Supplemental Material

Anne Kupczok, Giddy Landan, Tal Dagan

June 2, 2015

Supplementary Material and Methods

Here the details of the algorithm are described.

The spacer graph

Given a list of arrays $a_1, ..., a_n$ with S being the set of different spacers, a_i^j is the j-th spacer from the i-th array, spacer index 1 is proximal to the leader.

The spacer graph has nodes S and an edge connecting s_k and s_l if $a_i^j = s_k \wedge a_i^{j+1} = s_l$ for any i, j. Nodes s and edges e have binarily encoded strain labels, $l(v)$ and $l(e)$. The i-th bit is set in $l(s)$ if $s \in a_i$ and the i-th bit is set in $l(e)$ if $e = (s_k, s_l)$ and $a_i^j = s_k \wedge a_i^{j+1} = s_l$ for any j. We say an array a_i contains an edge $e = (s_k, s_l)$ if $a_i^j = s_k \wedge a_i^{j+1} = s_l$ for any j.

p is a function returning whether two spacers are connected. $p(s_i, s_j) = 1$, iff there is a directed path from s_i to s_j. $p^b(s_i, s_j) = 1$ iff there is a directed path from s_i to s_j where only edges e with $b\&l(e)!=0$ are traversed (i.e., only edges originating from particular strains are allowed to be used). $\hat{P}(s_i, s_j) = 1$ iff there is a directed path from s_i to s_j that is only traversing edges from one array.

The set of preceding spacers is $P(s) = \{t : p(t, s) = 1\}$, $P^b(s) = \{t : p^b(t, s) = 1\}$, $\hat{P}(s) = \{t : \hat{P}(t, s) = 1\}$. The set of successive spacers is $N(s) = \{t : p(s, t) = 1\}$, $N^b(s) = \{t : p^b(s, t) = 1\}$, $\hat{N}(s) = \{t : \hat{P}(s, t) = 1\}$.

Replicated spacers

If $s = a_i^j \wedge s = a_i^k$ for any i and $j \neq k$, then s is a replicated spacer.

Inversions

Inverted spacer orderings (short: inversions) preclude a common order of all spacers and introduce loops in the spacer graph. Decisions have to be made which spacers are eliminated to resolve a particular loop, these are assigned to be involved in inversions. Here the parsimonious decision of always taking the shortest segment as an inversion is taken. If two segments of equal length would resolve the loop, both are assigned as inversions. The following algorithm detects a set of spacers L, that are involved in inversions. If these are eliminated from the arrays, the resulting spacer graph has no loops.

1. Spacers $L = \{\}$.
2. $\hat{P}(s) = P^*(s)$, $\hat{N}(s) = N^*(s)$ for all s.
3. For every spacer, calculate the intersection spacers: $\hat{I}(s) = \hat{P}(s) \cap \hat{N}(s)$.
4. $\hat{I}_m = \max |\hat{I}(s)|$ (the highest number of intersection spacers for a spacer)
5. If $\hat{I}_m = 0$:
   If spacers can be ordered such that $\forall_e\, e = (s_i, s_j) \rightarrow i < j$: return L.
   Else: go to 9.
6. $L = L \cup \{s : |\hat{I}(s)| = \hat{I}_m\}$.
7. For each s: $\hat{I}(s) = \hat{I}(s) \setminus L$
8. Continue with step 4.
9. Expand P and N to include more paths: $\hat{P}(s) = \hat{P}(s) \cup \bigcup_{t \in \hat{P}(s)} \hat{P}(t)$ and analogously for $\hat{N}(s)$.
   (In words, before we were looking at preceeding and successive spacers in one array, thus inversions that are caused by the information of two arrays togeter. Now, preceeding and successive spacers are formed by two arrays each resulting in inversions that are caused by the information in at most four arrays.)
   Continue with step 3.

Order divergence events

A spacer graph is built from a data set without inversions.

1. $F = \{\}$

2. For each $s \in S$:

   (a) Find the set of non-redundant successive spacers $N_{\text{nr}}(s) = N^{l(s)}(s) \setminus N_{\text{r}}(s)$, with $N_{\text{r}}(s) = \{t \in N^{l(s)}(s) : \exists_{f \in N^{l(s)}(s)} p^{l(s)}(f,t) = 1\}$ (i.e., a spacer t is redundant, if there is a path from another f to t).

   (b) If $|N_{\text{nr}}(s)| \leq 1$: continue with 2. Else generate a new divergence event (H, D) with shared spacers H and different spacers D, these sets will be filled as follows.

   (c) Find the set of spacers that can be reached by all $f \in N_{\text{nr}}(s)$: $R(s) = \{t \in N^{l(s)}(s) : \forall_{f \in N_{\text{nr}}(s)} p^{l(s)}(f,t)\}$

   (d) Find the closest spacers that can be reached by all. For each $r \in R(s)$: $score(r) = \sum_{n \in N_{\text{nr}}(s)} dist(n,r)$ where $dist(n,r)$ is the minimum number of edges along a path from n to r only traversing edges from $l(s) \& l(n)$. $score_{\text{min}} = \min_{r \in R(s)} score(r)$. $R_{\text{min}} = \{r : score(r) = score_{\text{min}}\}$.

   (e) Find the different segments, these are the sets of spacers between between s and R_{min} : For each t with an edge $e = (s,t)$: $D(t) = \bigcup_{r \in R_{\text{min}}(s)} N^{l(s)}(t) \cap P^{l(s)}(r)$; $D(t) = D(t) \setminus R_{\text{min}}(s)$. Merge $D(t)$ such that the sets are non-overlapping. These non-overlapping sets are D.

   (f) Find the beginning of the shared segment

      i. Mask s from the data set and consider only the edges in $l(s)$ now.

      ii. Evaluate all b with an $e = (b,s)$: build $N_{\text{nr}}(b)$;

      iii. If at least two sets from $\{N^{l(s)}(c) : c \in N_{\text{nr}}(b)\}$ are intersecting with at least two different elements from D, then:
      add b to H, mask b and continue with (ii) by evaluating all $e = (c,b)$

   (g) $F = F \cup \{(H, D)\}$

   (h) If $|N_{\text{nr}}(s)| > 2$: Check each subset of $N_{\text{nr}}(s)$, if the set of spacers that can be reached by them contains no element of $R(s)$. In this case, generate a new event (H, D) of the subset of taxa and calculate D and H analogous to (c)-(g).

Supplementary Tables

Length	Strains	Hit	Description	Range	E-value
E. coli CRISPR1.1					
1293	3	pfam00665	Integrase core domain	173-523	1.02e-31
503	341	WP_001342819	membrane protein	196-273	2e-5
1763	1	PHA02517	putative transposase OrfB	487-1287	1.10e-67
1382	2	pfam01609	Transposase DDE domain	280-1005	2.70e-15
1280	1	pfam03400	IS1 transposase	128-520	3.81e-84
564	1	WP_001430068	membrane protein	257-334	5e-6
E. coli CRISPR1.2					
1381	3	pfam01609	Transposase DDE domain	394-1119	4.23e-20
P. aeruginosa					
1272	2	pfam13683	Integrase core domain	938-1138	5.43e-34
1271	1	PHA02517	putative transposase OrfB	442-1215	1.12e-97
1271	1	PHA02517	putative transposase OrfB	39-812	1.01e-95
394	1	WP_023115125	hypothetical protein	67-207,201-359	2e-27
1271	1	PHA02517	putative transposase OrfB	431-1204	1.03e-95
S. thermophilus					
847	1	COG3316	Transposase and inactivated derivatives	70-720	9.76e-105
538	1	EWM59576	transposase	176-367	6e-35

Table S4: Protein hits of spacers longer than 100 nucleotides. If a conserved domain is detected (Marchler-Bauer et al. 2011), it is given as hit, otherwise the best hit with blastx (Altschul et al. 1997) is listed. Number of strains are given for the unique strains with loops.

Supplemental Figures

Figure S1: Recombination scenarios for power analysis. Breakpoints are marked by ||.

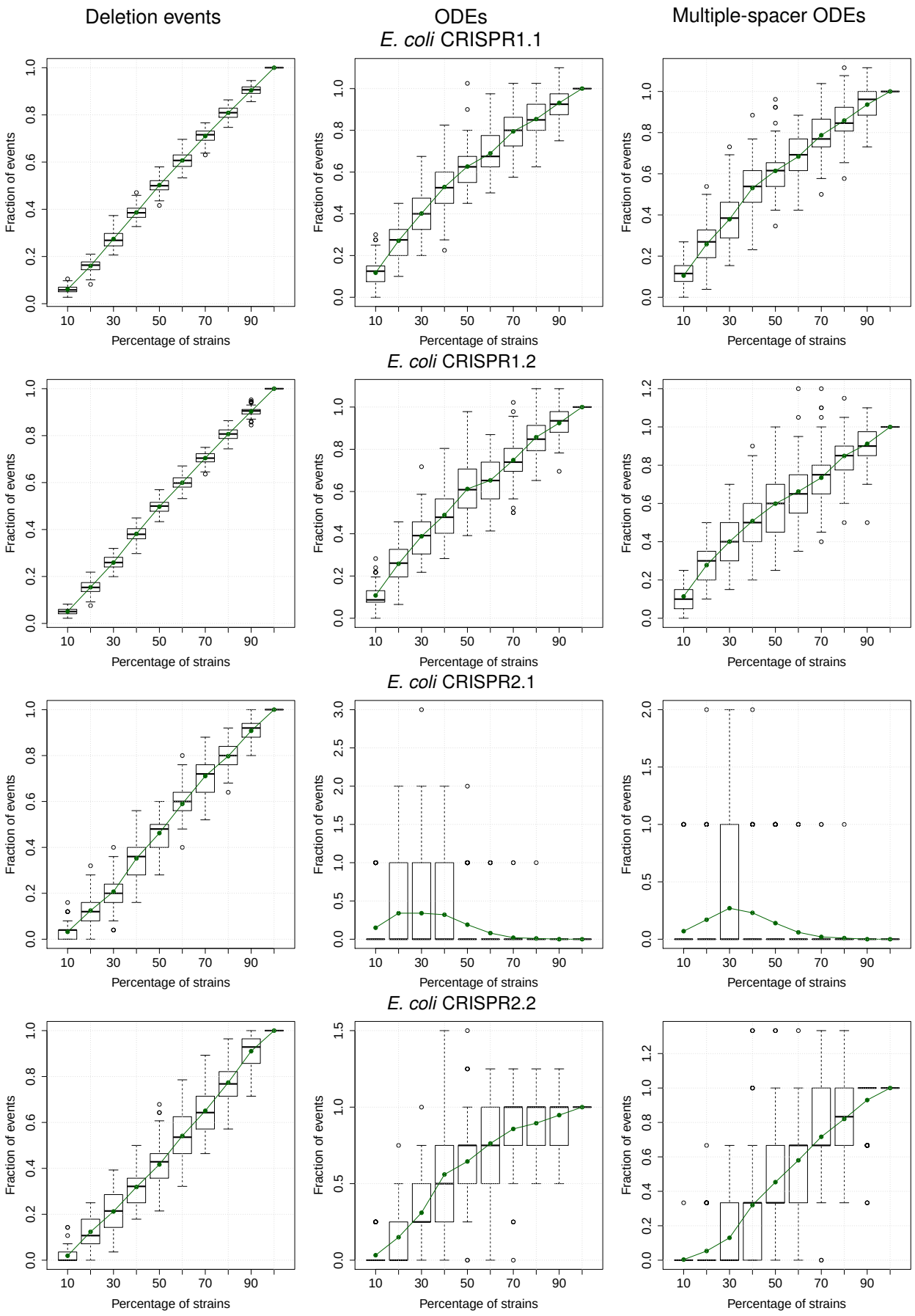
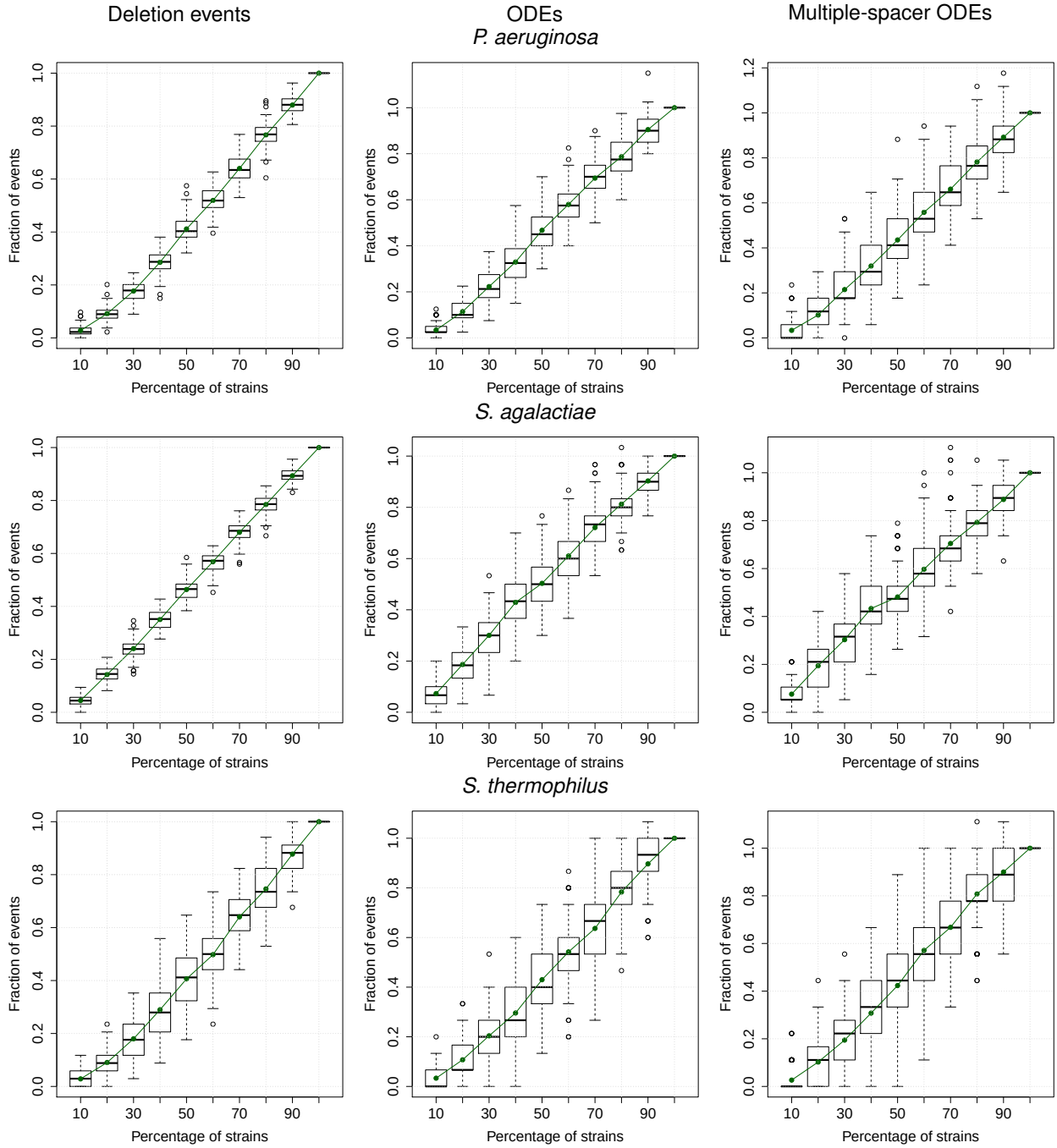


Figure S2: Robustness analysis. See also legend in Figure 4.

Figure S2 continued



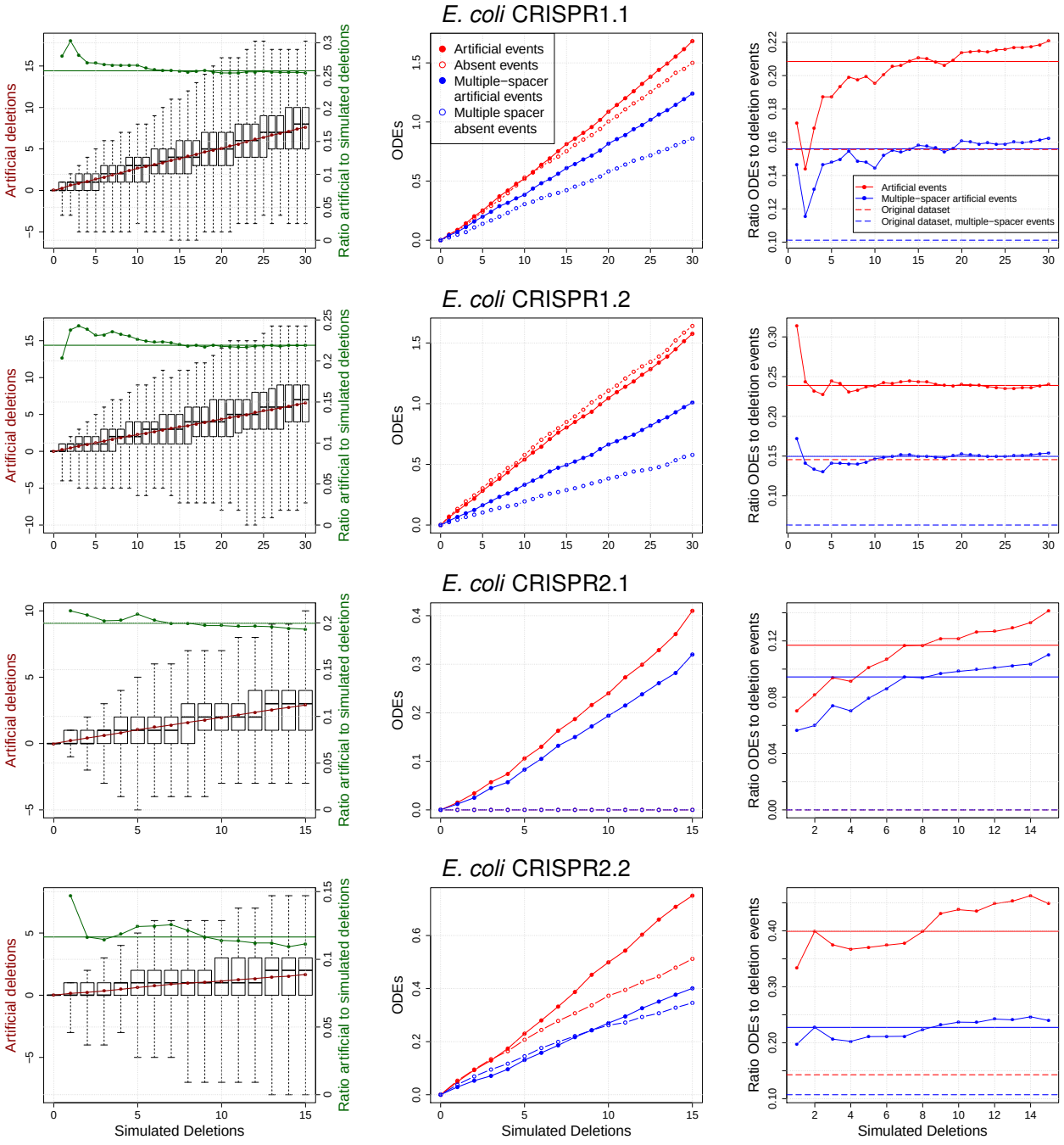


Figure S3: Estimation of deletion effects from 1000 perturbed replicates for the *E. coli* datasets. See also legend in Figure 6.

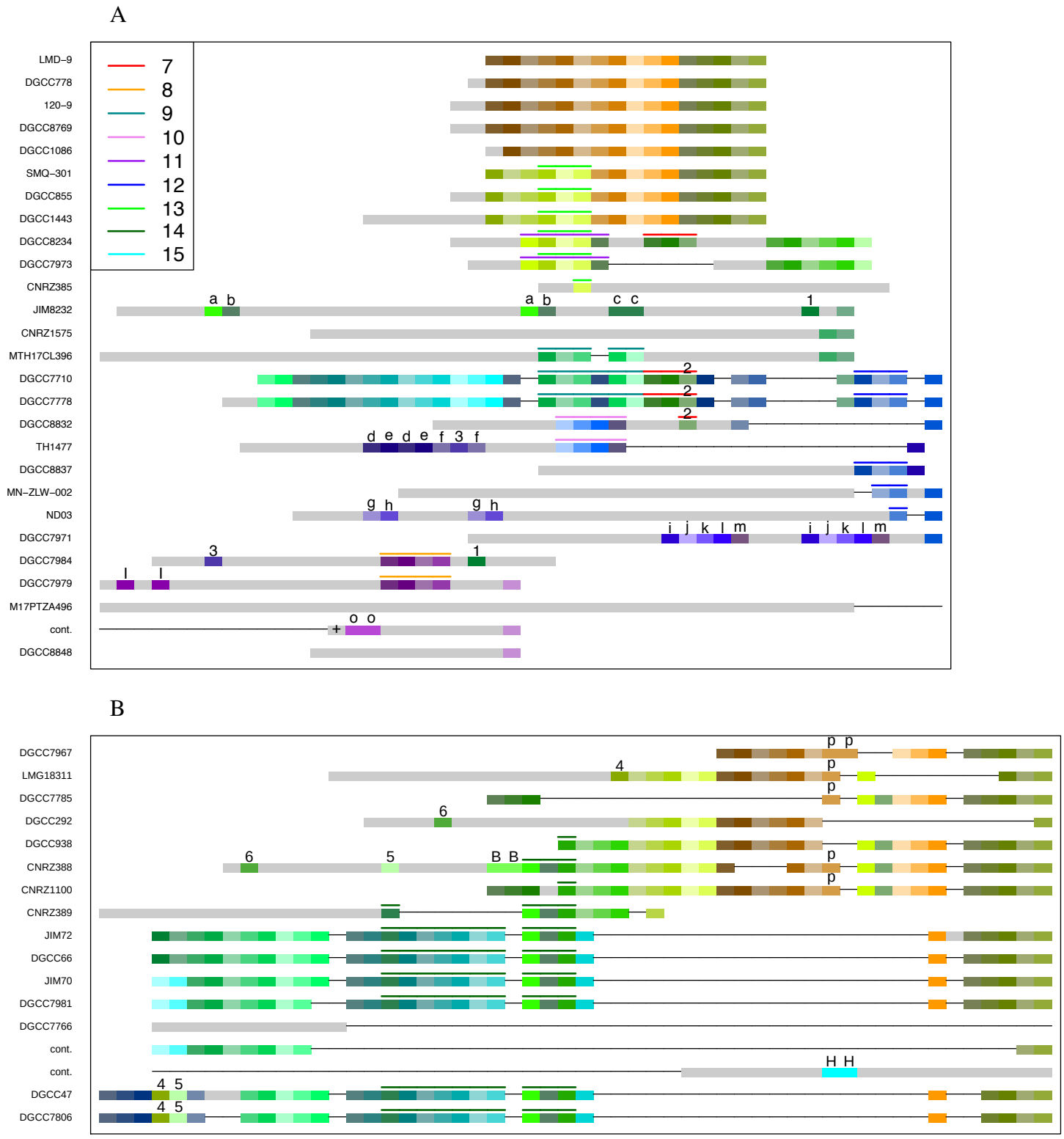
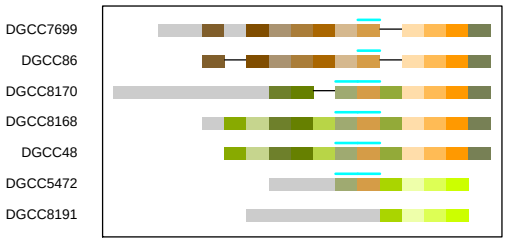


Figure S4: *S. thermophilus* CRISPR locus (continued on next page). A-H - connected components; I - singleton arrays. Leader-end is displayed on the left. Spacers are coded by colors. Unique spacers are shaded in gray. Spacers detected as order inversions are marked by Latin letters. Detected single-spacer ODEs are marked by Arabic numerals placed at the shared segment (1-6). Detected multiple-spacer ODEs are marked by a solid line placed above the shared segment (7-13). Arrays spanning multiple lines are marked by cont. (continued). Spacers longer than 100 nt are marked by "+".

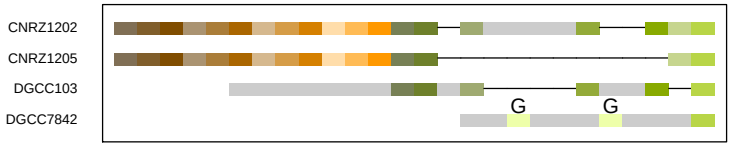
Figure S4 continued

C

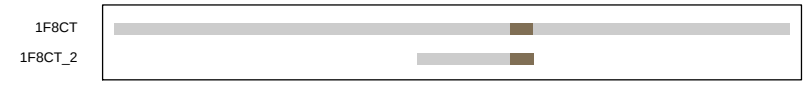


D

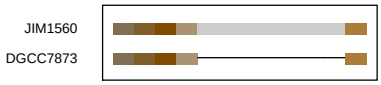
E



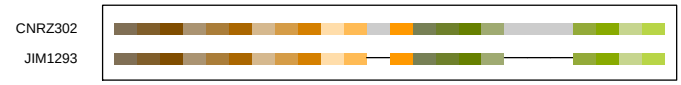
F



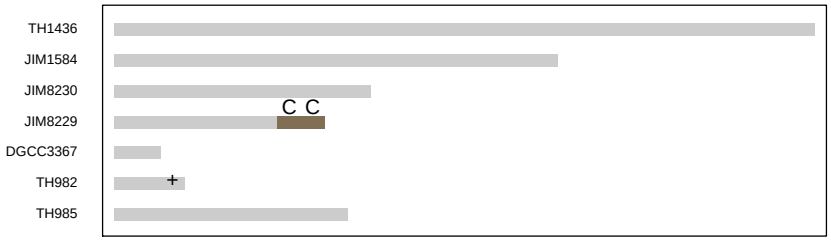
G



H



I



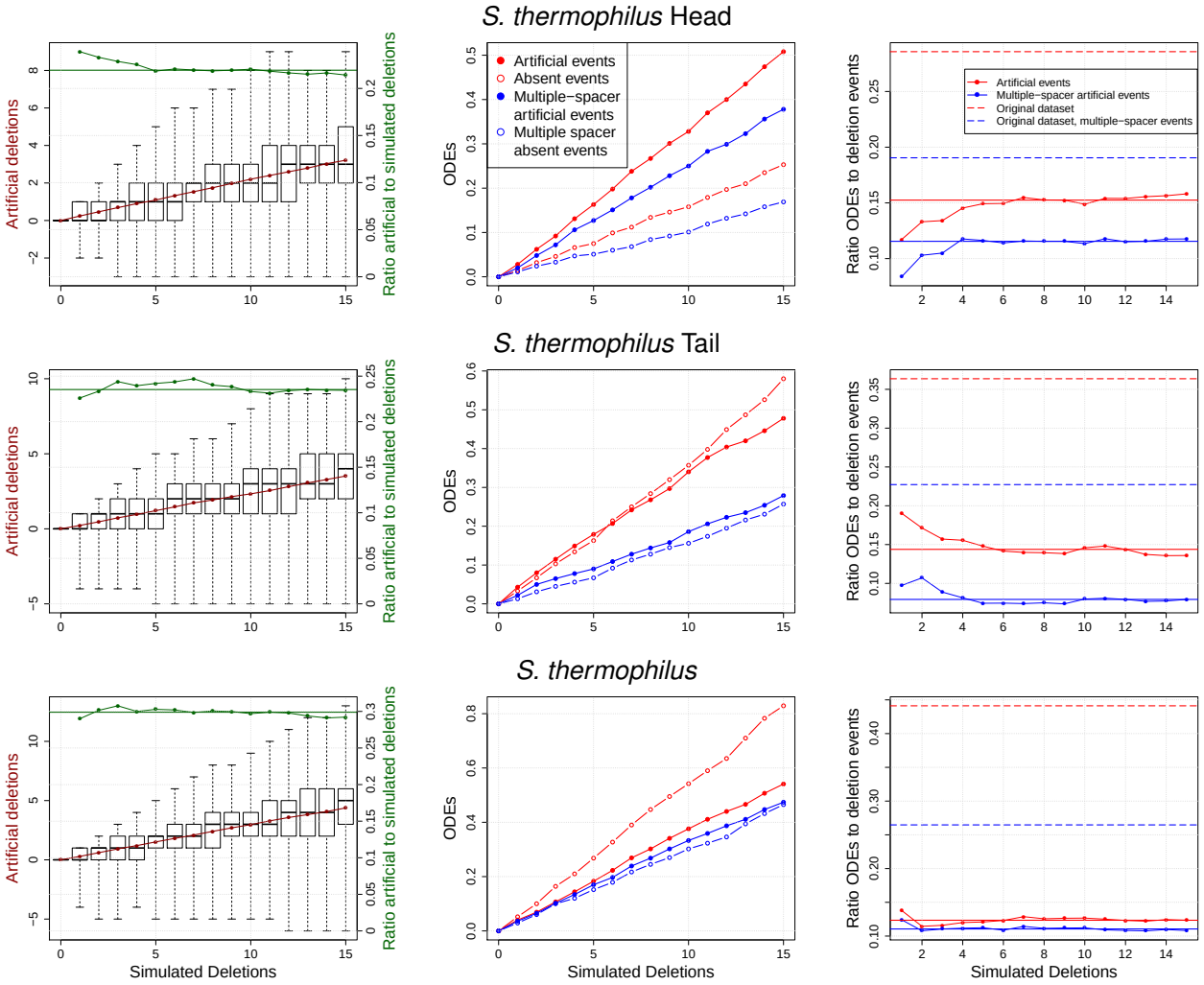


Figure S5: Estimation of deletion effects from 1000 perturbed replicates for *S. thermophilus*. See also legend in Figure 6.

References

Altschul SF, Madden TL, Schäffer AA, Zhang J, Zhang Z, Miller W, Lipman DJ, 1997. Gapped BLAST and PSI-BLAST: a new generation of protein database search programs. *Nucleic Acids Res.* 25:3389–402.

Marchler-Bauer A, Lu S, Anderson JB, Chitsaz F, Derbyshire MK, DeWeese-Scott C, Fong JH, Geer LY, Geer RC, Gonzales NR, et al., 2011. CDD: a Conserved Domain Database for the functional annotation of proteins. *Nucleic Acids Res.* 39:D225–9.