Author/s:
Edwards, DJ;Duchene, S;Pope, B;Holt, KE

Title:
SNPPar: identifying convergent evolution and other homoplasies from microbial whole-genome alignments

Date:
2021-01-01

Citation:
Edwards, D. J., Duchene, S., Pope, B. & Holt, K. E. (2021). SNPPar: identifying convergent evolution and other homoplasies from microbial whole-genome alignments. Microbial Genomics, 7 (12), pp.000694-. https://doi.org/10.1099/mgen.0.000694.

Persistent Link:
https://hdl.handle.net/11343/305243

MICROBIAL GENOMICS

RESEARCH ARTICLE

Edwards *et al.*, *Microbial Genomics* 2021;7:000694

DOI 10.1099/mgen.0.000694

SNPPar: identifying convergent evolution and other homoplasies from microbial whole-genome alignments

David J. Edwards[1]†, Sebastián Duchene[2], Bernard Pope[3,4,5,6] and Kathryn E. Holt[1,7,*]

Abstract

Homoplasic SNPs are considered important signatures of strong (positive) selective pressure, and hence of adaptive evolution for clinically relevant traits such as antibiotic resistance and virulence. Here we present a new tool, SNPPar, for efficient detection and analysis of homoplasic SNPs from large whole genome sequencing datasets (>1000 isolates and/or >100 000 SNPs). SNPPar takes as input an SNP alignment, tree and annotated reference genome, and uses a combination of simple monophyly tests and ancestral state reconstruction (ASR, via TreeTime) to assign mutation events to branches and identify homoplasies. Mutations are annotated at the level of codon and gene, to facilitate analysis of convergent evolution. Testing on simulated data (120 *Mycobacterium tuberculosis* alignments representing local and global samples) showed SNPPar can detect homoplasic SNPs with very high specificity (zero false-positives in all tests) and high sensitivity (zero false-negatives in 89% of tests). SNPPar analysis of three empirically sampled datasets (*Elizabethkingia anophelis*, *Burkholderia dolosa* and *M. tuberculosis*) produced results that were in concordance with previous studies, in terms of both individual homoplasies and evidence of convergence at the codon and gene levels. SNPPar analysis of a simulated alignment of ~64 000 genome-wide SNPs from 2000 *M. tuberculosis* genomes took ~23 min and ~2.6 GB of RAM to generate complete annotated results on a laptop. This analysis required ASR be conducted for only 1.25% of SNPs, and the ASR step took ~23 s and 0.4 GB of RAM. SNPPar automates the detection and annotation of homoplasic SNPs efficiently and accurately from large SNP alignments. As demonstrated by the examples included here, this information can be readily used to explore the role of homoplasy in parallel and/or convergent evolution at the level of nucleotide, codon and/or gene.

DATA SUMMARY

The authors confirm all supporting data, code and protocols have been provided within the article, through supplementary data files or other online sources as indicated in the article.

New content generated for this paper:

(1) SNPPar code is available from https://github.com/d-j-e/SNPPar. The version described here is v1.0.

(2) A GitHub repository containing the full protocol, 'in-house' code and data used to carry out the validation and performance testing is available at https://github.com/d-j-e/SNPPar_test. This repository includes all the simulated and real data sets used here.

INTRODUCTION

Bacterial pathogen populations are under strong selection from antimicrobials and host immune defences, and there is increasing interest in utilizing whole genome sequencing (WGS) data to detect adaptive evolution in response to these strong selective pressures. Of particular interest in the field

Received 16 July 2020; Accepted 14 September 2021; Published 07 December 2021

Author affiliations: [1]Department of Infectious Diseases, Central Clinical School, Monash University, Melbourne, Victoria, Australia; [2]Department of Microbiology and Immunology, Peter Doherty Institute for Infection and Immunity, The University of Melbourne, 792 Elizabeth Street, Melbourne, Victoria, Australia; [3]Melbourne Bioinformatics, The University of Melbourne, 187 Grattan Street, Carlton, Victoria, Australia; [4]Department of Clinical Pathology, The University of Melbourne, Victorian Comprehensive Cancer Centre, 305 Grattan Street, Melbourne, Victoria, Australia; [5]Department of Medicine, Central Clinical School, Monash University, Clayton, Victoria, Australia; [6]Department of Surgery (Royal Melbourne Hospital), Melbourne Medical School, Faculty of Medicine, Dentistry and Health Sciences, The University of Melbourne, Victoria, Australia; [7]Department of Infection Biology, Faculty of Infectious and Tropical Diseases, London School of Hygiene & Tropical Medicine, London, UK.

***Correspondence:** Kathryn E. Holt, kat.holt@lshtm.ac.uk

Keywords: bacteria; evolution; homoplasy; phylogeny.

Abbreviations: ASR, ancestral state reconstruction; CDS, coding sequence; ML, maximum-likelihood; Mtb, *Mycobacterium tuberculosis*; WGS, whole genome sequencing.

†Deceasedon October 1, 2020

Data statement: All supporting data, code and protocols have been provided within the article or through supplementary data files. Four supplementary figures are available with the online version of this article.

are repeated or convergent evolutionary patterns [1, 2] in response to specific selective pressures (such as drug exposure, or within-host evolution), which imply predictability of adaptive responses that could be used to track disease progression, and to guide selection of vaccine and drug targets or therapeutic strategies that are more difficult for bacteria to evade through adaptive evolution [3, 4].

A key signature of adaptive evolution is the presence of homoplasies in the population [2–4]. Homoplasic traits are those that have been gained (or lost) independently in two or more lineages since their divergence from a common ancestor, in contrast to those traits that were gained or lost only once in a population and are shared by virtue of vertical inheritance from a common ancestor [5] (see Fig. 1). Extending this to SNPs, homoplasic SNPs are those where the same derived nucleotide is present in two or more lineages due to independent mutation events that occurred since their divergence from a common ancestor (which harboured a distinct ancestral nucleotide). Under the infinite sites model of molecular evolution [6], the same substitution event should not be observed multiple times in the absence of positive selection, and thus homoplasic SNPs are considered important signatures of adaptive evolution.

Homoplasic SNPs can arise through different series of mutation events, classed as parallel, convergent or revertant evolution (see Fig. 1). Parallel homoplasic SNPs are those where the same substitution, e.g. A to T, occurs independently at the same site in multiple diverged lineages; convergent homoplasic SNPs are those where the same nucleotide arises in diverged lineages through a distinct series of substitution events at the same site (e.g. A to T in one lineage, and A to G to T in another lineage; see Fig. 1). Revertance refers to restoration of a derived nucleotide back to the ancestral nucleotide, resulting in sharing of the ancestral nucleotide through mechanisms other than direct inheritance (e.g. a prior mutation of A to T is reversed in a descendant subpopulation by a subsequent substitution of T to A; see Fig. 1). Note that because selection acts on the function of encoded proteins, convergence can also be examined at the level of codons, whereby independent

Impact Statement

DNA sequences of bacterial pathogens are mutating all the time; most changes are deleterious or neutral, but sometimes a mutation leads to functional change that allows the pathogen to evade a potential threat. These random mutational changes (SNPs) are so very rarely beneficial that when they do arise in parallel in distantly related isolates (known as homoplasic SNPs) this indicates that the change may be positively selected because it confers an adaptive advantage to the bacteria. Finding homoplasic SNPs in large sets of bacterial genomes is challenging because current tools require substantial time and computational resources to run. Here we present SNPPar, a software program to efficiently and accurately automate the detection and annotation of homoplasic SNPs from large whole-genome sequence data sets. We use simulated data to demonstrate accuracy of the program, and re-analyse published datasets using SNPPar to illustrate how the results can be used to gain insights into the evolution of antibiotic resistance and other traits. We envisage SNPPar will help facilitate the undertaking of long-term, real-time surveillance of bacterial pathogens, and their adaptive evolutionary response to interventions and control measures such as new drugs or vaccines.

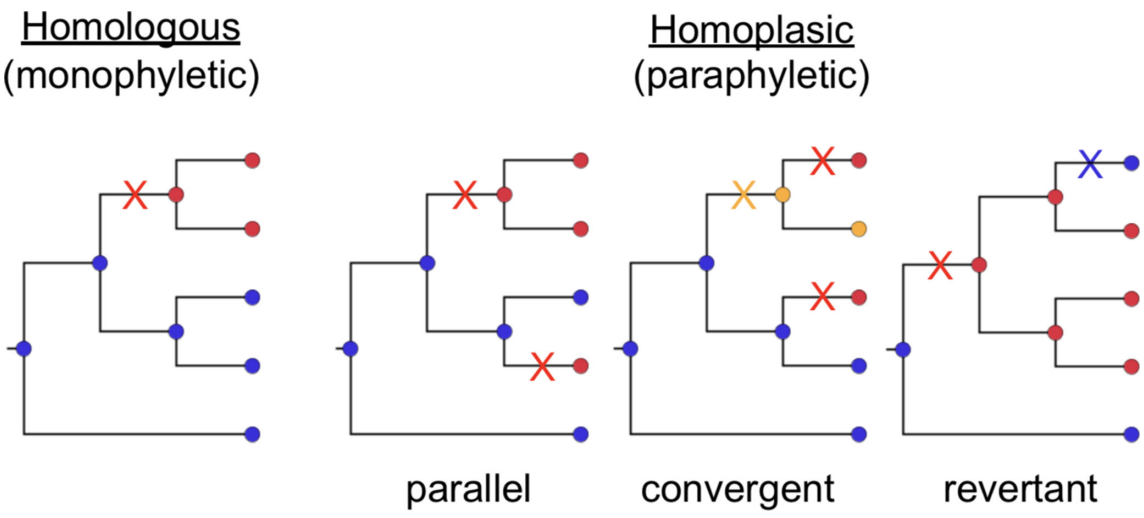

Fig. 1. Example trees demonstrating the relationship between homology (monophyly) and homoplasy (paraphyly). Homoplasic SNPs can be the result of parallel, convergent or revertant mutation events. Different colours of nodes indicate different nucleotide bases [e.g. blue, adenine (A); red, thymine (T); yellow; guanine (G)]. Different coloured crosses indicate mutations giving rise to the SNPs.

DNA mutations (e.g. a third or fourth allele at the same SNP site, or an SNP at another site in the same codon) result in the same or functionally equivalent amino acid substitution; or at the level of genes, whereby independent mutations affecting different codons have similar functional effects on the encoded protein's structure and function.

Homoplasic SNPs can be produced by spontaneous mutation (i.e. errors during chromosome duplication prior to cell division) but can also arise via recombination. Whilst recombination is a major driving force of evolution in many bacterial populations, in clonal pathogens that exhibit very little recombination, most new variation is acquired through spontaneous mutations. Genomics-based studies aiming to understand clonal evolution and positive selection typically seek to identify and remove homoplasic SNPs that result from recombination [7, 8], which improves the resolution of vertical patterns of evolution in a phylogenetic tree [9] and preserves homoplasic SNPs that arose through spontaneous mutation to be examined as signals of positive selection across the genome.

A general approach to discovering homoplasic SNPs from whole genome alignments is to construct a phylogenetic tree from the alignment of all SNP sites, then probabilistically infer ancestral states for each SNP site at internal nodes of the tree using ancestral state reconstruction (ASR) [10]. Example implementations of maximum-likelihood (ML) approaches to ASR include tools developed before WGS such as FastML [11–13] or PAML [14], and recently more efficient tools developed for genome-scale datasets such as TreeTime [15]. These ASR tools typically report inferred internal node sequences, and thus further analysis is required to infer mutation events and localize these to branches of the tree, to identify homoplasic SNPs, and to distinguish between parallel, convergent and revertant homoplasic SNPs. Further analyses are also required to predict coding effects of SNPs in order to identify convergence at the amino acid or gene level. Currently, at least two programs specifically target the discovery of homoplasic SNPs from microbial whole-genome alignments: HomoplasyFinder [16] and TreeTime [15]. Whilst both find homoplasic sites, the outputs are limited and require substantial further processing for most applications. HomoplasyFinder takes as input a tree and an SNP alignment, and reports sites with a consistency index <1 as homoplasic [16]. However, it does not identify the type of homoplasy, report details of the specific mutations and where they occur on the tree, nor include any functionality for analysing potential coding consequences to detect convergence at the codon level. TreeTime [15] is developed primarily for inferring ML-dated phylogenies and conducting ASR, and its ASR function is much faster than FastML or PAML (increase in execution time for the former is approximately linear with increasing sample size, whilst with the latter two the time increase is exponential). TreeTime also has a homoplasy function that can take as input a tree and an alignment, perform ASR and report sites that are homoplasic. TreeTime can report where each mutation at homoplasic SNP sites occur in relation to the tree, but does not attempt to differentiate the various types of homoplasy, nor consider any potential coding consequences of the SNPs. Therefore, if the goal is to find and discard homoplasic SNPs, then HomoplasyFinder or TreeTime are suitable; however, if the goal is detailed analysis of mutations potentially contributing to adaptive evolution, further processing of the output is required.

WGS has been widely adopted for the study of bacterial pathogens, and it is frequently applied to datasets numbering in tens to thousands of isolates. WGS is also increasingly used for routine public health surveillance of tuberculosis [17, 18] and foodborne pathogens [19], generating tens to hundreds of thousands of genomes that could potentially be used to interrogate parallel/convergent evolution in natural populations, providing important signals of adaptive evolution via selection for clinically relevant traits such as antibiotic resistance or virulence. There is therefore a need for efficient tools to detect and analyse homoplasic SNPs from large WGS datasets. Such datasets are particularly valuable for selection analysis because they can be highly informative for both phylogenetic tree inference and ASR [20–22] and tend to lead to the recovery of more homoplasic SNPs. However, they also pose substantial computational challenges [22, 23], including generating the phylogenetic tree and mapping the SNPs back to the tree using ASR, as well as post-ASR analysis to extract and classify the homoplasic SNPs. These challenges occur, in part, because both the phylogenetic and ASR inferences require increasingly computationally expensive likelihood calculations for each SNP site (as there are more nodes to resolve over), and because there are typically more SNP sites to test, as the tree size grows. Whilst there have been advances in improving rapid tree inference (e.g. RAxML-NG [24, 25] and IQTREE [26, 27]), ASR (e.g. TreeTime [15]) and simple detection and filtering of homoplasic sites (HomoplasyFinder [16] and TreeTime [15]), less attention has been paid to facilitating rapid analysis of convergent evolution from WGS data through detailed analysis of homoplasic SNPs.

Here we present SNPPar, a Python program that seeks to speed up the process of homoplasic SNP discovery and analysis in large bacterial WGS data sets (>1000 isolates and/or >100 000 SNPs). It calls on TreeTime for efficient ASR, and then processes the output to provide a wealth of additional data on mutation events and coding effects to facilitate the detailed assessment of evidence for adaptive evolution in large bacterial SNP data sets.

Theory and implementation

SNPPar is written in Python 3, utilizing the ETE3 [28] package for tree manipulation and TreeTime [15] or (optionally) FastML [11–13] for ASR. The package is available at https://github.com/d-j-e/SNPPar. Required inputs are a phylogenetic tree (Newick format), annotated reference genome (GenBank format) and details of SNPs for analysis (alignment formats are discussed below). The workflow for SNPPar is outlined in Fig. 2 and summarized below.

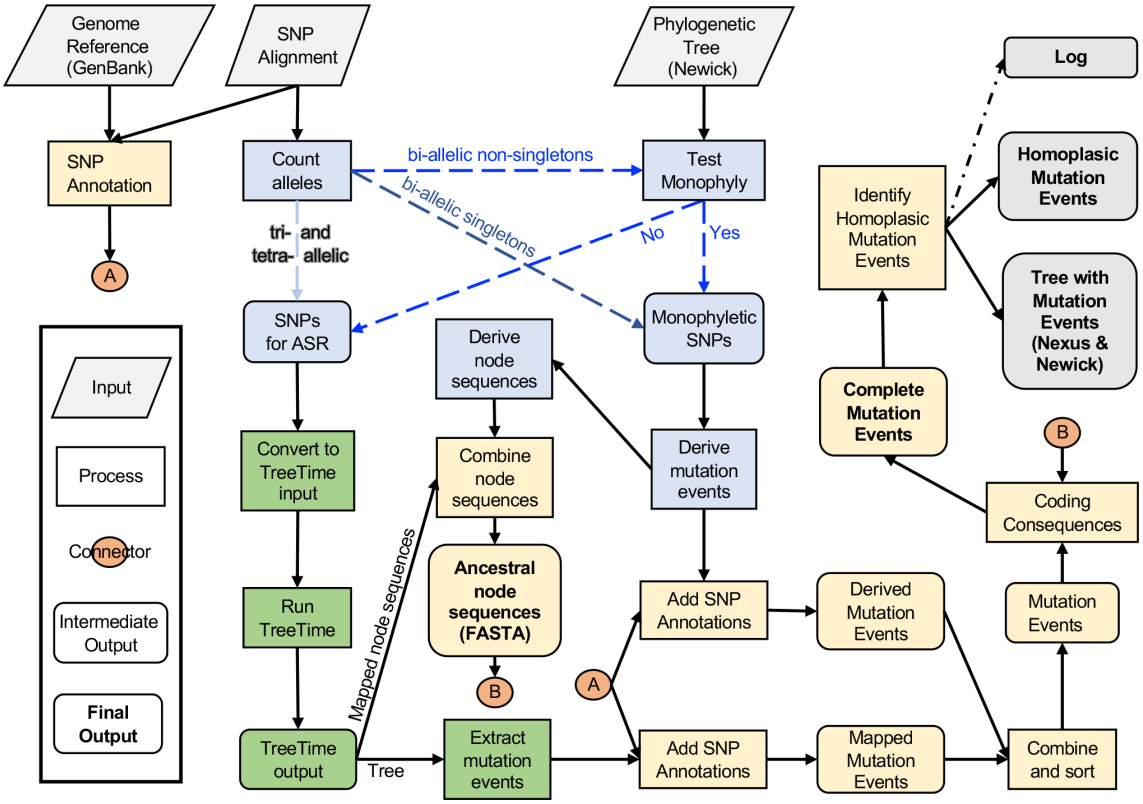

Fig. 2. Outline flowchart of SNPPar. Inputs are across the top (reference genome, SNP alignment, phylogenetic tree). The first steps involve SNP sorting (blue boxes), then ancestral state reconstruction (green boxes), and finally collation and functional annotation of mutation events (gold boxes). Final outputs are indicated in grey (list of mutation events, and tree file annotated with these events).

SNP sorting

First, SNPPar sorts SNP sites into different pathways for analysis (see Fig. 2).

(1) Bi-allelic SNPs that are singletons (minor allele present in only one input sequence) are trivially identified as monophyletic and assigned as a single mutation event arising on the relevant terminal branch (see Fig. 3a).
(2) Bi-allelic SNPs for which the minor allele is present in two or more input sequences (i.e. non-singleton) are assessed against the input tree to determine whether they are monophyletic (and thus assigned as a mutation event arising on the relevant internal branch), or paraphyletic and thus passed to ASR to infer mutation events on the tree (see Fig. 3b, d). See the Supplementary Information (available in the online version of this article) for details of this step.
(3) Tri-allelic and tetra-allelic SNPs are passed directly to ASR to infer mutation events on the tree.

SNPPar has three different sorting options available, which use all or part of the above pathways. 'Complex' sorting employs all of the above steps. 'Simple' sorting employs step (1) only and sends all but those SNPs identified as singletons to ASR. 'Intermediate' sorting (the default) employs most of the above steps, except any non-singleton biallelic SNP with one or more missing allele calls (i.e. an absent or ambiguous nucleotide, e.g. due to low read coverage, for any isolate in the dataset at that SNP position) is sent to ASR without assessing monophyly against the tree, because dealing with such missing data can be time-consuming (see below). For the largest empirical dataset used in this study (GL124 r2000, described below), the resulting SNP set designated for ASR included only a fraction of sites in the input SNP set (~2% of total SNP sites in the case of complex sorting and ~42% for simple sorting; intermediate sorting depends on the rate of missingness but falls between the other two). Hence, this sorting stage can considerably reduce processing time compared to conducting ASR on all sites. Initial testing, further confirmed by the validation testing below, led to the conclusion that intermediate sorting for datasets with missing calls was much quicker than the initial implementation, complex sorting, albeit with a small hit to peak memory use (see Results). Therefore, intermediate sorting was set as the default behaviour for SNPPar, but the other options are still available because (a) simple sorting can be used if users are concerned about the accuracy of the internal monophyly test and prefer to send all non-singleton SNPs to TreeTime, or (b) complex sorting can be used if memory is limited.

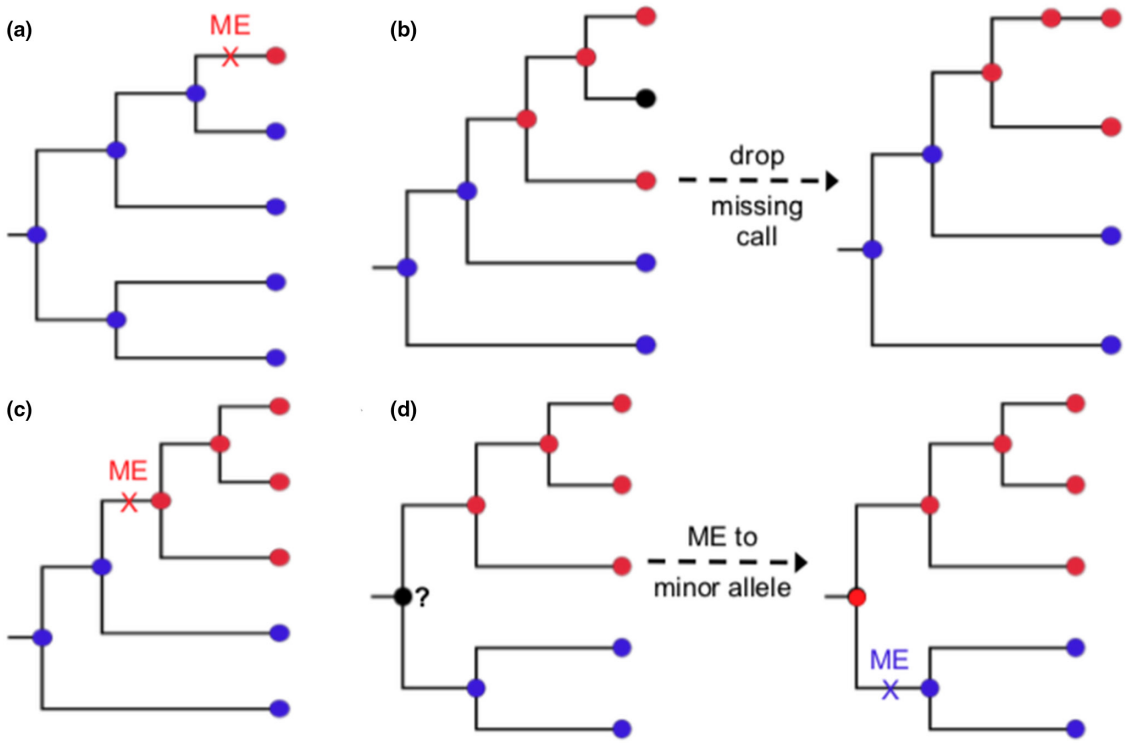

Fig. 3. Examples of assigning mutation events for bi-allelic SNP cases. (a) Singletons. (b) Handling missing calls. (c) One allele monophyletic. (d) Both alleles monophyletic. Each assigned mutation event (ME) is indicated on the branch with a cross, where colour indicates the derived base change involved. Note that if neither allele is monophyletic, the site is considered paraphyletic and passed to ASR.

Ancestral state reconstruction

ASR is performed for SNP sites identified during sorting, in order to infer mutation events on the tree using an ML approach. By default, SNPPar performs the ASR step by concatenating alleles from the SNP sites selected for analysis into a single FASTA-format alignment, then passing this together with the input tree to TreeTime's 'ancestral' command (note that missing or unknown base-calls can appear in this alignment but are set to 'N', as with all ML methods). TreeTime's default behaviour is to ignore sites with any missing base-calls, and hence SNPPar uses the TreeTime option to enforce reporting of all sites. Two output files are captured for processing by SNPPar: the sequences reconstructed for each internal node in the tree ('node sequences'), and a copy of the input tree with the inferred mutation events annotated on each branch ('event tree'), from which SNPPar extracts a list of all mutation events per SNP site. Optionally, FastML can be selected as an alternative ML algorithm for ASR, but this is much slower, and we found no difference in accuracy (see Results).

Collation and functional annotation of mutation events

All mutation events, whether inferred via internal sorting or ASR, are collated into a master event list that records the SNP coordinate in the reference genome, the precise base substitution (ancestral to derived base) and branch for each mutation event. These events are then annotated with their functional effects, with reference to the protein-coding gene (coding sequence, CDS) features annotated in the input reference genome and taking into account the inferred codon at each node not just the SNP site (see Supplementary Information for details). The final step in SNPPar is to identify, count and report unique homoplasic SNPs found in the full mutation event list. SNPPar can also identify the type of homoplasy for each homoplasic SNP (i.e. parallel, convergent and/or revertant mutation events), either in the main analysis run or as a post-processing step applied to a previously generated mutation event list and tree. SNPs called at the same position are compared to classify them as parallel, convergent, revertant or divergent. Comparisons are done pairwise, beginning with those that share both the same ancestral and derived calls, which are assigned as parallel. Where two events share the same ancestral call but a different derived call, they are assigned as divergent; where the derived call is the same but the ancestral call is different, this is assigned as convergent. Revertants are defined as cases where mutations (x→y) and (y→x) are identified at the same site, and one of the corresponding nodes is a descendant of the other, in which case the descendant is labelled as revertant.

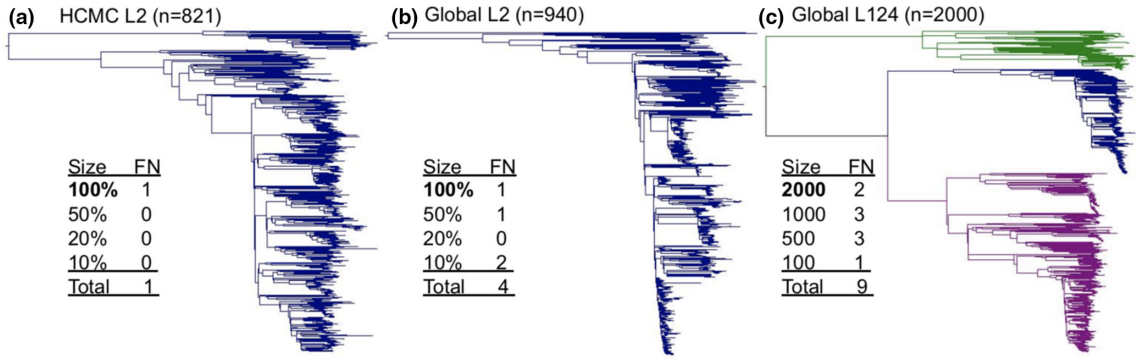

Fig. 4. Example of the largest trees from the three populations used for simulating *Mtb* data sets: (a) HCMC L2; (b) Global L2; and (c) Global L124. The tree in (c) consists of three lineages of *Mtb*; lineage 1 (green), lineage 2 (blue) and lineage 4 (grey), whilst (a) and (b) consist of only lineage 2 isolates. Inset tables show the number of false-negative (FN) homoplasy calls found in each set of 10 replicate simulations conducted for each sample size for each population (bold indicates tree size illustrated). FN calls are defined as homoplasies present in the simulated datasets that were not detected by SNPPar. Sample sizes for the HCMC L2 and Global L2 populations are random samples of genomes representing a set percentage subsample of the total sample size; for Global L124, a fixed number of genomes was sampled. Note there were no false-positive homoplasy calls across all simulated datasets.

Input and output files and formats

The reference genome used to call SNPs should be provided in GenBank format. It must be annotated, as this is essential to infer the function of the SNPs. SNPPar assumes the reference sequence is circular (for the purpose of reporting relative position of intergenic SNPs), and ignores any CDS that are split across the origin. The input tree must be rooted, bifurcating and in Newick format, with branch lengths expressed in substitutions per site. SNPs can be input in two ways: (i) a multi-FASTA concatenated alignment of SNP alleles (FASTA headers being sample names that match exactly the tip names in the input tree) plus a list indicating the coordinates of SNPs relative to the reference genome (one SNP coordinate per line, in the order in which SNP alleles appear in the alignment); or (ii) a comma-separated SNP allele table, with SNPs in rows (column 1 indicating the SNP coordinates, relative to the reference genome) and samples in columns (column headings being sample names that match exactly the tip names of the input tree).

Primary outputs are an annotated list of all mutation events and their coding effects, an annotated list of just those mutation events found at homoplasic SNP positions, and two tree files in which nodes are labelled with identifiers and mutation event counts (total SNPs and homoplasic SNPs). One tree file is in NEXUS format, compatible with visualization in FigTree (http://tree.bio.ed.ac.uk/software/figtree/) or iTOL [29], while the other contains the same data but in extended Newick format, suitable for use with the R package ggtree [30]. Internal node sequences are also output as multi-FASTA alignments. All program events are recorded to a time-stamped log file, including any relevant warnings (e.g. split gene in GenBank reference) or critical errors (e.g. providing FASTA alignment, but not the SNP position list).

METHODS FOR VALIDATION AND PERFORMANCE TESTING

All development and testing was done on a 2017 Apple MacBook Air (1.8 GHz Intel Core i5, 8 GB of RAM) using Python v3.7.2, ETE3 v3.1.1 and TreeTime v0.6.3. Results using FastML v3.11 for ASR on simulated data are included for comparison purposes. The full protocol, scripts and data used to carry out the validation and performance testing are available on GitHub (https://github.com/d-j-e/SNPPar_test).

Validation with simulated data

In order to test the accuracy of SNPPar when performing homoplasic SNP detection, we simulated 120 sets of sequences based on substitution model parameters and phylogenetic trees estimated from real *Mycobacterium tuberculosis* (*Mtb*) SNP alignment data [31]. Note this SNP data had been filtered previously to mask repetitive regions of the genome [31]. The data were divided into three subpopulations representing distinct epidemiological scenarios: (1) single lineage, single location (821 *Mtb* lineage 2 isolates from Ho Chi Minh City; HCMC L2); (2) single lineage, globally distributed (940 *Mtb* lineage 2 isolates; Global L2); and (3) species-wide, globally distributed (2965 isolates from lineages 1, 2 and 4; Global L124). Empirical parameters (i.e. number of SNPs and GTR model parameter estimates) were extracted from 10 randomly sampled subsets of various sizes from each population (10, 20, 50 and 100% of isolates from each of HCMC L2 and Global L2; n=100, 500, 1000 and 2000 isolates from Global L124). For each subset, (i) the number of SNPs was recorded and used later to correct the number of SNPs generated by simulation; and (ii) a reference tree and GTR model parameter estimates (alpha, base mutation frequencies and nucleotide frequencies) were obtained by selecting the tree with the highest likelihood from a random sample of five trees

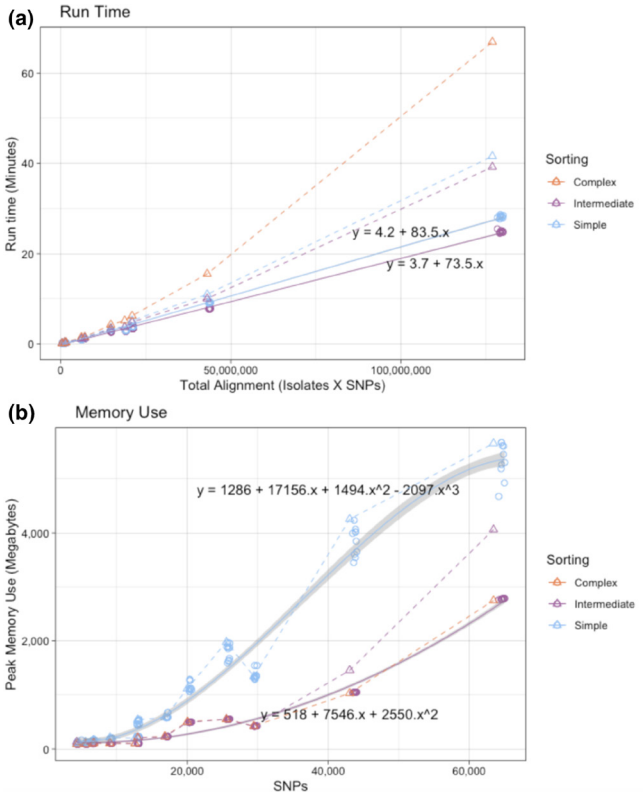

Fig. 5. SNPPar performance during analysis of subsampled empirical and simulated datasets using different sorting options. (a) Total run time vs total alignment length. (b) Peak memory use vs SNP alignment length. Circles denote simulated datasets, triangles empirical datasets; colours indicate sorting options as per inset key. The simulated datasets have 10 replicates of each sample size, and a line of best fit through these is indicated (solid line, grey shading indicates 90% fit interval). Note complex sorting was not used for the simulated datasets as there are no missing calls, and complex and intermediate sorting are identical in this case. The real datasets have only a single observation per sorting algorithm, which are joined with simple dashed lines.

estimated from the SNP alignment using RAxML v8.2.12 using a GTR +G model and ascertainment bias correction. SeqGen v1.3.4 [32] was used then used to simulate sequence evolution along these trees (see details in Supplementary Information and the SNPPar_test GitHub repository).

SNPPar was run on each simulated data set using either simple or intermediate sorting, with homoplasic SNP classification turned on. (Note that as there are no missing SNP alleles in the simulated data, complex sorting would be identical to intermediate sorting.) The list of homoplasic SNPs output by SNPPar was compared to the list of those expected based on the simulation. The starting tree was used as input rather than inferring new trees from each simulated alignment, because this would test the accuracy of the tree-building stage as much as the ability of SNPPar to find homoplasic SNPs. Any SNP expected to be called as homoplasic that was reported as such by SNPPar was scored as a true-positive; any other homoplasic SNP reported was considered a false-positive; any expected homoplasic SNP not reported by SNPPar was considered a false-negative. For homoplasic SNP calls, we also considered whether the type was classified correctly (i.e. parallel, revertant or convergent). As only one case of convergence was expected (and was always detected), only parallel and revertant type calls were considered.

SNPPar's total run time and maximum memory usage was recorded for each replicate run. The results were analysed in R v3.5.1 using RStudio v1.1.383, with independent ANOVA tests to examine the effects of isolate count, SNP count or total alignment length (i.e. total isolate count multiplied by SNP count) on total run time and memory.

Performance validation with real data

Testing of the performance of SNPPar in real-world analysis scenarios was demonstrated using two previously published bacterial WGS data sets in which convergent SNPs have been examined. One was from a 2015–16 *Elizabethkingia anophelis* outbreak in Wisconsin, USA [33] (68 isolates, 369 SNPs), the other from a retrospective sequencing study of in-host evolution and transmission of *Burkholderia dolosa* isolates amongst cystic fibrosis patients receiving treatment at a hospital in Boston, USA, in the 1990s [34] (114 isolates, 511 SNPs). As *B. dolosa* has three chromosomes, each reference chromosome and its associated SNPs were analysed separately using SNPPar (299, 153 and 59 SNPs on chromosomes 1, 2 and 3, respectively). The published tree for *B. dolosa* had branch lengths in units of substitutions (i.e. SNP counts), and hence these were scaled to units of substitutions per site by dividing each by the chromosome sequence length prior to use as input for SNPPar.

The performance of SNPPar in real-world analysis scenarios was further demonstrated using the largest subsampled empirical data set, Global L124 with 2000 isolates. To this group of samples, we added an outgroup (a single Lineage 5 isolate) and extracted the SNPs as previously above. We then constructed the phylogeny using IQTREE 2 [26, 27] using a transversion model for the mutation matrix, as indicated by IQTREE, and rooted it using the outgroup. The resulting tree was used in SNPPar along with the SNP dataset to obtain the mutation events across all three lineages (i.e. excluding the outgroup). Before analysis with SNPPar, we identified genes that overlap with masked repetitive regions, removed from the SNP alignment all SNPs located in these genes, and excluded these genes from further downstream analysis.

RESULTS

Validation of parallel SNP detection using simulated data

We assessed he accuracy of SNPPar using *Mtb* data simulated for three different population structures (single lineage, single city; single lineage, global; species-wide, global), subsampled at four different sizes, each with 10 replicates (total 120 simulations; see details in Methods). The two key outcomes

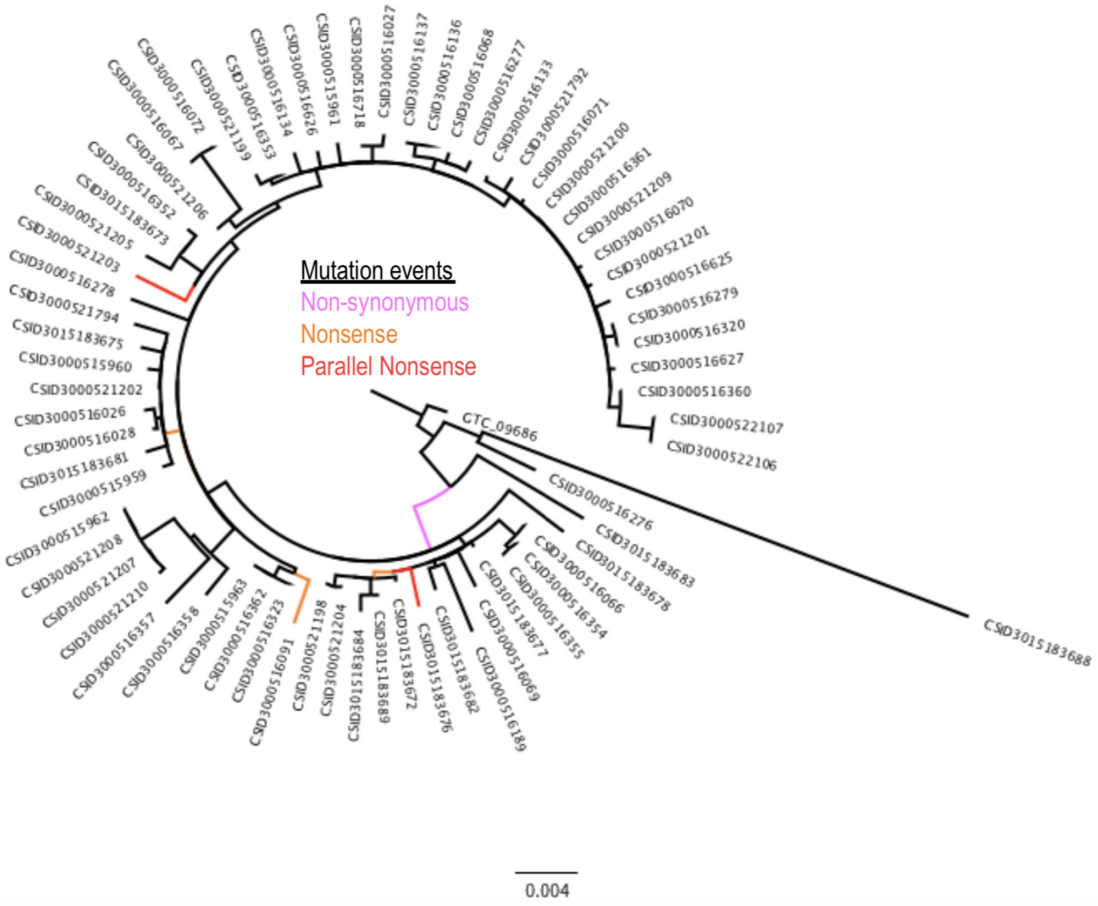

Fig. 6. Real data example: 2016–17 *Elizabethkingia anophelis* outbreak ML phylogeny [33] showing location (branch) of the six protein-altering mutation events of the *wzc* capsular export gene. The three types of mutation event shown are non-synonymous (one event, pink), nonsense (three events, orange) and parallel nonsense (two events, red). The tree includes an outgroup isolate (GTC_09686). Scale bar indicates substitutions per site.

assessed were the correct detection of homoplasic SNPs, and the correct identification of the type of homoplasy for each reported homoplasic SNP. The effects of sample characteristics (population structure, sample size and SNP count) on the two key outcomes was also explored.

Accuracy

Accuracy results (using the default option of intermediate sorting) are summarized in Figs 4 and 5. Note the results produced by SNPPar are deterministic; running the same inputs more than once produces exactly the same outputs. This extends to the choice of sorting; amending how SNPPar sorts the SNPs has no effect on accuracy when there is no missing data (as with the simulated datasets), and very minor effect if there is missing data.

SNPPar generated no false-positive homoplasy calls across the 120 replicate tests. Most replicate runs produced either zero (n=107 runs) or one (12 runs) false negative; the single exception was one run on a dataset of the smallest sample size from the Global L2 population, in which two false negative calls were made. As a general trend, the more complex the population, the more likely a homoplasy would be missed (Fig. 4). Notably, in all cases it was more likely to encounter no false negatives at all, i.e. all homoplasic SNPs were detected. Three types of false-negative homoplasy calls were observed; all involved SNPPar failing to identify a homoplasy in a scenario for which the available data supported a simpler single-mutation explanation (e.g. parallel mutation events on sister branches, which were called as a single mutation on the branch leading to the parent node; see Fig. S1a–c). Note that such scenarios are also not possible to detect using alternative approaches (such as ASR on all sites, or HomoplasyFinder's consistency index).

The true-positive homoplasic SNPs were assessed to see whether the type (parallel, convergent or revertant) was correctly reported by SNPPar. Overall, most types were correctly identified, and accuracy was improved with sample size (see Fig. S2a). In the smallest sample size (n=100), a mean of 13.7 homoplasic SNPs were called, of which a mean

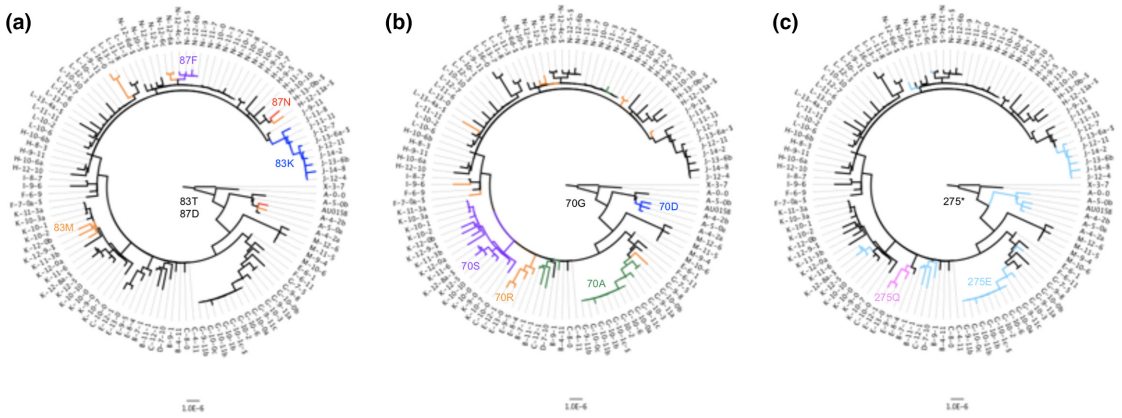

Fig. 7. Real data example: *B dolosa* phylogeny [34] showing evolution of codons 83 and 87 in *gyrA* (a), codon 70 in *rpl4* (b) and codon 275 in *wbaD* (c). Bar, substitutions per site. The tree includes an outgroup (X-3–7) from prior to the transmission chain. Labels for isolates from the same patient start with the same capital letter. Encoding of the codon(s) at any position in the tree is indicated by branch colour, with the ancestral call indicated with black. In (a), the ancestral codon 83 of *gyrA* that encodes threonine (T) is changed to methionine (M, orange), phenylalanine (F, purple) or lysine (K, blue). Codon 87 in the same gene changes from aspartic acid (D) to asparagine (N, red). In (b), the ancestral codon 70 of *rpl4* that encodes glycine (G) changes to serine (S, purple), arginine (R, orange), alanine (A, green) or aspartic acid (D, blue). In (c), the ancestral (truncating) stop codon of *wbaD* (position 275) is changed to either glutamic acid (E, light blue) or glutamine (Q, pink).

of 8.4% were classified incorrectly; in the largest samples (n=2000), a mean of 425.5 homoplasic SNPs were called of which a mean of 0.84% were classified incorrectly. The most common error was a reversion being incorrectly classified as a parallel event, followed by a parallel event being incorrectly classified as a reversion (see Fig. S2b); both cases were caused by the necessity to make arbitrary ancestral allele calls at the root node, which is unresolvable with any method.

Using FastML for the ASR step yielded no improvement in accuracy over TreeTime, producing the same false-negative calls and no false-positive calls. FastML yielded some differences in homoplasy type call errors, with a higher overall error rate (~5% more incorrect type calls with the smallest data set and ~0.7% more errors with the largest, see Fig. S3a), but this was due solely to differences in arbitrary assignments at the root node (see Fig. S3b).

Performance

Resource usage (total run time and peak memory) for SNPPar analyses with empirical and simulated data is shown in Fig. 5. For the simulated datasets, sorting options included simple and intermediate (as there is no missing data), whilst all three options were tested with the empirically subsampled (real) datasets. Variance in total run time was most strongly associated with total alignment size of the input dataset, whilst the peak memory use was most strongly associated with the number of SNPs (see Supplementary Information). Whilst the choice of sorting algorithm for the simulated datasets had only a minor impact on total run time (intermediate being slightly quicker than simple sorting, solid lines in Fig. 5a; ~11% quicker for intermediate sorting with the largest data set, Global L124, with a sample size of 2000), there was a major difference in the peak memory use as more SNPs were sent to ASR with simple sorting (solid lines in Fig. 5b; ~49% more memory used for intermediate sorting with the largest data set). Memory use followed the same pattern for the empirical data (which include missing alleles and many more homoplasic SNPs), with simple sorting (most SNPs sent to ASR) using the most memory and complex sorting (fewest SNPs sent to ASR) the least (dashed lines in Fig. 5b). However, this comes at the expense of runtime; empirical datasets took longer to analyse than similar-sized simulated datasets, and complex sorting took ~50% longer than simple or intermediate sorting (dashed lines in Fig. 5a). Hence, whilst the default setting of intermediate sorting was consistently the fastest (~40 min on the largest empirical dataset), this increase does come at a cost to peak memory use compared to the slower complex sorting (~4 GB vs ~2.5 GB on the largest empirical dataset), which may become problematic with very large datasets.

Comparing the performance of SNPPar to TreeTime with regard to the ASR step, the whole alignment of a simulated data set of Global L124 with 2000 isolates took ~22 min for TreeTime to analyse all ~64 000 SNPs using >24 GB of peak memory; SNPPar using the default intermediate sorting sent only ~800 SNPs (1.25% of the total) to TreeTime for ASR, which took ~23 s with a peak memory use of ~0.4 GB of RAM. Interestingly, TreeTime alone had a lower sensitivity than SNPPar in this test (n=419/430 sites correctly called as homoplasic, vs n=429/430 for SNPPar or HomoplasyFinder; see Supplementary Information), suggesting that SNPPar's sorting approach provides benefits over TreeTime analysis of the full alignment in terms of recall accuracy as well as memory use.

Table 1. Summary of SNPPar results for *B. dolosa* loci in which homoplasic or divergent SNPs were identified [34]

Genes in bold type are those with parallel SNPs that occur three or more times independently and are discussed in the text. The type of change (coding effect) of SNPs can be synonymous (S) or non-synonymous (NS). Amino acid changes are indicated in the format [ancestral amino acid, codon, derived amino acid], e.g. P537L indicates that the amino acid at codon 53 has changed from proline (P) to leucine (L); an asterisk (*) indicates a stop codon.

Locus tag	Chromosome::position	Mutation events	Type of mutation event(s)	Base substitution	Type of change	Amino acid
BDAG_00179	1::232a898	2	One reversion	C to T, then T to C	NS	P537L then L537P
BDAG_01022	1::1 267 993	2	One parallel	A to C	NS	K30T
intergenic	1::1 431 059	*2*	Two divergent	C to G	–	–
				C to T	–	–
BDAG_02180 (***gyrA***)	1::2 694 721	4	Two parallel	G to A	NS	D87N
			One divergent	C to G	NS	H87D
				G to T	NS	D87F
	1::2 694 732	6	Four parallel	C to T	NS	T83M
			One divergent	C to A	NS	T83K
BDAG_02317 (***wbaD***)	1::2 849 264	10	Eight parallel	T to G	NS	*275E
			One divergent	T to C	NS	*275Q
BDAG_02593	1::3 170 526	*2*	Two divergent	T to C and T to G	both S	–
intergenic	1::3 380 836	*2*	Two divergent	G to T and G to C	–	–
BDAG_02781 (***rpl4***)	1::3 390 724	5	Three parallel	G to C	NS	G70A
			One divergent	G to A	NS	G70D
	1::3 390 725	9	Seven parallel	G to C	NS	G70R
			One divergent	G to A	NS	G70S
BDAG_03003	2::261 481	2	One parallel	A to C	NS	K1525T
BDAG_03043	2::330 177	2	One parallel	T to G	NS	S28R
BDAG_03095	2::397 597	2	One parallel	T to G	NS	L35V
intergenic	2::701 457	2	One parallel	A to C	–	–
intergenic	2::819 544	2	One parallel?	T to G, *or*	–	–
				G to T, then T to G		
BDAG_03694 (***bla1***)	2::1 231 197	6	Three parallel	C to A	NS	R172S
			One parallel	C to G	NS	R172G
intergenic	2::1 616 780	3	Two parallel	G to A	–	–
BDAG_04180	2::1 838 491	3	Two parallel	C to T	NS	A103V
BDAG_04506	3::70 968	2	One parallel	C to G	NS	H209D
intergenic	3::72 093	2	One parallel	C to A	–	–

Using FastML rather than TreeTime for ASR in SNPPar saw a minor improvement on memory usage but increased the run time markedly, especially for larger data sets (see Fig. S4a-b). For example, the largest simulated data set (Global L124 with 2000 isolates) took on average ~25 min and used ~2.7 GB of memory with TreeTime for ASR (using intermediate sorting) and ~3 h and ~1.6 GB with FastML for ASR. FastML by itself, given the same whole alignment of Global L124 simulated data and computing resources as TreeTime above, failed to complete the analysis after 7 days. SNPPar using FastML also failed (i.e. exited with a fatal error) during the ASR step for two of the largest real SNP alignments (Global L124 with 2000 isolates, 1254 SNPs sent to ASR; and HCMC L2 with 821 isolates, 395 SNPs sent to ASR) which completed in ~25 and~3 min, respectively, using SNPPar with TreeTime and intermediate sorting).

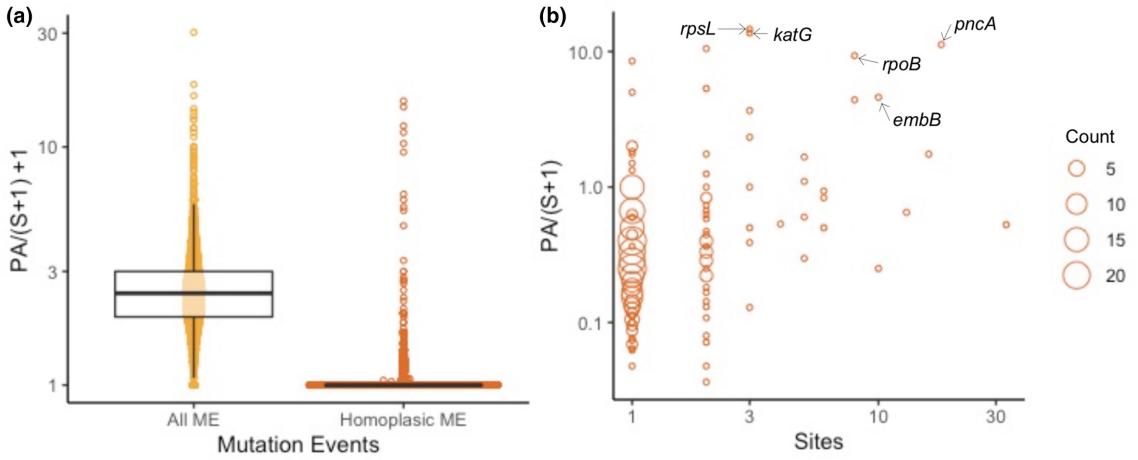

Fig. 8. Real data example: gene-level summary of homoplasies detected in *Mtb*. (a) Distributions for the ratio of protein-altering (PA) events per gene normalized to synonymous SNP counts per gene, for all (yellow) and homoplasic (orange) mutation events. (b) Number of nucleotide sites affected by homoplasies (*x*-axis) vs PA ratio (*y*-axis); each point represents one or more genes with the same (*x,y*) values, and circle size indicates the number of genes (as per inset key). Resistance genes discussed in the text are labelled. Note that this analysis includes 2000 strains randomly sampled from the Global L124 set [31].

Demonstration of SNPPar use cases with real data

Two data sets from previously published papers reporting homoplasy analysis in *E. anophelis* [33] and *B. dolosa* [34] along with the largest empirically sampled *Mtb* dataset used for simulation (Global L124 with a sample size of 2000 plus outgroup), were analysed with SNPPar using default parameters (details in Methods). The data sets, commands and outputs are available on GitHub (https://github.com/d-j-e/SNPPar_test).

E. anophelis is an opportunistic pathogen that rarely causes outbreaks, and the genome data from an unusually large outbreak in the US provide a rare opportunity to explore positive selection in this organism [33]. SNPPar analysis of the alignment of 369 SNP sites from 68 *E. anophelis* outbreak genomes took <3 s, and identified the same homoplasic SNP that was reported in the original study (which used FastML) [33] (see Fig. 6). The SNP (G to T at position 2 169 660 of the reference sequence) was assigned to two different terminal branches, indicating it had arisen via parallel substitutions in two different outbreak strains (CSID3000521203 and CSID3015183676). SNPPar correctly annotated this as a parallel SNP affecting the first position of codon 469 in the *wzc* capsular export gene (A2T74_09840), resulting in a nonsense mutation (converting the glutamate codon 'GAA' to stop codon 'TAA') and thus truncating the protein as previously reported [35]. Because SNPPar annotates all mutation events, the output can also trivially be used to extract gene-level information about selection, not just homoplasic SNPs. For example, the original study [33] reported that 27 genes had two or more independent protein-altering mutations (i.e. non-synonymous or nonsense mutation events) in the outbreak population, including three with five or more protein-altering mutations (a hypothetical protein potentially involved in starch utilization, the capsular export gene *wzc*, and another sugar transporter similar to *wza*). SNPPar identified the same numbers of mutation events in these 27 genes. Five protein-altering SNPs (involving six independent mutation events) were correctly found in *wzc*, including the two for the parallel nonsense SNP at codon 469 (see Fig. 6).

WGS from serial *B. dolosa* isolates from chronically infected cystic fibrosis patients provides an opportunity to investigate in-host evolution of the pathogen [34]. Under these conditions the bacterial population is constrained by isolation and strong selective pressure from the host, and hence substitutions are a dominant mechanism of adaptation and parallel mutations can provide a strong signal of positive selection. *B. dolosa* has three chromosomes and SNPPar was run on separate alignments for each chromosome (using a single consensus tree of 114 isolates inferred from the concatenated alignment of 511 SNPs), which took a total of ~11 s. Twenty SNP sites with more than one mutation event were previously reported in this data set [34]; SNPPar correctly identified all 20 (see Table 1), despite the alignment having high levels of missing alleles for some SNPs (up to 106/114 alleles missing at a given SNP site). The SNPPar output includes the inferred type of homoplasy for each position, along with the precise mutation event(s) involved and annotation of the coding effects (see Table 1). Fourteen intragenic parallel SNPs were detected, and notably all were non-synonymous. Three genes harboured multiple non-synonymous parallel SNPs, consistent with strong positive selection for modified protein function (highlighted in Table 1). All three of these genes were associated with antibiotic resistance. Parallel and divergent SNPs were detected in codons 83 and 87 of *gyrA* (BDAG_02180; see Fig. 7a); these were linked to fluoroquinolone resistance in the original study [34]. Two distinct parallel SNPs were identified in codon 172 of beta-lactamase *bla1* (BDAG_03694), consistent with the original study in

Table 2. Top ten *Mtb* genes in terms of the ratio of protein-altering homoplasic mutation events vs all synonymous mutation events

Gene names in bold indicate genes previously reported by Fahrat *et al*. [35] and Coll *et al*. [36] to be targets of convergent positive selection. ME, mutation event count; S, synonymous SNP count; PA, protein-altering mutation count; sites, number of nucleotide sites affected by one or more homoplasic SNP.

Gene	ME	S	PA	PA/(S+1)	Sites	Gene	Product	Function (Reference)
Rv0682	76	3	73	14.6	3	***rpsL***	30S ribosomal protein S12	Streptomycin resistance [37]
Rv1908c	96	0	96	13.7	3	***katG***	Catalase-peroxidase	Isoniazid resistance [38]
Rv2043c	45	0	45	11.3	18	***pncA***	Pyrazinamidase/ nicotinamidase	Pyrazinamide resistance [39]
Rv0336	21	0	21	10.5	2	–	Hypothetical protein	–
Rv0667	140	0	140	9.3	8	***rpoB***	DNA-directed RNA polymerase subunit beta	Rifampin resistance [40]
Rv2828A	17	0	17	8.5	1	–	Hypothetical protein	–
Rv1944c	16	0	16	5.3	2	–	Hypothetical protein	–
Rv0814c	5	0	5	5.0	1	*sseC2*	Hypothetical protein (sulphur metabolism)	Latency [42]
Rv3795	124	0	124	4.6	10	***embB***	Arabinosyltransferase B	Ethambutol resistance [41]
Rv0750	52	8	44	4.4	8	–	Hypothetical protein	–

which this gene was noted as under positive selection [34]. The third gene was the ribosomal 50S L4 protein-encoding gene *rpl4* (BDAG_02781), which is involved in both pathogenicity and macrolide resistance [36]. SNPPar identified parallel SNPs in the first and second base positions of codon 70 of *rpl4*, consistent with the original report which identified it as under positive selection, with four independent mutations affecting codon 70 [36]. As shown in Fig. 7b, SNPs at these two positions in *rpl4* arose in parallel a number of times (four and eight respectively), but never in the same isolate. Overall, SNPPar identified five parallel non-synonymous SNPs that arose three or more times independently in the population, which we take to indicate a strong signal of positive selection: one *gyrA*-83 SNP, one *bla1*-172 SNP, the two *rpl4* SNPs, and an SNP in codon 275 of the glycosyltransferase gene *wbaD* (BDAG_02317). Notably an additional divergent SNP also occurred at the same site in *wbaD* codon 275 (see Table 1). As noted in the original study, the ancestor of this *B. dolosa* cluster harbours a premature stop codon at this position, and the SNPs at this site restore the truncated form of the protein to the full-length protein, restoring proper function of the gene and generating O-antigen repeats. SNPPar results are consistent with the analysis conducted in the original study (see Fig. 7c), which reported that restoration of the protein via SNPs at this site occurred independently in nine subjects [34].

For the *Mtb* dataset (total of 63 065 SNPs in 2000 genomes), SNPPar took 51 min (using 2.6 GB) and identified a total of 2879 homoplasic mutation events affecting 461 genes. We used the output to calculate the rate of homoplasic protein-altering mutations per gene, normalized to the total number of synonymous mutation events per gene, as a measure of adaptive selection (see Fig. 8a). The top ten scoring genes (see Table 2) include five known targets of positive selection associated with antibiotic resistance [35, 36] (*rpsL* [37], *katG* [38], *pncA* [39], *rpoB* [40] and *embB* [41]). The other five encode hypothetical proteins, though one of these (*sseC2*) has been implicated in down-regulation of genes involved in latency in *Mtb* [42]. The five antibiotic resistance-associated genes each harboured homoplasies at three or more nucleotide positions (from three in *katG* and *rpsL*, across two and three codons respectively, to 18 in *pncA*, across 18 different codons; see Fig. 8b). In *katG*, which is known for its isoniazid-resistance-conferring mutations [38], most of the homoplasic mutation events (90/96) occurred in codon 315 (see Fig. 9a). The majority of these (80/90) resulted in the same amino acid substitution (S315T), though there were also three (parallel) reversions at the same codon position (i.e. T315S, largest affected clade shown in Fig. 9b). Note that these exploratory analyses are simple to conduct using the tabular and tree output of SNPPar.

DISCUSSION

Performance testing of SNPPar based on simulated data demonstrated that it has very high sensitivity to detect homoplasic mutation events (with 89% of tests yielding perfect recall, range 0–2 false-negatives per test) and also has high specificity (i.e. no false-positive calls were produced). The number of false-negative calls did increase slightly with increasing sample complexity (see Fig. 1), and hence the raw number of false-negatives may increase for larger empirical data sets. SNPPar was also highly accurate in classifying the identified homoplasies into types; the percentage of incorrect calls was inversely proportional to the sample size (see Fig. S2a), with <1% of incorrect calls in the largest simulated dataset. Most homoplasy-type call errors were either revertant mutation events called as parallel events, or vice versa, where the call involved the root node and thus cannot be resolved by any method without the use of outgroups, or additional information about ancestral states (see Fig. S2b).

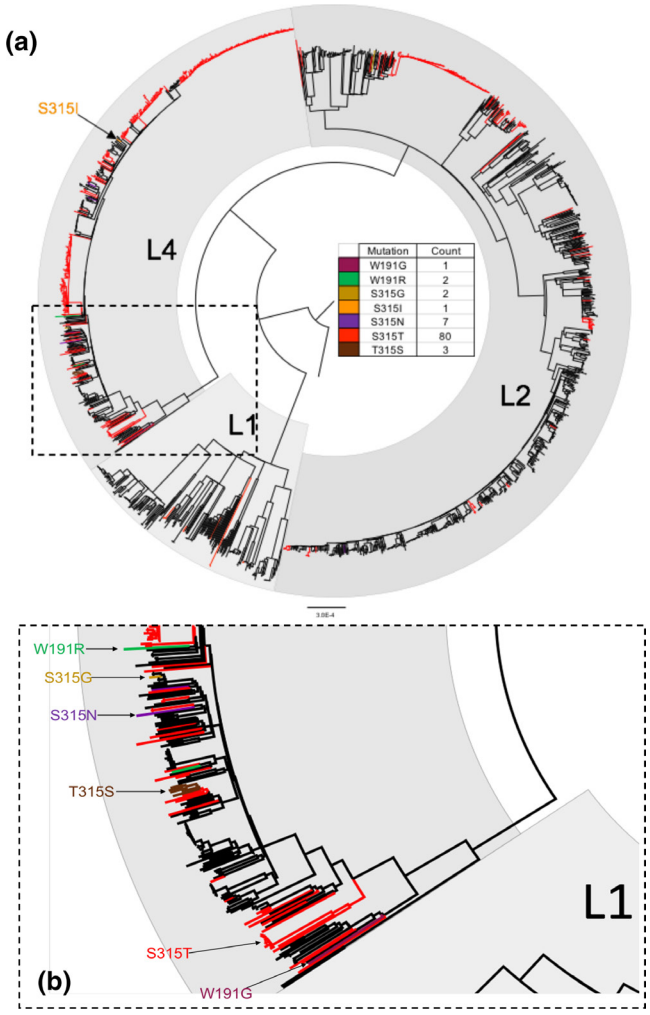

Fig. 9. Real data example: *katG* homoplasies detected in *Mtb*. (a) ML phylogeny showing the location of homoplasic mutation events in the gene *katG* identified using SNPPar, coloured to indicate the resulting amino acid substitution (as per inset key). (b) Expanded insert from (a). Note that this analysis includes 2000 strains randomly sampled from the Global L124 set [31]; L1, L2 and L4 indicate lineages 1, 2 and 4.

Examination of the computational efficiency of SNPPar demonstrated a linear relationship between execution time and total alignment length for simulated data, when using the default TreeTime for ASR (see Fig. 5a). Real data took more time and memory to execute due to a greater number of allele patterns to test, mainly due to the presence of missing alleles in real data. Of the three sorting methods tested, intermediate sorting was the quickest for both simulated and empirical datasets (see Fig. 5a). However, this came with an increase in memory usage compared to the slowest, complex sorting, particularly with the larger datasets (see Fig. 5b), as more SNPs are passed to TreeTime for ASR. Notably all three sorting methods were much more efficient than using TreeTime to analyse the entire alignment (~4 GB for SNPPar with intermediate sorting and the largest empirical subsampled dataset vs >24 GB for TreeTime, see Supplementary Information). As the time saving was notably higher for intermediate sorting, and memory usage was still in a reasonable range, we have set this sorting method as the default. If memory becomes limiting, e.g. on much larger data sets, complex sorting may be preferable but can be expected to take about twice as long. Considering where improvements in performance could be made in future for tackling much larger datasets, possible strategies include doing simple sorting to save time but farming out the ASR step into small batches to reduce memory, or first dividing the task before the sorting step (whilst ensuring SNPs in the same gene stay within the same batch). Using FastML for the ASR step in SNPPar was notably slower for both simulated and real data, with no improvement in accuracy or memory use; hence, we recommend using the default TreeTime for ASR.

SNPPar analysis made it relatively easy to reproduce the selection analyses reported previously on empirical data; one command, run using standard input files of tree and SNP alignment, took a few seconds on a laptop, producing details of all SNPs annotated with their coding effects and position in the tree. The richness of the output makes it straightforward to not only identify homoplasic mutation events, but also to explore convergent evolution events in detail, including by summarizing the data at the levels of SNP site, codon and gene (see Tables 1–2) and visualizing convergent and/or diversifying evolution across the genome and in the context of the phylogeny (see Figs 6–9). Notably, the SNPPar analysis is more reproducible than the published analyses and can be readily repeated as more isolates are added to the dataset.

There are some caveats with using SNPPar. First, results from SNPPar are only as good as the input data that are provided. Prior quality control of the SNP alignment is important, not only to obtain high-confidence SNP allele calls by removing those found in repeat regions or other regions with ambiguous or low-quality calls (e.g. the (P)PE genes found in *Mtb* [31]), but also to improve the resulting tree by removing the noise from these potentially ambiguous sites. The phylogenetic tree itself must then be inferred from the data, and there may be multiple trees that describe the data equally well; indeed, a bifurcating tree may not capture the relationships between strains accurately, and there may be low confidence regions of the tree topology. All these factors should be considered when feeding input to SNPPar, and when interpreting the results. For example, where multiple trees describe the data equally well (e.g. where there are many polytomies) or any other phylogenetic uncertainty, each tree from a set of candidates (e.g. those with the same statistical fit or from bootstrap replicates) could be run through SNPPar and the results combined to obtain consensus mutation event data from which to infer selection. Second, SNPs can arise through spontaneous mutations (independent substitution events) or through recombination events (horizontal transfer of a group of linked variants); the latter need to be efficiently removed from the input SNP alignment if a user wishes to

interpret SNPPar mutation event data as unlinked substitutions signifying independent parallel evolution. However, the output of SNPPar could be used to check an alignment for evidence of clusters of SNPs mapped to the same branch of the input tree, which may represent recombination; these could then be properly identified and removed from the alignment with targeted software (e.g. Gubbins [7] or ClonalFrameML [8]) prior to re-running SNPPar with a recombination-filtered tree and SNP alignment. Finally, users should keep in mind the inherent limitation that ASR cannot accurately distinguish mutation events immediately below the root node [43, 44], and hence ensure to include an outgroup.

We have demonstrated that SNPPar works efficiently on large genome-wide SNP sets; it is highly accurate in detecting true positive homoplasic SNPs relative to an input tree with no false positive calls. SNPPar located homoplasic SNPs efficiently and accurately with both the simulated and real data sets tested here. Currently, SNPPar is the only tool that automates the detection and annotation of homoplasic SNPs efficiently from large SNP alignments. As further demonstrated by the real examples, this information can be used to explore the role of homoplasy in parallel and/or convergent evolution at the codon and gene level, in addition to identification of homoplasic SNPs per se. Whilst SNPPar was designed and tested specifically for the detection of homoplasic SNPs in bacteria, it could potentially be used for investigating homoplasic SNPs in other haploid organisms and viruses.

Funding information
This work was supported by a Bill and Melinda Gates Foundation grant to K.E.H. (OPP1175797). K.E.H. was supported by a Senior Medical Research Fellowship from the Viertel Foundation of Australia. S.D. was supported by an Australian Research Council Discovery Early Career Researcher Award (DE190100805). B.J.P. was supported by a Victorian Health and Medical Research Fellowship from the Department of Health and Human Services in the State of Victoria.

Acknowledgements
The authors would like to thank Dr Tami Lieberman for kindly providing the published version of the *B. dolosa* tree.

Authors and contributors
Conceptualization: K.E.H., D.J.E., B.P., S.D.; Methodology: K.E.H., D.J.E., B.P., S.D.; Software: D.J.E; Validation: D.J.E.,; Formal Analysis: D.J.E., K.E.H., B.P., S.D.; Resources: K.E.H.; Data Curation: D.J.E.; Writing (Original): D.J.E., K.E.H; Writing (Review): D.J.E., K.E.H., B.P.,S.D.; Visualization: D.J.E.; Supervision: K.E.H., B.P., S.D.; Project Administration: K.E.H., D.J.E.; Funding: K.E.H.

Conflicts of interest
The authors declare that there are no conflicts of interest.

References

1. Stern DL. The genetic causes of convergent evolution. *Nat Rev Genet* 2013;14:751–764.
2. Bailey SF, Blanquart F, Bataillon T, Kassen R. What drives parallel evolution? *Bioessays* 2017;39:e201600176.
3. Bryant J, Chewapreecha C, Bentley SD. Developing insights into the mechanisms of evolution of bacterial pathogens from whole-genome sequences. *Future Microbiol* 2012;7:1283–1296.
4. Didelot X, Walker AS, Peto TE, Crook DW, Wilson DJ. Within-host evolution of bacterial pathogens. *Nat Rev Microbiol* 2016;14:150–162.
5. Sanderson MJ, Hufford L. *Homoplasy: The Recurrence of Similarity in Evolution*. Academic Press, 1996.
6. Kimura M, Crow JF. The number of alleles that can be maintained in a finite population. *Genetics* 1964;49:725–738.
7. Croucher NJ, Page AJ, Connor TR, Delaney AJ, Keane JA, *et al*. Rapid phylogenetic analysis of large samples of recombinant bacterial whole genome sequences using Gubbins. *Nucleic Acids Res* 2015;43:e15.
8. Didelot X, Wilson DJ. ClonalFrameML: efficient inference of recombination in whole bacterial genomes. *PLoS Comput Biol* 2015;11:e1004041.
9. Hedge J, Wilson DJ. Bacterial phylogenetic reconstruction from whole genomes is robust to recombination but demographic inference is not. *mBio* 2014;5:e02158.
10. Hedge J, Wilson DJ. Practical approaches for detecting selection in microbial genomes. *PLoS Comput Biol* 2016;12:12.
11. Pupko T, Pe'er I, Shamir R, Graur D. A fast algorithm for joint reconstruction of ancestral amino acid sequences. *Mol Biol Evol* 2000;17:890–896.
12. Pupko T, Pe'er I, Hasegawa M, Graur D, Friedman N. A branch-and-bound algorithm for the inference of ancestral amino-acid sequences when the replacement rate varies among sites: application to the evolution of five gene families. *Bioinformatics* 2002;18:1116–1123.
13. Ashkenazy H, Penn O, Doron-Faigenboim A, Cohen O, Cannarozzi G, *et al*. FastML: a web server for probabilistic reconstruction of ancestral sequences. *Nucleic Acids Res* 2012;40:W580-4.
14. Yang Z. PAML 4: phylogenetic analysis by maximum likelihood. *Mol Biol Evol* 2007;24:1586–1591.
15. Sagulenko P, Puller V, Neher RA. TreeTime: Maximum-likelihood phylodynamic analysis. *Virus Evol* 2018;4:vex042.
16. Crispell J, Balaz D, Gordon SV. HomoplasyFinder: a simple tool to identify homoplasies on a phylogeny. *Microb Genom* 2019;5:e000245.
17. Jeanes C, O'Grady J. Diagnosing tuberculosis in the 21st century - Dawn of a genomics revolution? *Int J Mycobacteriol* 2016;5:384–391.
18. Votintseva AA, Bradley P, Pankhurst L, Del Ojo Elias C, Loose M, *et al*. Same-day diagnostic and surveillance data for tuberculosis via whole-genome sequencing of direct respiratory samples. *J Clin Microbiol* 2017;55:1285–1298.
19. Brown E, Dessai U, McGarry S, Gerner-Smidt P. Use of whole-genome sequencing for food safety and public health in the United States. *Foodborne Pathog Dis* 2019;16:441–450.
20. Zwickl DJ, Hillis DM. creased taxon sampling greatly reduces phylogenetic error. *Syst Biol* 2002;51:588–598.
21. Baele G, Suchard MA, Rambaut A, Lemey P. Emerging concepts of data integration in pathogen phylodynamics. *Syst Biol* 2017;66:e47–e65.
22. Baele G, Dellicour S, Suchard MA, Lemey P, Vrancken B. Recent advances in computational phylodynamics. *Curr Opin Virol* 2018;31:24–32.
23. Lees JA, Kendall M, Parkhill J, Colijn C, Bentley SD, *et al*. Evaluation of phylogenetic reconstruction methods using bacterial whole genomes: a simulation based study. *Wellcome Open Res* 2018;3:33.
24. Stamatakis A. RAxML version 8: a tool for phylogenetic analysis and post-analysis of large phylogenies. *Bioinformatics* 2014;30:1312–1313.
25. Kozlov AM, Darriba D, Flouri T, Morel B, Stamatakis A. RAxML-NG: a fast, scalable and user-friendly tool for maximum likelihood phylogenetic inference. *Bioinformatics* 2019;35:4453–4455.
26. Nguyen LT, Schmidt HA, von Haeseler A, Minh BQ. IQ-TREE: a fast and effective stochastic algorithm for estimating maximum-likelihood phylogenies. *Mol Biol Evol* 2015;32:268–274.

27. Minh BQ, Schmidt H, Chernomor O, Schrempf D, Woodhams M, *et al*. IQ-TREE 2: New models and efficient methods for phylogenetic inference in the genomic era [Preprint]. *bioRxiv* 2019:849372.
28. Huerta-Cepas J, Serra F, Bork P. ETE 3: Reconstruction, Analysis, and Visualization of Phylogenomic Data. *Mol Biol Evol* 2016;33:1635–1638.
29. Letunic I, Bork P. Interactive Tree Of Life (iTOL) v4: recent updates and new developments. *Nucleic Acids Res* 2019;47:W256–W259.
30. Yu G, Smith DK, Zhu H, Guan Y, Lam TT-Y. ggtree: an R package for visualization and annotation of phylogenetic trees with their covariates and other associated data. *Methods Ecol Evol* 2016;828–36.
31. Holt KE, McAdam P, Thai PVK, Thuong NTT, Ha DTM, *et al*. Frequent transmission of the *Mycobacterium tuberculosis* Beijing lineage and positive selection for the EsxW Beijing variant in Vietnam. *Nat Genet* 2018;50:849–856.
32. Rambaut A, Grass NC. Seq-Gen: an application for the Monte Carlo simulation of DNA sequence evolution along phylogenetic trees. *Bioinformatics* 1997;13:235–238.
33. Perrin A, Larsonneur E, Nicholson AC, Edwards DJ, Gundlach KM, *et al*. Evolutionary dynamics and genomic features of the *Elizabethkingia anophelis* 2015 to 2016 Wisconsin outbreak strain. *Nat Commun* 2017;8:15483.
34. Lieberman TD, Michel JB, Aingaran M, Potter-Bynoe G, Roux D, *et al*. Parallel bacterial evolution within multiple patients identifies candidate pathogenicity genes. *Nat Genet* 2011;43:1275–1280.
35. Farhat MR, Shapiro BJ, Kieser KJ, Sultana R, Jacobson KR, *et al*. Genomic analysis identifies targets of convergent positive selection in drug-resistant *Mycobacterium tuberculosis*. *Nat Genet* 2013;45:1183–1189.
36. Coll F, Phelan J, Hill-Cawthorne GA, Nair MB, Mallard K, *et al*. Genome-wide analysis of multi- and extensively drug-resistant *Mycobacterium tuberculosis*. *Nat Genet* 2018;50:307–316.
37. Nair J, Rouse DA, Bai GH, Morris SL. The *rpsL* gene and streptomycin resistance in single and multiple drug-resistant strains of *Mycobacterium tuberculosis*. *Mol Microbiol* 1993;10:521–527.
38. Cade CE, Dlouhy AC, Medzihradszky KF, Salas-Castillo SP, Ghiladi RA. Isoniazid-resistance conferring mutations in *Mycobacterium tuberculosis* katG: catalase, peroxidase, and INH-NADH adduct formation activities. *Protein Sci* 2010;19:458–474.
39. Juréen P, Werngren J, Toro JC, Hoffner S. Pyrazinamide resistance and *pncA* gene mutations in *Mycobacterium tuberculosis*. *Antimicrob Agents Chemother* 2008;52:1852–1854.
40. Miller LP, Crawford JT, Shinnick TM. The rpoB gene of *Mycobacterium tuberculosis*. *Antimicrob Agents Chemother* 1994;38:805–811.
41. Bakuła Z, Napiórkowska A, Bielecki J, Augustynowicz-Kopeć E, Zwolska Z, *et al*. Mutations in the *embB* gene and their association with ethambutol resistance in multidrug-resistant *Mycobacterium tuberculosis* clinical isolates from Poland. *Biomed Res Int* 2013;2013:167954.
42. Hegde SR, Rajasingh H, Das C, Mande SS, Mande SC. Understanding communication signals during mycobacterial latency through predicted genome-wide protein interactions and Boolean modeling. *PLoS One* 2012;7:e33893.
43. Joy JB, Liang RH, McCloskey RM, Nguyen T, Poon AFY. Ancestral reconstruction. *PLoS Comput Biol* 2016;12:e1004763.
44. Duchêne S, Lanfear R. Phylogenetic uncertainty can bias the number of evolutionary transitions estimated from ancestral state reconstruction methods. *J Exp Zool B Mol Dev Evol* 2015;324:517–524.