Inference of recombination breakpoints using STARRInIGHTS

B. Jesse Shapiro
jesse1@mit.edu

*Based on methods described in "**Population Genomics of Early Events in the Ecological Differentiation of Bacteria**" by B. Jesse Shapiro, Jonathan Friedman, Otto X. Cordero, Sarah P. Preheim, Sonia C. Timberlake, Gitta Szabó, Martin F. Polz, Eric J. Alm.*

Strain-based **T**ree **A**nalysis and **R**ecombinant **R**egion **In**ference **I**n **G**enomes from **H**igh-**T**hroughput **S**equencing-projects (STARRInIGHTS; software and documentation available at http://almlab.mit.edu/star/) combines aspects of the two major classes of methods to detect homologous recombination events in microbial genomes: 'substitution distribution methods' (*e.g.* ClonalFrame *(1)*) and 'phylogenetic methods' (*e.g. (2, 3)*). Like substitution distribution methods, STARRInIGHTS allows the mutation rate to vary along the genome, allowing for the detection of recombination events that import a large number of substitutions simultaneously into a stretch of the genome. Like phylogenetic methods, STARRInIGHTS also explicitly models the tree topology separately for different parts of the genome, allowing for detection of recombination events that change the tree topology without necessarily importing a large number of new substitutions.

The input to STARRInIGHTS is an aligned contig (or set of contigs) of genomic sequences, from which recombination breakpoints and relative rates of mutation and recombination are inferred. This also enables estimation of recombination events using phylogenetically informative single nucleotide polymorphisms (SNPs) in the aligned contigs – in this case, the 'core' genome described above. We propose that the core genome, consisting of G contigs each of length L_G bp, can be divided into B+1 blocks divided by B recombination breakpoints. Due to recombination between blocks, each block may have its own phylogeny and substitution rates (branch lengths). We assume each block has evolved according to its maximum-likelihood (ML) phylogeny. ML trees were inferred using phyML v. 2.4.5 *(4)* with a JC69 substitution model, a BIONJ starting tree, and two gamma-distributed evolutionary rate categories. The gamma distribution shape parameter was set to 0.03 for *Vibrio* data from a forthcoming manuscript (also 0.03 for *Salmonella enterica* serovar Typhi), the median value estimated by phyML in 5 kb windows along the core genome (the window size was varied from 500 bp to 20 kb without affecting the estimate). An example of the STARRInIGHTS procedure is shown in fig. S8. Due to the high sequencing coverage, we do not expect a considerable amount of sequencing error. Any remaining errors are not expected to introduce spurious breakpoints but rather to be accounted for by an increased local mutation rate. The only way that sequence errors could cause spurious breakpoints is if they introduced spatially clustered groups of polymorphisms all supporting a single phylogeny, an unlikely scenario to occur by chance.

To find the optimal number of breakpoints (B) and their locations in the genome, we define a cost function C, where both mutation events (on an ML tree within a block) and recombination breakpoints between blocks contribute to the cost incurred by a stretch of DNA from base i to base j ($i \leq j$).

$$
\begin{aligned}
C(i,j) = {} & c_b \cdot b_{ij} \\
& + \; c_{nb} \cdot (l_{ij} - b_{ij}) \\
& + \; c_{Tree(i,j)}
\end{aligned}
\qquad \text{Equation 1}
$$

where l_{ij} is the length, in base pairs, from i to j, c_b is the per-site cost of adding a breakpoint, c_{nb} is the cost for not adding a breakpoint, b_{ij} is the number of breakpoints between i and j, and $c_{Tree(i,j)}$ is the cost of the ML tree topology and branch lengths (mutation events) estimated for the alignment between i and j. The costs are in fact negative log probabilities:

$$
\begin{aligned}
c_b & = -\log P(b) \\
c_{nb} & = -\log(1 - P(b)) \\
c_{Tree(i,j)} & = -\log P(\tau, \nu, \theta \mid A_{i,j})
\end{aligned}
\qquad \text{Equations 2a-d}
$$

where $P(b)$ is the probability of a breakpoint and $P(\tau,\nu,\theta|A_{i,j})$ is the probability, estimated by phyML, of the tree topology τ, branch lengths ν and substitution model θ given the alignment A from i to j. Note that breakpoints can separate blocks with arbitrarily different tree topologies, requiring at least one recombination event, but potentially more. The number of inferred recombination events is therefore a lower bound.

We then minimize C over the whole genome using the dynamic programming recursion:

$$
M_j = \underset{\{i,j;\ 1 \le i \le j\}}{\min}\big(M_{i-1} + C(i,j)\big) \qquad \text{Equation 3}
$$

where M_j is the minimum cost for the first j bp, and setting $M_0 = 0$.

The number of breakpoints (B) and their locations will depend on the value of $P(b)$. The probability of a breakpoint, $P(b)$, is estimated from the data using expectation maximization (E-M) *(5)*. The E-M steps are as follows:

1. Initialize $P(b)$ with a value between 0 and 0.5. (In practice, try 10 different values and check for convergence).
2. Using the current value of $P(b)$, solve for the optimal number and location of breakpoints using dynamic programming.
3. Compute the log likelihood of observing all SNPs in all G contigs of the genome:

$$\log L = \sum_{g=1}^{G} \log P(b) \cdot B_g + \log(1 - P(b)) \cdot (L_g - B_g) + \sum_{k=1}^{B_g+1} \log P(\tau_k, \nu_k, \theta \mid A_k)$$

Equation 4

where L_g is the length in bp of contig g, B_g is the number of inferred breakpoints and B_g+1 the number of blocks in contig g, and τ_k, ν_k and A_k are respectively the ML tree topology, branch lengths and alignment within block k.

4. Update the value of *P(b)* such that:

$$P(b) = \sum_{g=1}^{G} \frac{B_g}{L_g}$$

Equation 5

5. Iterate through steps 2-4 using the updated value of *P(b)* and continuing until convergence: $\log L_t - \log L_{t-1} \approx 0$, where t is the number of iterations.

Correction for model complexity.

To avoid inference of spurious breakpoints by STARRInIGHTS, we introduced a correction for model complexity. Every time a new breakpoint is included, a new tree topology and branch lengths are added as additional parameters to the model. Unless this is corrected for, many false breakpoints might be added in order to increase the likelihood of the model. For example, consider a hypothetical subsequence of 100 bp containing 10 phylogenetically-informative SNPs. For simplicity, assume that all 10 SNPs support the exact same tree topology, partitioning the isolates into two groups. If by chance 5 of the SNPs fell within the first 40 bp and the other 5 SNPs in the last 60 bp, the likeliest model might result in two blocks, each supporting the same tree topology, but with a longer branch length in the 40 bp block (5 SNPs / 40 bp $\approx$ 0.125 subs/site) than the 60 bp block (4 SNPs / 60 bp $\approx$ 0.067 subs/site). To quantify the contribution of this effect, we simulated sequences ranging in length from l = 10 bp to l = 211 kb (in binned increments each spanning ~10% of the observed core genome subsequences), and ranging in mutation rate from λ = 0.001 to λ = 1, where λ is the number of SNPs per site. For each combination of λ and l we simulated 100 sequences using seq-gen *(6)* with a tree chosen at random from the distribution of trees observed across subsequences of the core genome, and estimated the likelihood of a model with no breakpoints (*L0*) and a model with exactly one breakpoint (*L1*), placed optimally in the sequence to maximize the likelihood. Note that sequences were simulated using a single tree, so breakpoints introduced in the *L1* model are necessarily due to increased model complexity rather than actual recombination events. The maximum values of the log (*L1*/*L0*) ratio observed in 100 simulated contigs for each combination of l and λ are used as an empirical correction for model complexity. STARRInIGHTS was modified to include an appropriate correction factor for the values of l and λ in the subsequence being considered, and we again benchmarked on contigs simulated under Scenarios A and B. The correction for model complexity was accomplished by adding a penalty, $pen(i,j) = \log (L1/L0)_{\lambda^*,l^*}$, to the cost function $C(i,j)$ from Equation 1, where $\log (L1/L0)_{\lambda^*,l^*}$ is the observed maximum log ratio from 100 simulations with a given $\lambda = \lambda^*$ and $l = l^*$. In this notation, λ^* and l^* are the discrete, percentile-incremented parameter values used in the simulations that most closely match the observed $\lambda(k,j)$ and $l(k,j)$, where k = traceback(i) and the subsequence (k,j) consists of two flanking breakpoints at k and j, with a third breakpoint in between at i. By adding the appropriate penalty, we are correcting for the probability that the breakpoint at i due to model overfitting (*e.g.* due to the effect described in the hypothetical subsequence of 100 bp with unevenly distributed SNPs).

iv. Benchmarking on simulated and real sequence data

We tested the sensitivity and specificity of breakpoint detection by applying STARRInIGHTS to simulated contig sequences with predetermined recombination events. Simulated contig sequences of length 2500 bp (representative of the median LCB length in our data) were generated using simMLST *(7)*, using the default population model, a recombination tract length of 236 bp, with mutation rate (theta) of 5, 10 or 25, recombination rate (rho) of 0, 1, 10 or 25, and a sample of 8, 16 or 24 genomes. Mutation rates of 5, 10, and 25 resulted in 0.5-0.8%, 1-1.7%, and 2.6-4% polymorphic sites, respectively, and should correspond well with expected levels of 'within-species' diversity. The true-positive rate (TPR) for each parameter combination was computed as the number of inferred breakpoints divided by the number of actual breakpoints in the simulated ancestral recombination graph (ARG). Any breakpoints inferred in simulations with rho = 0 (no true recombination) were counted as false-positives, as were any additional breakpoints beyond those present in the ARG. These counts were divided by the number of sites with no true breakpoint to compute the false positive rate (FPR). True- and false-positive rates for each parameter combination (averages across 10 replicate simulations) are shown in fig. S9

The results of these simulations show that STARRInIGHTS is generally quite conservative, with zero false-positives for mutation rates below 25. The correction for model complexity (fig. S9B) reduces the TPR by ~1.3-5X, but improves the FPR to near zero for most parameter combinations. STARRInIGHTS is less sensitive than ClonalOrigin *(8)* in inferring true breakpoints, but also suffers slightly fewer false-positives. As an additional negative control, we examined 19 closely related *S. enterica* serovar Typhi genomes, previously reported to be largely clonal *(9)*, and found no evidence of recombination (Table S3).

Pre-filtering for regions of phylogenetic incongruence

The cost functions and dynamic programming described above rely on ML trees for each possible subsequence (i,j) of the genome. This requires precomputing a large number ($\sim L^2$) of trees. Specifically, $N(T)$, the number of ML trees to be inferred is:

$$N(T) = \sum_{g=1}^{G} \frac{L_g \cdot (L_g - 1)}{2} \qquad \text{Equation 6}$$

where L_g is the length in bp of contig g , and there are G contigs in the core genome. To reduce the computational burden of building $\sim O(L^2)$ ML trees, we perform a pre-filtering step to avoid building trees for subsequences (i,j) that almost certainly contain at least one breakpoint. To search for these clear cases of phylogenetic incongruence, we slide a 150 bp window along the sequence of informative SNPs and calculate, for each window, the probability that the SNPs on the left side and the right side of the window come from the same distribution. This is done by registering the frequency of the different SNPs observed in the window in a n-row, 2-column contingency table, where n is the number of different SNPs observed in the window and the columns correspond to the left and right sides of the window. We can then use a χ^2 test to calculate the significance that the observed SNPs are unevenly distributed over the window. This gives us a statistical criterion to split the alignment into smaller blocks tractable by the downstream dynamic-programming algorithm.

In addition, we distinguish cases of significant unevenness caused by incongruent phylogenetic topologies from those caused by long branch lengths, by explicitly measuring the average percentage of conflicting SNPs between all pairs of topologies found in the window:

$$\overline{F}_{dis} = \left< \frac{\min(SNP_{T1}, SNP_{T2}) - SNP_{T1 \cap T2}}{SNP_{T1} + SNP_{T2} - SNP_{T1 \cap T2}} \right> \qquad \text{Equation 7}$$

where SNP_{T1}, SNP_{T2} and $SNP_{T1 \cap T2}$ are the number of SNPs supporting topologies *T1*, *T2* or both *T1* and *T2*. The average in Equation 7 runs over all pairs of topologies found in the window.

For the *Vibrio* analysis presented in the main text, we split the alignment in smaller blocks at all positions with chi-square *p*-value < 1e-6 (corresponding to ~1 false positive breakpoint inserted every 1 million informative SNPs, or < 1 expected false positives in the 70,038 informative SNPs in the core genome) and a $\overline{F_{dis}}$ of at least 15% discordance between topologies (an example of the phylogenetic incongruence filtering procedure is shown in fig. S10). No pre-filtering step was applied to the *S. enterica* serovar Typhi genomes.

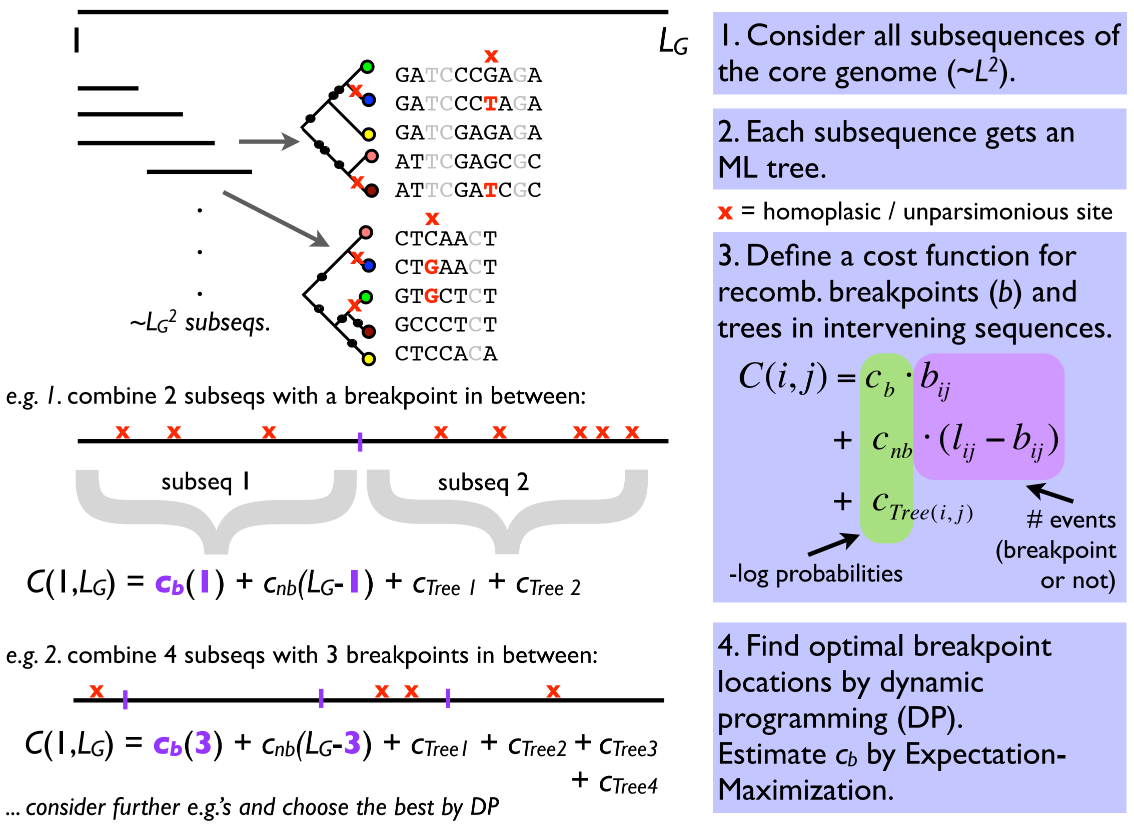
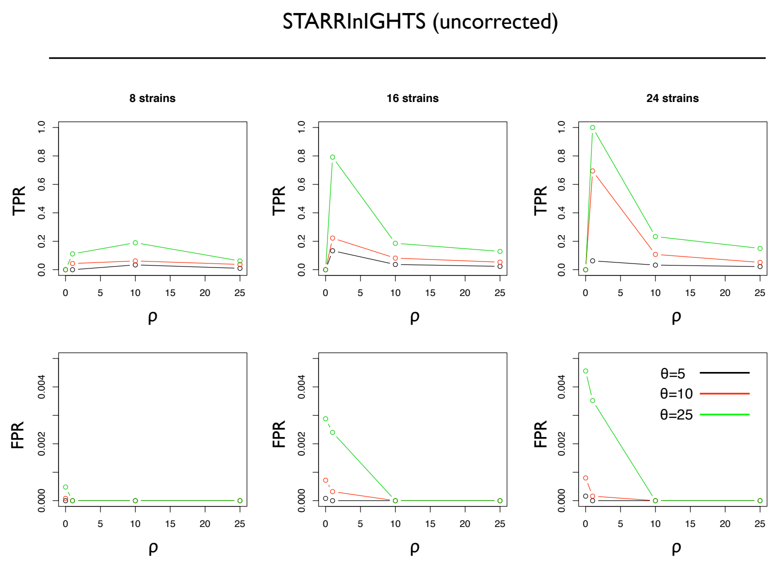

1
L_G
GATCCCGAGA
GATCCCTAGA
GATCGAGAGA
ATTCGAGCGC
ATTCGATCGC
CTCAACT
CTGAACT
GTGCTCT
GCCCTCT
CTCCACA
~L_G^2 subseqs.
e.g. 1. combine 2 subseqs with a breakpoint in between:
subseq 1
subseq 2
$C(1,L_G) = c_b(1) + c_{nb}(L_G-1) + c_{Tree\ 1} + c_{Tree\ 2}$
e.g. 2. combine 4 subseqs with 3 breakpoints in between:
$C(1,L_G) = c_b(3) + c_{nb}(L_G-3) + c_{Tree1} + c_{Tree2} + c_{Tree3} + c_{Tree4}$
... consider further e.g.'s and choose the best by DP
1. Consider all subsequences of the core genome (~L^2).
2. Each subsequence gets an ML tree.
x = homoplasic / unparsimonious site
3. Define a cost function for recomb. breakpoints (b) and trees in intervening sequences.
$C(i,j) = c_b \cdot b_{ij} + c_{nb} \cdot (l_{ij} - b_{ij}) + c_{Tree(i,j)}$
-log probabilities
# events (breakpoint or not)
4. Find optimal breakpoint locations by dynamic programming (DP). Estimate c_b by Expectation-Maximization.

Figure S8. Example STARRInIGHTS calculations and workflow.
See methods for a detailed description.

A



B

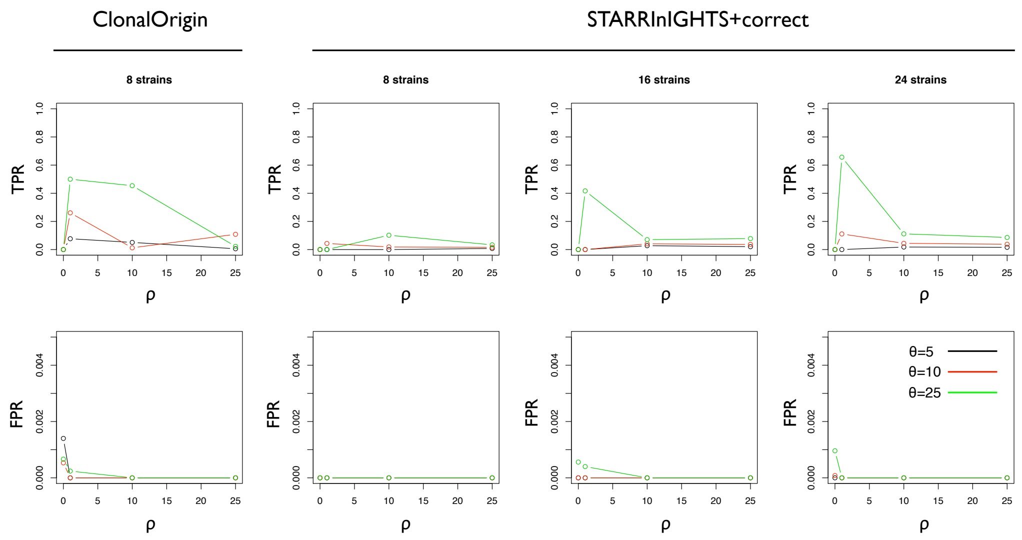


Fig. S9. STARRInIGHTS and ClonalOrigin benchmarked on simulated sequence.

True positive rate (TPR) and false positive rate (FPR) for recombination breakpoint inference in 2500 bp simulated core genome blocks. Simulations were performed as described in Methods. Points represent the mean of 10 replicate simulations for each combination of parameters recombination rate ρ, mutation rate θ, and the number of sampled strains. (A) uncorrected STARRInIGHTS, and (B) corrected for model complexity. ClonalOrigin (CO) was also used to infer breakpoints, as described in Methods.

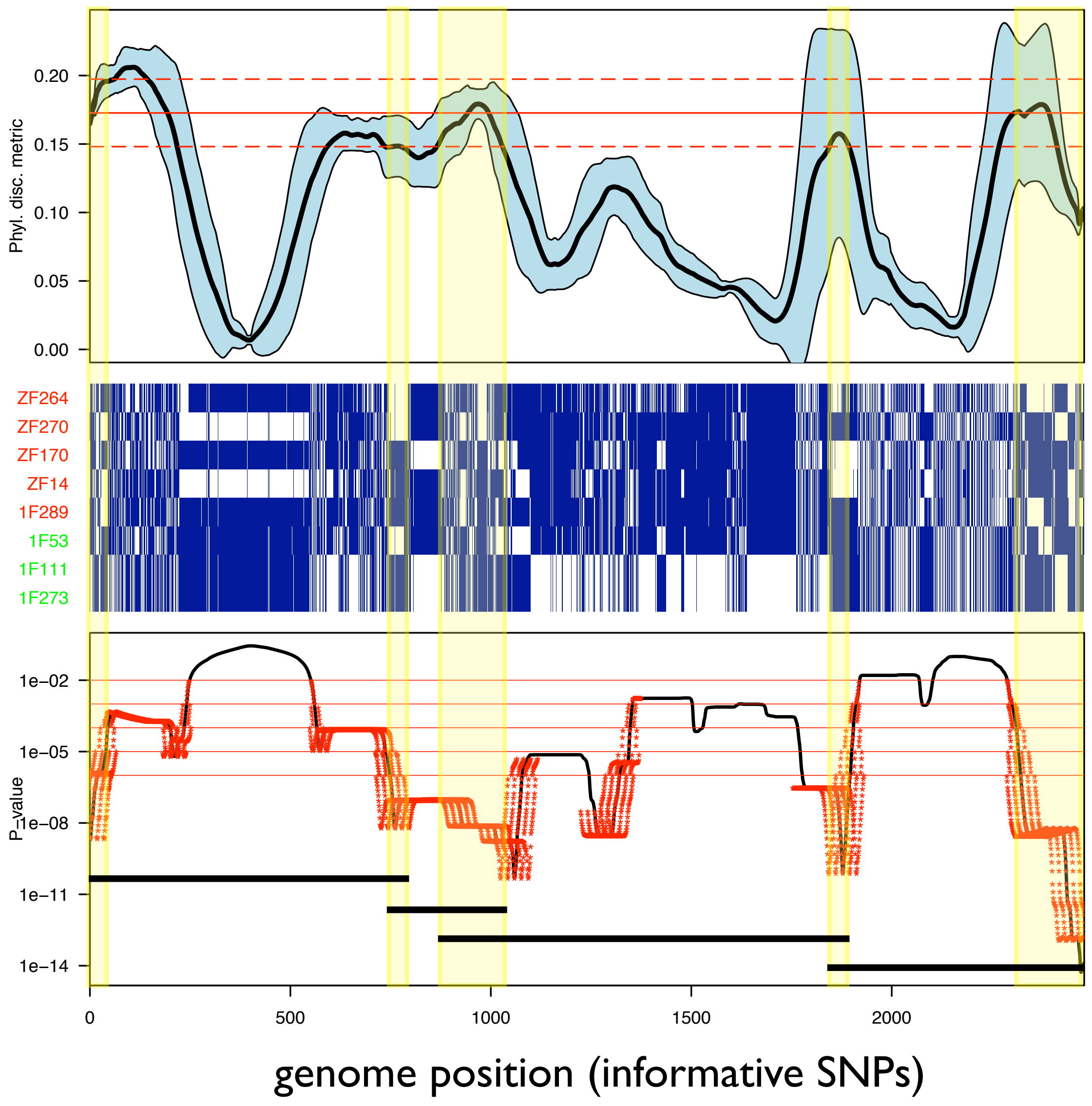


Fig. S10. Example phylogenetic incongruence filter.

The phylogenetic discordance metric, described in Methods, is plotted in the upper panel (proportion of incongruent SNPs). The red line corresponds to the expected value in a randomized alignment and the dashed red line is the same value minus the standard deviation. The middle panel displays informative, dimorphic SNPs in 8 strains (*e.g.* strains with the same base shown in white; strains with the alternative base shown in blue) for part of the core genome. The bottom panel shows the χ^2 *p*-value for stretches of significantly discordant SNPs. The yellow highlighted regions show significantly discordant stretches in which at least one breakpoint must be present (discordance metric > 0.15 and p < 1e-6). The contig (or LCB) is broken in these stretches such that four different, partially overlapping subcontigs are considered in the downstream STARRInIGHTS algorithm (horizontal black lines in lower panel). Within each subcontig, all subsequences (*i*,*j*) are used to build ML trees and infer further breakpoints.

Table S3. Recombination, polymorphism and divergence time estimates in *Vibrio* and *Salmonella* population genomes.

All values and parameters were inferred using STARRInIGHTS, setting the breakpoint probability using E-M or the 'strict' requirement of near-zero expected false-positive breakpoints (Methods). Genome size refers to the aligned 'core' only. Polymorphic sites are defined as either phylogenetically informative or not. (*e.g.* a SNP present in just a single terminal leaf is not informative). Homoplasic sites were counted with respect to the maximum parsimony tree for each block.

	Ingroup genomes		**Polymorphic sites**		**method to**	**Recombination**		**# homoplasic**
organism	**#**	**size (Mbp)**	**informative**	**non-inform.**	**set *P*(Break)**	**log *P*(Break)**	**# breakpoints**	**sites**
V. cyclotrophicus	20	3.54	70,038	40,974	E-M	-6.49	5,398	2,003
					strict	-14.10	3,550	3,545
S. enterica	19	5.81	473	1,313	E-M	-15.38	0	13

References

1. X. Didelot, D. Falush, *Genetics* **175**, 1251–1266 (2007).

2. V. N. Minin, K. S. Dorman, F. Fang, M. A. Suchard, *Bioinformatics (Oxford, England)* **21**, 3034–3042 (2005).

3. B. Mau, J. D. Glasner, A. E. Darling, N. T. Perna, *Genome Biol* **7**, R44 (2006).

4. S. Guindon, O. Gascuel, *Systematic Biology* **52**, 696–704 (2003).

5. R. Durbin, S. Eddy, A. Krogh, G. Mitchison, *Biological Sequence Analysis: probabilistic models of proteins and nucleic acids* (Cambridge University Press, Cambridge, 1998), pp. 54–66.

6. A. Rambaut, N. C. Grassly, *Computer applications in the biosciences : CABIOS* **13**, 235–238 (1997).

7. X. Didelot, D. Lawson, D. Falush, *Bioinformatics (Oxford, England)* **25**, 1442–1444 (2009).

8. X. Didelot, D. Lawson, A. Darling, D. Falush, *Genetics* **186**, 1435–1449 (2010).

9. K. E. Holt et al., *Nat Genet* **40**, 987–993 (2008).