-9.
27. Alan J. Collins and Rachel J. Whitaker. CRISPR comparison toolkit (CCTK): Rapid identification, visualization, and analysis of CRISPR array diversity. preprint, Bioinformatics, 2022.
28. Christine Pourcel, Marie Touchon, Nicolas Villeriot, Jean-Philippe Vernadet, David Couvin, Claire Toffano-Nioche, and Gilles Vergnaud. CRISPRCasdb a successor of CRISPRdb containing CRISPR arrays and cas genes from complete genome sequences, and tools to download and query lists of repeats and spacers. *Nucleic Acids Research*, gkz915, 2019. doi: 10.1093/nar/gkz915.
29. Hanrong Chen, Andreas Mayer, and Vijay Balasubramanian. A scaling law in CRISPR repertoire sizes arises from the avoidance of autoimmunity. *Current Biology*, 32(13):2897–2907.e5, 2022. doi: 10.1016/j.cub.2022.05.021.
30. David Couvin, Aude Bernheim, Claire Toffano-Nioche, Marie Touchon, Juraj Michalik, Bertrand Néron, Eduardo P C Rocha, Gilles Vergnaud, Daniel Gautheret, and Christine Pourcel. CRISPRCasFinder, an update of CRISRFinder, includes a portable version, enhanced performance and integrates search for Cas proteins. *Nucleic Acids Research*, 46 (W1):W246–W251, 2018. doi: 10.1093/nar/gky425.
31. Oliver Schwengers, Lukas Jelonek, Marius Alfred Dieckmann, Sebastian Beyvers, Jochen Blom, and Alexander Goesmann. Bakta: rapid and standardized annotation of bacterial genomes via alignment-free sequence identification. *Microbial genomics*, 7(11), 2021.
32. Wei Ding, Franz Baumdicker, and Richard A Neher. panX: pan-genome analysis and exploration. *Nucleic Acids Research*, 46(1):e5, 2018.
33. Lam-Tung Nguyen, Heiko A. Schmidt, Arndt Von Haeseler, and Bui Quang Minh. IQ-TREE: A Fast and Effective Stochastic Algorithm for Estimating Maximum-Likelihood Phylogenies. *Molecular Biology and Evolution*, 32(1):268–274, 2015. doi: 10.1093/molbev/msu300.
34. Jl. Weissman, William F. Fagan, and Philip L.F. Johnson. Selective Maintenance of Multiple CRISPR Arrays Across Prokaryotes. *The CRISPR Journal*, 1(6):405–413, 2018. doi: 10.1089/crispr.2018.0034.
35. Anne Kupczok and Jonathan P Bollback. Probabilistic models for CRISPR spacer content evolution. *BMC Evolutionary Biology*, 13(1):54, 2013. doi: 10.1186/1471-2148-13-54.
36. F. Baumdicker, A.M.I. Huebner, and P. Pfaffelhuber. The independent loss model with ordered insertions for the evolution of CRISPR spacers. *Theoretical Population Biology*, 119:72–82, 2018. doi: 10.1016/j.tpb.2017.11.001.
37. Maria-José Lopez-Sanchez, Elisabeth Sauvage, Violette Da Cunha, Dominique Clermont, Elisoa Ratsima Hariniaina, Bruno Gonzalez-Zorn, Claire Poyart, Isabelle Rosinski-Chupin, and Philippe Glaser. The highly dynamic CRISPR1 system of *Streptococcus agalactiae* controls the diversity of its mobilome: *Streptococcus agalactiae* CRISPR dynamics. *Molecular Microbiology*, 85(6):1057–1071, 2012. doi: 10.1111/j.1365-2958.2012.08172.x.
38. Victor A Padilha, Omer S Alkhnbashi, Shiraz A Shah, André C P L F de Carvalho, and Rolf Backofen. CRISPRcasIdentifier: Machine learning for accurate identification and classification of CRISPR-Cas systems. *GigaScience*, 9(6):giaa062, 2020. doi: 10.1093/gigascience/giaa062.
39. Susan T. Lovett. Encoded errors: mutations and rearrangements mediated by misalignment at repetitive DNA sequences. *Molecular Microbiology*, 52(5):1243–1253, 2004. doi: 10.1111/j.1365-2958.2004.04076.x.
40. Todd J. Treangen, Anne-Laure Abraham, Marie Touchon, and Eduardo P.C. Rocha. Genesis, effects and fates of repeats in prokaryotic genomes. *FEMS Microbiology Reviews*, 33(3):539–571, 2009. doi: 10.1111/j.1574-6976.2009.00169.x.
41. Malgorzata Bzymek and Susan T. Lovett. Instability of repetitive DNA sequences: The role of replication in multiple mechanisms. *Proceedings of the National Academy of Sciences*, 98 (15):8319–8325, 2001. doi: 10.1073/pnas.111008398.
42. Pu Han, Liang Ren Niestemski, Jeffrey E Barrick, and Michael W Deem. Physical model of the immune response of bacteria against bacteriophage through the adaptive CRISPR-Cas immune system. *Physical Biology*, 10(2):025004, 2013. doi: 10.1088/1478-3975/10/2/025004.
43. Eugene V. Koonin and Yuri I. Wolf. Evolution of the CRISPR-Cas adaptive immunity systems in prokaryotes: models and observations on virus–host coevolution. *Molecular BioSystems*, 11(1):20–27, 2015. doi: 10.1039/C4MB00438H.
44. Alexander Martynov, Konstantin Severinov, and Iaroslav Ispolatov. Optimal number of spacers in CRISPR arrays. *PLOS Computational Biology*, 13(12):e1005891, 2017. doi: 10.1371/journal.pcbi.1005891.
45. Yekaterina S. Pavlova, David Paez-Espino, Andrew Yu. Morozov, and Ilya S. Belalov. Searching for fat tails in CRISPR-Cas systems: Data analysis and mathematical modeling. *PLOS Computational Biology*, 17(3):e1008841, 2021. doi: 10.1371/journal.pcbi.1008841.
46. Kazutaka Katoh, John Rozewicki, and Kazunori D Yamada. MAFFT online service: multiple sequence alignment, interactive sequence choice and visualization. *Briefings in Bioinformatics*, 20(4):1160–1166, 2019. doi: 10.1093/bib/bbx108.
47. Rodrigo Achigar, Martina Scarrone, Geneviève M. Rousseau, Cécile Philippe, Felipe Machado, Valentina Duvós, María Pía Campot, Moïra B. Dion, Yuyu Shao, María Julia Pianzzola, and Sylvain Moineau. Ectopic Spacer Acquisition in Streptococcus thermophilus CRISPR3 Array. *Microorganisms*, 9(3):512, 2021. doi: 10.3390/microorganisms9030512.
48. Rodrigo Achigar, Alfonso H. Magadán, Denise M. Tremblay, María Julia Pianzzola, and Sylvain Moineau. Phage-host interactions in Streptococcus thermophilus: Genome analysis of phages isolated in Uruguay and ectopic spacer acquisition in CRISPR array. *Scientific Reports*, 7(1):43438, 2017. doi: 10.1038/srep43438.
49. Cas Mosterd and Sylvain Moineau. Characterization of a Type II-A CRISPR-Cas System in *Streptococcus mutans*. *mSphere*, 5(3):e00235–20, 2020. doi: 10.1128/mSphere.00235-20.
50. Tal Pupko, Itsik Pe, Ron Shamir, and Dan Graur. A Fast Algorithm for Joint Reconstruction of Ancestral Amino Acid Sequences. *Molecular Biology and Evolution*, 17(6):890–896, 2000.

doi: 10.1093/oxfordjournals.molbev.a026369.
51. Pavel Sagulenko, Vadim Puller, and Richard A Neher. TreeTime: Maximum-likelihood phylodynamic analysis. *Virus Evolution*, 4(1), 2018. doi: 10.1093/ve/vex042.
52. J Felsenstein. Phylogenies from Restriction Sites: A Maximum-Likelihood Approach. *Evolution*, 46:159–173, 1992.
53. Anne Kupczok, Giddy Landan, and Tal Dagan. The Contribution of Genetic Recombination to CRISPR Array Evolution. *Genome Biology and Evolution*, 7(7):1925–1939, 2015. doi: 10.1093/gbe/evv113.
54. Alice Toms and Rodolphe Barrangou. On the global CRISPR array behavior in class I systems. *Biology Direct*, 12(1):20, 2017. doi: 10.1186/s13062-017-0193-2.
55. Adrián López-Beltrán, João Botelho, and Jaime Iranzo. Dynamics of CRISPR-mediated virus-host interactions in the human gut microbiome. *bioRxiv*, 10.1101/2024.01.23.576851, 2024. doi: 10.1101/2024.01.23.576851.
56. Nabil Karah, Ørjan Samuelsen, Raffaele Zarrilli, Jason W. Sahl, Sun Nyunt Wai, and Bernt Eric Uhlin. CRISPR-cas Subtype I-Fb in Acinetobacter baumannii: Evolution and Utilization for Strain Subtyping. *PLOS ONE*, 10(2):e0118205, 2015. doi: 10.1371/journal.pone.0118205. Publisher: Public Library of Science.
57. Anton Rykachevsky, Alexander Stepakov, Polina Muzyukina, Sofia Medvedeva, Mark Dobrovolski, Evgeny Burnaev, Konstantin Severinov, and Ekaterina Savitskaya. SCRAMBLER: A Tool for De Novo CRISPR Array Reconstruction and Its Application for Analysis of the Structure of Prokaryotic Populations. *The CRISPR Journal*, 4(5):673–685, 2021. doi: 10.1089/crispr.2021.0012.
58. Lingdi Zhang, Jahan Rahman, Matthew Chung, Lauren Lashua, Aubree Gordon, Angel Balmaseda, Guillermina Kuan, Richard Bonneau, and Elodie Ghedin. CRISPR arrays as high-resolution markers to track microbial transmission during influenza infection. *Microbiome*, 11(1):136, 2023. doi: 10.1186/s40168-023-01568-0.