Published online 11 May 2019 *Nucleic Acids Research, 2019, Vol. 47, No. 11* ***5539–5549***
doi: 10.1093/nar/gkz361

Fast hierarchical Bayesian analysis of population structure

Gerry Tonkin-Hill [1,*], John A. Lees [2], Stephen D. Bentley [1], Simon D.W. Frost [3,4] and Jukka Corander [1,5,6]

[1]Parasites and Microbes, Wellcome Sanger Institute, Cambridge, CB10 1SA, UK, [2]Department of Microbiology, New York University School of Medicine, NY 10016, USA, [3]Department of Veterinary Medicine, University of Cambridge, Cambridge, CB3 0ES, UK, [4]The Alan Turing Institute, London, NW1 2DB, UK, [5]Department of Biostatistics, University of Oslo, Blindern 0317, Norway and [6]Helsinki Institute for Information Technology HIIT, Department of Mathematics and Statistics, University of Helsinki, Aalto FI-00076, Finland

Received March 25, 2019; Editorial Decision April 22, 2019; Accepted April 29, 2019

ABSTRACT

We present fastbaps, a fast solution to the genetic clustering problem. Fastbaps rapidly identifies an approximate fit to a Dirichlet process mixture model (DPM) for clustering multilocus genotype data. Our efficient model-based clustering approach is able to cluster datasets 10–100 times larger than the existing model-based methods, which we demonstrate by analyzing an alignment of over 110 000 sequences of HIV-1 pol genes. We also provide a method for rapidly partitioning an existing hierarchy in order to maximize the DPM model marginal likelihood, allowing us to split phylogenetic trees into clades and subclades using a population genomic model. Extensive tests on simulated data as well as a diverse set of real bacterial and viral datasets show that fastbaps provides comparable or improved solutions to previous model-based methods, while being significantly faster. The method is made freely available under an open source MIT licence as an easy to use R package at https://github.com/gtonkinhill/fastbaps.

INTRODUCTION

Identifying clusters of genetically similar individuals within a larger population is a common problem in genetics and ecology. Population structure is helpful in understanding past historical population events, conservation genetics, the analysis of invasive species and disease outbreaks. Confounding population structure must also be considered in tests for natural selection as well as genetic association studies (1–3).

Methods for clustering genotype data can generally be separated into those based on a statistical model of population structure, where observations from each underlying cluster are assumed to be drawn from a parametric population genetics model (4), and distance-based methods that rely on more general clustering techniques such as *k*-means (5). While distance-based approaches are generally faster, they are not as readily interpretable and typically do not provide group membership probabilities making distinguishing weak separation between clusters and strong population structure difficult (6). Model-based methods typically indicate the probability that an individual belongs to a certain cluster, allow for different models to be compared, and can provide estimates of uncertainty for the inferred parameters. Generally, either Bayesian or maximum-likelihood based approaches are widely used for inference in population structure models.

A commonly considered model of population structure assumes that there is a fixed number of uncorrelated underlying populations, *K*. An individual is then assumed to originate either from a single population, the no-admixture model or to carry alleles from multiple populations, which corresponds to the admixture model (4). Solutions to this problem include STRUCTURE (4,7), BAPS (8–10), Admixture (11), fast Structure (12), sNMF (13) and the method of Anderson and Thompson (14). More recently, methods that combine model-based techniques with an initial faster distance-based clustering have been proposed including snapclust (6) and hierBAPS (15). Admixture, fastStructure and sNMF solve for the admixture model and are not considered here.

A common theme of most clustering methods is that they require the underlying number of clusters, *K*, to be provided. In practice, the methods are usually run over a range of *K* and the 'best fitting' model is selected (16,17). An alternative Bayesian solution is to put a prior on the number of clusters, usually using a Dirichlet process mixture model (DPM) in an attempt to infer *K* as part of the model (18,19). While this provides a natural method for inferring an appropriate *K*, the approach is very computationally expen-

*To whom correspondence should be addressed. Tel: +44 1223 834244; Fax: +44 1223 494919; Email: gqt20@cam.ac.uk

© The Author(s) 2019. Published by Oxford University Press on behalf of Nucleic Acids Research.
This is an Open Access article distributed under the terms of the Creative Commons Attribution License (http://creativecommons.org/licenses/by/4.0/), which permits unrestricted reuse, distribution, and reproduction in any medium, provided the original work is properly cited.

Downloaded from https://academic.oup.com/nar/article/47/11/5539/5488018 by guest on 21 August 2022

sive and does not scale to large numbers of loci and individuals. The greedy stochastic optimization approach in hierBAPS (15) places a discrete uniform prior on K up to a supplied maximum value and is able to scale to larger datasets. However, as datasets become very large, inferring K using model-based clustering approaches has so far been infeasible. An example of a dataset that is too large for current model-based methods is the set of HIV-1 pol genes, which are routinely generated in clinical settings for the purpose of identifying resistance to antivirals. Despite the relatively short length of this region (typically 1200–1500 bp long), the very large number of sequences—in excess of 100 000 in public databases alone—makes choosing an appropriate value for K very challenging.

A DPM assumes a joint prior over the cluster allocation vector p, which indicates which cluster each individual is allocated to as well as the overall number of clusters K (18,20). More specifically the prior is assumed to follow a Dirichlet process where the prior joint probability of the allocation vector p and the number of clusters K is given by

$$f(p, K|\alpha, n) = \alpha^k \frac{\prod_{i=1}^{K}(n_i - 1)}{\prod_{i=1}^{n}(\alpha + i - 1)}$$

The concentration parameter α determines the degree to which individuals group together in clusters. A smaller α value leads to larger clusters. However, it has previously been shown that the choice of α has little impact on the results when the number of loci is large (20). In addition to using a Dirichlet process prior on the allocation vector and number of clusters K we assume a separate Dirichlet prior distribution on the underlying allele frequencies within each cluster parameterized by β. Thus, we draw alleles frequencies from the model by first drawing the allocation vector and the number of clusters K and then drawing the allele count data D from a Multinomial–Dirichlet distribution. The marginal distribution of counts in each cluster component is given in the 'Materials and Methods' section.

Unfortunately, methods that rely on a fully Bayesian inference approach using MCMC such as structurama are not feasible for very large datasets (19,20) with current faster approximations such as hierBAPS also failing to scale to some real world datasets (see 'Results' section). Other approximate inference methods for DPM models include variational inference (21), fast search (22), sequential updating and greedy search (SUGS) (23,24) and expectation propagation (25). While these methods offer substantial speed improvements over a full MCMC-based approach none of them are able to take advantage of the phylogenetic structure present in the datasets we consider. Instead, we leverage the inherent hierarchical structure of prokaryotic population genomic datasets to provide substantial speed improvements in inference.

The Bayesian hierarchical clustering (BHC) algorithm (26) presents an alternative approach that approximates a DPM while guaranteeing a solution in polynomial time. The method performs a version of agglomerative bottom up clustering using a Dirichlet process to account for uncertainty in the data and Bayesian model selection to decide which clusters should be merged at each step. While the approach has been shown to succeed in clustering text documents, microarray data and even electrical demand profiles (26–28), it has yet to be successfully applied to the problem of identifying population structure. A reason for this is that the hyperparameters, β, of the model defining each component of the mixture are set to be proportional to the counts of each allele at each locus in the complete dataset. In practice setting the hyperparameters to be proportional to the entire dataset leads to an overpartitioning of population genetic data whereby the algorithm identifies a very high number of clusters (see 'Results' section). A common alternative prior in population structure studies is to use a symmetric Dirichlet prior (4). However, at the lowest levels of the hierarchy a symmetric prior will lead to different clustering combinations having the same likelihood. This greatly reduces the ability of the algorithm to accurately cluster sequences at the lower levels. In contrast, the hierBAPS algorithm relies on a fast initial clustering using complete-linkage agglomerative clustering to provide an initial partition of the dataset. hierBAPS then performs a greedy stochastic optimization procedure to identify the local maximum a posteriori (MAP). By taking advantage of a fast initial clustering approach similar to that used in hierBAPS we are able to place a symmetric or BAPS prior on the mixture components, enabling the BHC algorithm to distinguish between different combinations of clusters.

Though methods such as hierBAPS and snapclust have allowed for analyses to scale to much larger datasets than previous approaches, they remain in practice inapplicable to the currently emerging datasets comprising tens of thousands or even hundreds of thousands of sequences, where large numbers of underlying clusters may be present. Here, through incorporating ideas from both BHC and hierBAPS, we produce an efficient inference solution to the no-admixture model for very large datasets when the underlying number of clusters may be in the tens or hundreds.

MATERIALS AND METHODS

Initial clustering

Similar to hierBAPS, but unlike the original BHC algorithm, fastbaps begins by generating a fast initial clustering of the data using a value for K much larger than expected (K_{init}). This has the advantage of both reducing the complexity of the Bayesian agglomerative stage while allowing for symmetric priors to be used. Starting with a multiple sequence alignment in FASTA format, we first generate a pairwise single nucleotide polymorphism distance matrix before clustering using Ward's agglomerative clustering (29). This hierarchy is cut to generate K_{init} clusters. The hierarchy is also used to estimate the hyperparameters β of the individual mixture component priors. Very large datasets make the calculation of the initial pairwise distance matrix prohibitive. To counter this limitation, for datasets with more than 10 000 individuals we produce the initial clustering by first performing a Principal Component Analysis (PCA) before clustering the first 50 principal components using a fast hierarchical algorithm such as genie or the memory efficient Ward algorithm in the fastcluster package (30,31). This removes the need to calculate the complete distance matrix. The results were found to be robust to the number of principal components used with this approach

Downloaded from https://academic.oup.com/nar/article/47/11/5539/5488018 by guest on 21 August 2022

found to provide near identical results to the distance matrix approach on the smaller datasets. This is expected in general as the PCA step is also only used at the lower levels of the clustering hierarchy and thus its impact is minimal on the final clustering solution.

Bayesian hierarchical clustering

Given a collection of small clusters and their respective hierarchies T_i where $i \in \{1..K_{init}\}$, the BHC algorithm of Heller and Ghahramani (26) proceeds in a similar fashion to traditional agglomerative clustering. In the place of a distance metric, Bayesian hypothesis testing is conducted at each level to decide which clusters to merge. Let $\mathcal{D} = x^{(1)}, ..., x^{(n)}$ describe the entire dataset of n samples and $\mathcal{D}_i$ the set of points at the leaves of the subtree T_i. We initialize the algorithm with the small subtrees obtained using the fast clustering step as shown in Figure 1. At each subsequent stage we then consider merging all pairs of existing trees. If trees T_i and T_j are merged into a combined tree T_k then the set of points corresponding to the new tree is $\mathcal{D}_k = \mathcal{D}_i \cup \mathcal{D}_j$. At each merge we compare two hypotheses. The merged hypothesis, which we denote $\mathcal{H}_1^k$, is that all data in $\mathcal{D}_k$ were generated identically and independently from the same probabilistic model $p(\boldsymbol{x}|\theta)$ with unknown parameters θ. In our case this model is a multinomial, with parameters θ = (n, ϕ). We also specify a Dirichlet prior over the parameters ϕ with hyperparameters β. Thus, we can write the probability of the data $\mathcal{D}_k$ under $\mathcal{H}_1^k$ as

$$p(\mathcal{D}_k|\mathcal{H}_1^k) = \int p(\mathcal{D}_k|\theta)p(\theta|\beta)d\theta$$

Assuming a multinomial–Dirichlet distribution, this integral is tractable and we can rewrite the above equation as:

$$p(\mathcal{D}_k|\mathcal{H}_1^k) = \prod_{j=1}^{N_L} \frac{\Gamma(\sum_l \beta_j)}{\Gamma\left(\sum_l(\beta_j + n_{jl})\right)} \prod_{l=1}^{N_{A(j)}} \frac{\Gamma(\beta_j + n_{jl})}{\Gamma(\beta_j)}$$

Where N_L is the number of loci, $N_{A_{(j)}}$ is the number of possible alleles at loci j and n_{jl} is the number of copies of allele l at locus j. β_j is the corresponding Dirichlet prior hyperparameter.

The alternative hypothesis to merging two clusters ($\mathcal{H}_1^k$) is that the data $\mathcal{D}_k$ have two or more clusters. Summing over all the possible partitions of $\mathcal{D}_k$ into two or more groups is in practice intractable. However, by only considering partitions that are consistent with the subtrees T_i and T_j, we can use recursion to quickly compute this sum. Under the restriction of remaining consistent with the subtrees, the probability of the data under the alternative hypothesis $\mathcal{H}_2^k$, is just the product over the subtrees $p(\mathcal{D}_k|\mathcal{H}_2^k) = p(\mathcal{D}_i|T_i)p(\mathcal{D}_j|T_j)$ (see Figure 1) where the probability of the data given the tree is given by

$$p(\mathcal{D}_k|T_k) = \pi_k p(\mathcal{D}_k|\mathcal{H}_1^k) + (1 - \pi_k)p(\mathcal{D}_i|T_i)p(\mathcal{D}_j|T_j)$$

Here, the first term for each initial cluster is defined recursively using the fast clustering based hierarchy with initial prior for the merged hypothesis set to $\pi_0 = 1$. The prior for the merged hypothesis $\pi_k := p(\mathcal{H}_1^k)$, can then be computed bottom-up as described in Heller and Ghahramani, (26) and is a deterministic function of the concentration parameter of the DPM. In practice, for the size of datasets we are considering, the actual value of α makes very little difference as the prior π_k is dominated by the factorial on the number of points within a cluster. Similar to Savage *et al.*, (28), we fix the value of α, the concentration parameter of the DPM, rather than keeping it as a user-tunable hyperparameter in the resulting clustering algorithm.

At last, given the probabilities of each hypothesis described above we can calculate the posterior probability of the merged hypothesis $r_k := p(\mathcal{H}_1^k|D_k)$ using Bayes' rule

$$r_k = \frac{\pi_k p(\mathcal{D}_k|\mathcal{H}_1^k)}{\pi_k p(\mathcal{D}_k|\mathcal{H}_1^k) + (1 - \pi_k)p(\mathcal{D}_i|T_i)p(\mathcal{D}_j|T_j)}$$

We use this quantity to greedily decide which two subtrees to merge. After generating a final hierarchy we then use the same quantity to identify which merges were justified and cut the tree when $r_k < 0.5$.

Hyperparameter selection

As described in Heller and Ghahramani (26), for any given set of hyperparameters, the root node of the hierarchy approximates the probability of the data given that setting of the hyperparameters. By leveraging this, we use the marginal likelihood of the root node $p(\mathcal{D}|T)$ to perform model comparisons for different settings of the hyperparameters. We found that different settings of the hyperparameter α had a negligible impact on the final clustering due to the dominance of the factorial component of π_k, which provided a strong motivation to fix the hyperparameter at the value $\alpha = 1$. Similar to Heller and Ghahramani (26) and Savage *et al.,* (28), we use golden section search as provided in R's 'optimize' function to select the β parameters and thus the variance of the Dirichlet prior. Unlike previous approaches, we did not set the β parameters to be proportional to the discrete total counts for the entire dataset. Instead we used either a symmetric prior, similar to Pritchard *et al.*, (4) or the non-informative prior $\beta_j \propto \frac{1}{N_{A_{(j)}}}$ as used in hierBAPS and suggested in Anderson and Thompson (14). These priors are referred to as optimized symmetric, and optimized BAPS priors hereafter. These priors were found to generally outperform the prior used in the original BHC algorithm. To increase the efficiency in optimizing the variance of the Dirichlet prior, we relied on an initial hierarchy produced using Ward's agglomerative clustering. The marginal likelihood of the root node was then used to select an appropriate scaling for the β hyperparameters. We subsequently refer to the combination of fast cluster initialization, BHC and prior hyperparameter optimization using a symmetric or BAPS based prior as 'fastbaps'.

Conditioning on a pre-computed hierarchy

Given a pre-computed hierarchy or phylogeny, we can use the recursion method described previously to decide when merging sub-clades of the tree is justified according to the DPM model. This provides a parametric model based alternative to previous tree partition algorithms such as Cluster

Downloaded from https://academic.oup.com/nar/article/47/11/5539/5488018 by guest on 21 August 2022

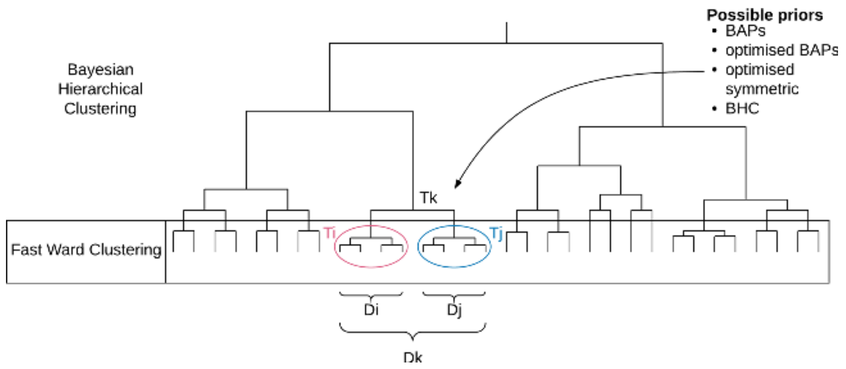

Figure 1. A diagram outlining the fastbaps algorithm. An initial clustering is performed by default using agglomerative clustering with the Ward linkage method. Subsequently the merge with the highest posterior probability is chosen. The final clustering is then determined by only accepting merges with a posterior probability >0.5

Picker (32), also relaxing the computationally intensive requirement for bootstrap replicates of the tree being partitioned. It allows us to identify a partition of the tree into clades that maximizes the marginal likelihood of a DPM model given a hierarchy. As the hierarchy is pre-calculated, the approach is highly efficient and has a linear computational complexity in the number of nodes in the hierarchy. By leveraging the sparse matrix data structures in fastbaps and coupling them with the efficient implementation of Ward's hierarchical clustering in the fastcluster package (31), we are able to generate clusterings of comparable quality to the full fastbaps algorithm very quickly (see Figure 4 and Supplementary Figure S8).

Cluster stability using the Bootstrap

In order to determine the sensitivity of the resulting clusters to the alignment, we implemented a simple bootstrap procedure. For each bootstrap replicate, the loci are sampled with replacement and the fastbaps clustering algorithm is run. A binary similarity matrix is then produced as described in Strehl and Ghosh (33) where for each pair of sequences (i,j), the (i,j)th entry in the matrix is 1 if the ith and jth sequences are clustered together and 0 otherwise. As this matrix is invariant to label switching (34) and its dimension is invariant to the underlying number of clusters, the sum of the resulting matrices can be used to provide an indication of the stability of a given clustering of two isolates. This can be plotted in a heatmap alongside the dendrogram produced using the full algorithm to illustrate the robustness of the final clustering.

Implementation

The fastbaps method is implemented using R and C++ and is available as an R package under the MIT open source licence at https://github.com/gtonkinhill/fastbaps. It can be run on Unix, Mac, or Windows operating systems and takes a multiple sequence alignment as input. The results can easily be parsed in R allowing for the generation of informative plots and further processing. The sparse matrix data structure generated by the package can also be used in other analysis types such as PCA or *k*-means. All code to reproduce the simulations and results in this paper is available at https://github.com/gtonkinhill/fastbaps_manuscript.

Simulation and evaluation

To compare the results of different algorithms, population structure was simulated using scrm as part of the Coala R package (35,36). The number of underlying populations, recombination rate and migration rate were varied as described in Supplementary Table S1. Each parameter combination was run for three replicates providing 90 test datasets for analysis. A more detailed description of how the simulations were run can be found in the supplementary R notebook. Three clustering algorithms were considered in addition to the fastbaps approach. Snapclust and hierBAPS are available as R packages and were compared with fastbaps along with the fully Bayesian Structure algorithm (4,6,15,37). Snapclust was run in parallel with the number of underlying clusters, *K*, varied between 2 and 30. The best fitting model was then chosen using either the Bayesian Information Criterion (BIC) or the Akaike Information Criterion (AIC). HierBAPS was run with 50 initial clusters. At last, Structure was run with the no admixture model with 100 000 burn in iterations and 500 000 sampled iterations with three separate starting conditions. The run with highest likelihood was kept. The underlying number of clusters was fixed to the simulated *K* when running Structure to obtain a result in a reasonable amount of time, which nevertheless gives the method an unrealistic advantage against the alternatives which are fast enough to identify *K*.

We implement a number of strategies to validate our clustering approach and demonstrate its ability to generate biologically informative clusters. That is clusters that identify broad population structure that allow us to divide large datasets into smaller related subsets for further investigation or to use as a covariate in phenotype association studies. We show that similar to previous programs our clustering approach is able to retrieve the true clusters from simulated data while maintaining a considerable advantage in terms of speed. We compare our algorithm to alternative approaches that solve similar problems such as phylogeny reconstruction using Fasttree and dimension reduction using UMAP. We show that our results are consistent with these algorithms while using a parametric model

Downloaded from https://academic.oup.com/nar/article/47/11/5539/5488018 by guest on 21 August 2022

of population structure which provides additional insight over clustering on the outputs of either Fasttree or UMAP alone. In addition, using the outputs of Fasttree we compare our algorithm with other competing approaches over a wide range of real world datasets using a distance function which accounts for both within-cluster and between-cluster similarity, penalizing solutions that achieve a high within-cluster similarity by producing a large number of very tightly connected clusters. At last, to investigate our results on the large HIV dataset we consider the concordance of our inferred clusters with the geography and sub-type of the HIV pol genes considered. In summary, our comparisons aimed to provide a wide range of biologically relevant scenarios where the statistical and computational choices made in the different methods may result in biologically relevant variation in the data partitions and thus enable an assessment of the general applicability and robustness of a particular method.

To compare the simulated populations with those inferred by the different clustering algorithms, we used the Fowlkes–Mallows index (38). This accounts for both false positives and false negatives in determining the similarity between two clusterings and is defined as

$$\mathrm{FM} = \sqrt{\frac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FP}}\frac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FN}}}$$

Here *TP* is the number of pairs of individuals that appear together in both clusterings, *FP* is the number that appear together in the inferred clusters but not the simulated and *FN* are those pairs that originated from the same population in the simulated data but are separated in the inferred clusters. Thus a clustering that perfectly matches the simulation will receive a Fowlkes–Mallows index of 1.

To compare with real data, six bacterial and two viral datasets were considered. The sets were chosen to cover a diverse range of pathogen species. The bacterial sets included the enteric bacteria *Escherichia coli*, the gram negative respiratory pathogen *Haemophilus influenzae*, as well as the firmicutes *Streptococcus pneumoniae*, *Listeria monocytogenes* and *Staphylococcus aureus* (39–43). The two viral datasets comprised a subset of sequences from the recent Ebola outbreak as well as a global dataset of over 110 000 HIV partial pol genes (positions 1–1497) obtained from the Los Alamos HIV Database (44,45), with associated metadata on subtype and country of sampling. A summary of the size of each dataset is given in Supplementary Table S2. As there is no gold standard truth set for the real datasets we instead chose to compare the inferred clusters with a phylogeny built using Fasttree v2.1.10 (46). Here, we counted the number of pairs of isolates within each cluster that were inconsistent with the phylogeny in that there was an isolate belonging to another cluster within the clade representing their most recent common ancestor. These were considered ‘false positives’. This count was then divided by the total number of possible pairs within each cluster to give an indication of the error rate of the clustering (assuming the phylogeny to be correct). As the number of possible pairs increases with cluster size, this approach appropriately penalizes clustering solutions with too many clusters.

RESULTS

Fastbaps accurately clusters previously intractable viral and bacterial datasets containing thousands of samples

To investigate the performance of fastbaps on very large datasets, we analyzed a dataset of over 110 000 HIV partial pol genes downloaded from the Los Alamos National Laboratories HIV Sequence Database (http://www.hiv.lanl.gov). The large number of sequences in this set presents a significant challenge to most model-based techniques as determining even the range of values of K is not straightforward. Additionally, we investigated the success of the algorithm on over 3100 pneumococcal genomes from the Maela refugee camp in Thailand (39). While this dataset has fewer genomes, the large number of variable sites (284 194) makes this dataset very challenging for all current methods.

Figure 2 illustrates the resulting clusters inferred by fastbaps using the optimized BAPS prior on both the pneumococcal and HIV datasets. We made use of the UMAP dimensionality reduction method as implemented in the umap-learn python package to project the isolates onto a 2D plane (47). UMAP is a non-linear dimensionality reduction technique that has been shown to perform well on large biological datasets including visualizing large genetic population datasets as well as in the analysis of single cell RNAseq data (48,49). As the other model based-methods considered here were unable to produce results on either of these datasets in under a week, we compared the resulting clusters to those inferred using k-means. We used both the K identified by fastbaps as well as the K found using the ‘elbow method’ (50) when running the k-means algorithm. Figure 2A indicates that fastbaps mostly corresponds with the groupings formed in the UMAP projection. The visualization also demonstrates the utility of investigating the variability of the level of clonality across the clusters, as the layout of some clusters is more scattered spatially, reflecting the varying influence of recombination on the relatedness of pneumococcal strains. While the k-means algorithm using the fastbaps inferred K ($K = 79$), provides a similar result, the K inferred using the elbow method leads to many inferred clusters being separated across the projection (Supplementary Figures 1 and 2). This highlights the advantages of using model-based techniques when investigating genetic datasets.

The fastbaps algorithm provides a cleaner clustering of the HIV dataset as seen in Figure 2B. The large group of isolates in the center of the UMAP projection has been assigned to one cluster. Conversely, k-means is unable to distinguish this group, and instead partitions it into many smaller overlapping clusters (Supplementary Figures S3 and 4). Similar to the pneumococcal dataset, when the lower value of K identified using the elbow method is used ($K = 21$), the resulting clusters are separated out over the UMAP projection (Supplementary Figure S5).

As HIV is highly recombinant, comparing the clustering of such a large global dataset to a phylogeny is unlikely to be informative. Instead, to further investigate the quality of the fastbaps clustering we examined the HIV subtype composition (inferred using a variety of methods, and including circulating recombinant forms as well as non-recombinant

Downloaded from https://academic.oup.com/nar/article/47/11/5539/5488018 by guest on 21 August 2022

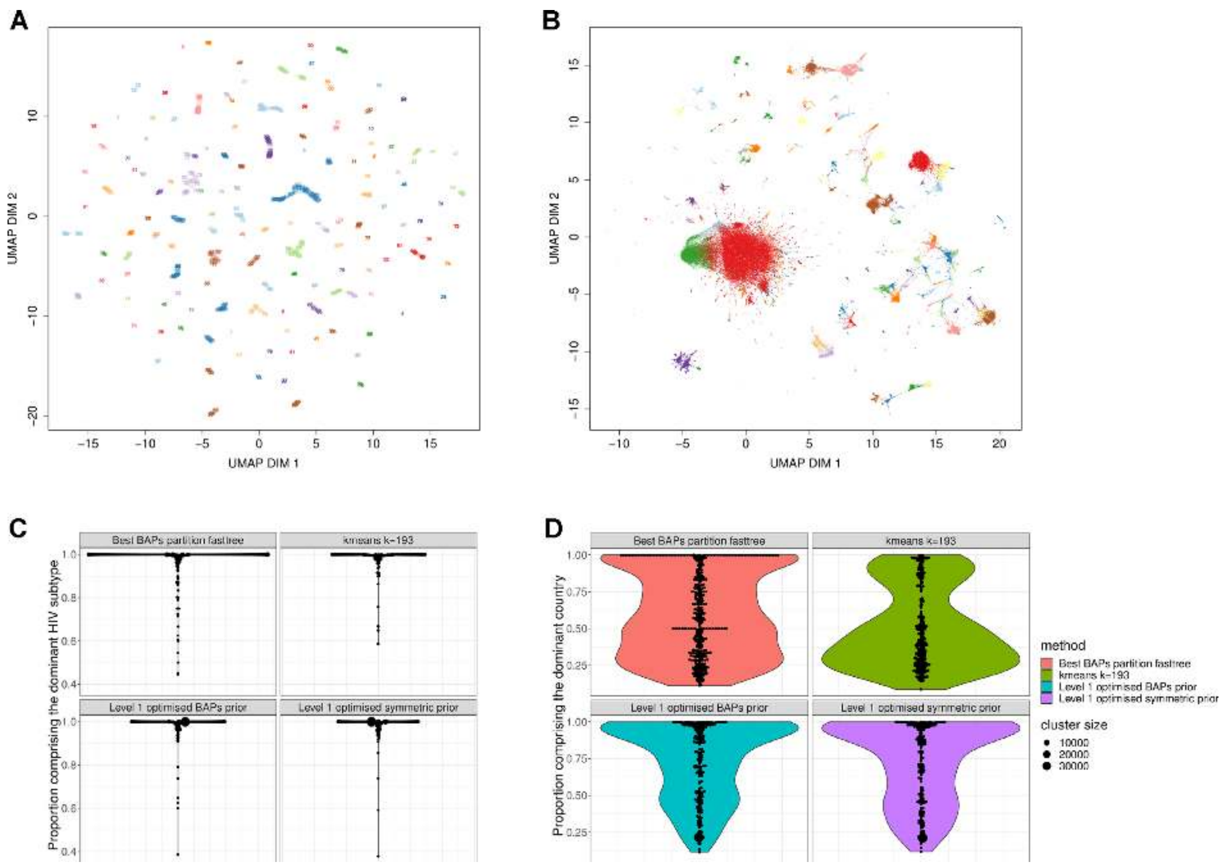

Figure 2. (**A**) A UMAP projection of over 3100 pneumococcal genomes from Maela Thailand. The points are labeled and colored by the cluster inferred using fastbaps with the optimized BAPS prior. In general the labeling appears to agree with the grouping observed in the UMAP projection. (**B**) Over 110 000 HIV-1 pol gene sequences from the Los Alamos public database. Sequences are colored by their fastbaps inferred cluster. The large red cluster in the center contrasts with the noisy clustering observed using *k*-means (Supplementary Figure S3). The same plot with points numbered with their corresponding cluster is shown in Supplementary Figure S4. (**C**) Violin plots indicating the proportion of each inferred cluster that make up its most dominant subtype. Cluster size is represented by the size of a point and the width of the violin plot is proportional to the total number of clusters inferred. The fastbaps algorithm with the optimized BAPS prior provides a clustering that is most consistent with the underlying HIV subtypes while not overly segmenting the dataset. (**D**) Violin plots indicating the proportion of each inferred cluster that make up its most dominant country. Interestingly, the fastbaps approach is able to provide clusters that are more consistent with the underlying geography than alternative approaches such as *k*-means.

subtypes) as well as the distribution of the countries of origin of the isolates found in each cluster. Figure 2 indicates the proportion of each cluster that was dominated by a single subtype or any of its recombinant forms. Thus a cluster that is composed entirely of one subtype will have a proportion of 1. The figure indicates that despite having the same number of clusters as *k*-means, the fastbaps algorithm with both the optimized BAPS and optimized symmetric prior identifies larger clusters of purely one subtype.

This suggests the algorithm is identifying legitimate clusters within the dataset without over-partitioning the data. A similar comparison was made to investigate the proportion of each cluster that was dominated by a single country. Figure 2D indicates that fastbaps provides clusters that are more consistent with the underlying geography where as the *k*-means is dominated by many clusters that comprise isolates from many countries. Fastbaps was also used to partition a phylogeny. The resulting clustering produced a larger number of clusters due to the restriction of the phylogeny but the resulting clusters were more consistent by subtype and geography than those inferred using *k*-means.

Simulations show that fastbaps cluster accuracy is the same as previous methods

In order to compare with different methods we made use of simulations where the true clustering solution was known. The Structure, hierBAPS, snapclust and fastbaps algorithms all achieved similar levels of accuracy on the simulated datasets (Figure 3). As expected, an increase in the migration rate between the simulated demes led to a subsequent decrease in the ability of the methods to accurately identify the underlying population structure. Additionally, a lower recombination rate led to poorer results, which is probably due to the algorithms correctly identifying additional phylogenetic structure within each cluster. Snapclust using AIC to select *K* as well as the fastbaps algorithm with the optimized BAPS prior achieved the best results over the simulated datasets. hierBAPS achieved a similar level of accuracy to the fastbaps algorithms using both the original (unscaled) BAPS prior as well as the optimized symmetric prior. Structure and Snapclust with BIC performed worst over the simulations. This was despite the MCMC chains of each Structure run showing good convergence charac-

Downloaded from https://academic.oup.com/nar/article/47/11/5539/5488018 by guest on 21 August 2022

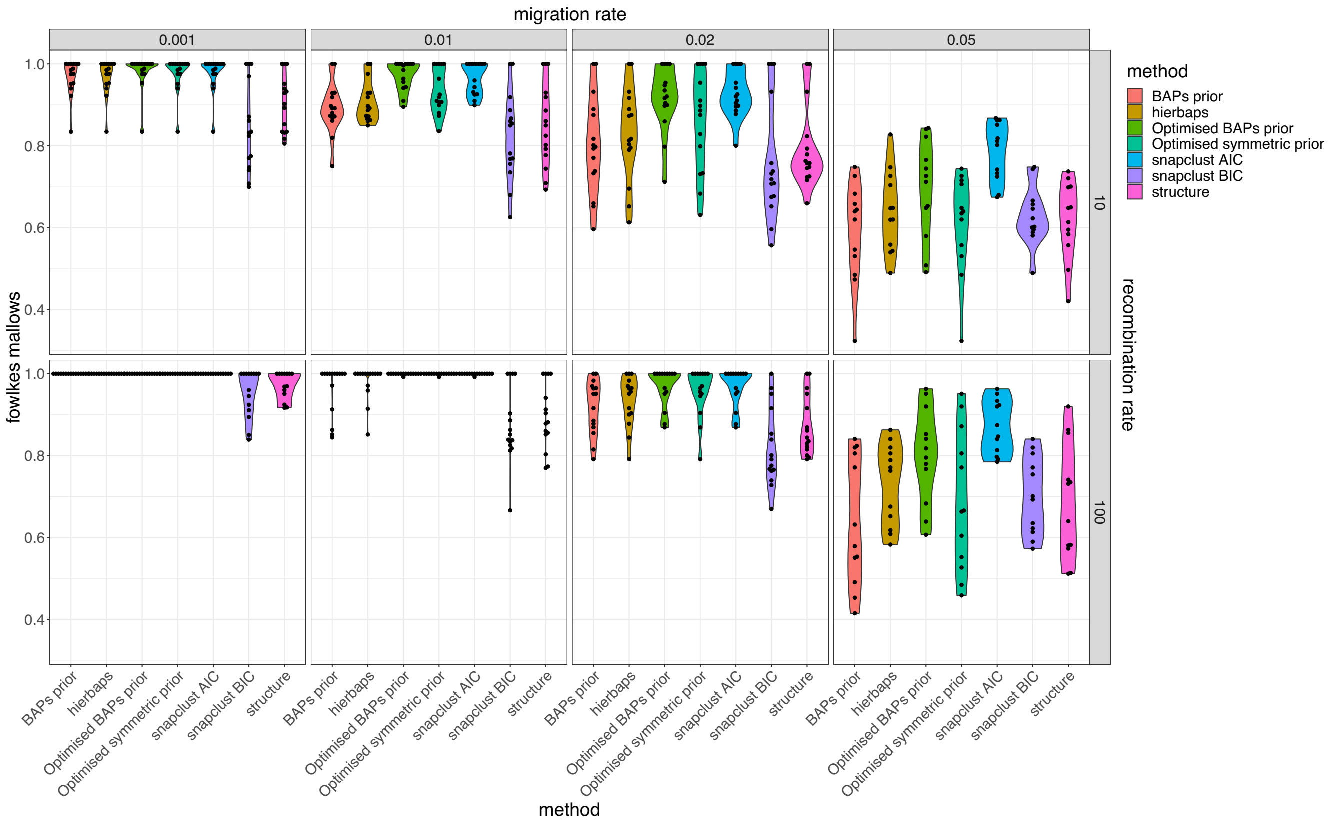

Figure 3. Violin plot indicating the Fowlkes-Mallows distance between the simulated clusters and those inferred by the different algorithms. Results for two distinct recombination rates are shown (10 and 100) as well as four different migration rates. The underlying number of clusters varied between 5 and 25. The plot indicates that as the migration rate increases the accuracy of the algorithm falls. Additionally, a lower recombination rate leads to lower accuracy, which is likely due to the algorithms identifying additional population structure within each simulated deme.

teristics (Supplementary Figure S6). A closer investigation of the Structure results indicated that it tended to allocate empty clusters despite being given the correct *K*. This suggests that both Snapclust with BIC and Structure tend to be overly conservative in estimating the number of underlying clusters *K*.

Fastbaps is efficient and accurate on a diverse range of viral and bacterial datasets

To investigate the ability of each algorithm to detect population structure in real datasets, we made use of six additional bacterial and viral datasets of varying sizes and complexity (Supplementary Table S2). Figure 4 indicates the resulting clusters inferred by the most promising algorithms from the simulation analysis on both a *Neisseria meningitidis* and *Haemophilus influenzae* dataset. Here, we also provide comparisons with the prior from the original BHC algorithm and a partition of a Fasttree phylogeny using fastbaps. On the *H. influenzae* dataset, fastbaps using the optimized BAPS prior provided a clustering that was the most consistent with the phylogeny. Both the hierBAPS and snapclust solutions included polyphyletic clusters while the BHC population mean based prior gave a similar result to the optimized BAPS solution. It should be noted that the initial fast clustering step was the same for both the BHC and optimized BAPS clusterings. The BHC prior failed to perform adequately on the large *N. meningitidis* dataset. Here, the BHC prior led to a highly partitioned solution with over 80 very small clusters. While such a high number of clusters may be valid it does not aid in the biological interpretation of the dataset, where we are generally interested in broad population structure that allows us to divide large collections into smaller subsets that can be more closely investigated or controlled for in phenotype association studies. Conversely, the optimized BAPS prior led to a solution of similar quality to both hierBAPS and snapclust. Uncertainty in the Fasttree phylogenies was considered by comparing them with a bootstrapped phylogeny built using IQTREE v1.6.5 with the results found to be very similar. In both of these datasets the partition of the phylogeny using fastbaps provided a solution similar to that found using the optimized BAPS prior indicating it is an appropriate choice if a user's goal is to simply partition a pre-calculated phylogeny. In addition, the uncertainty in the fastbaps clusters could be explored using the bootstrap procedure described in the 'Materials and Methods' section as shown in Supplementary Figure S7.

Figure 4C summarizes the results of running the algorithms on all these datasets except the large HIV and pneumococcal collections. The figure indicates the number of pairs of isolates that are clustered together using the population structure algorithms but that are inconsistent with

Downloaded from https://academic.oup.com/nar/article/47/11/5539/5488018 by guest on 21 August 2022

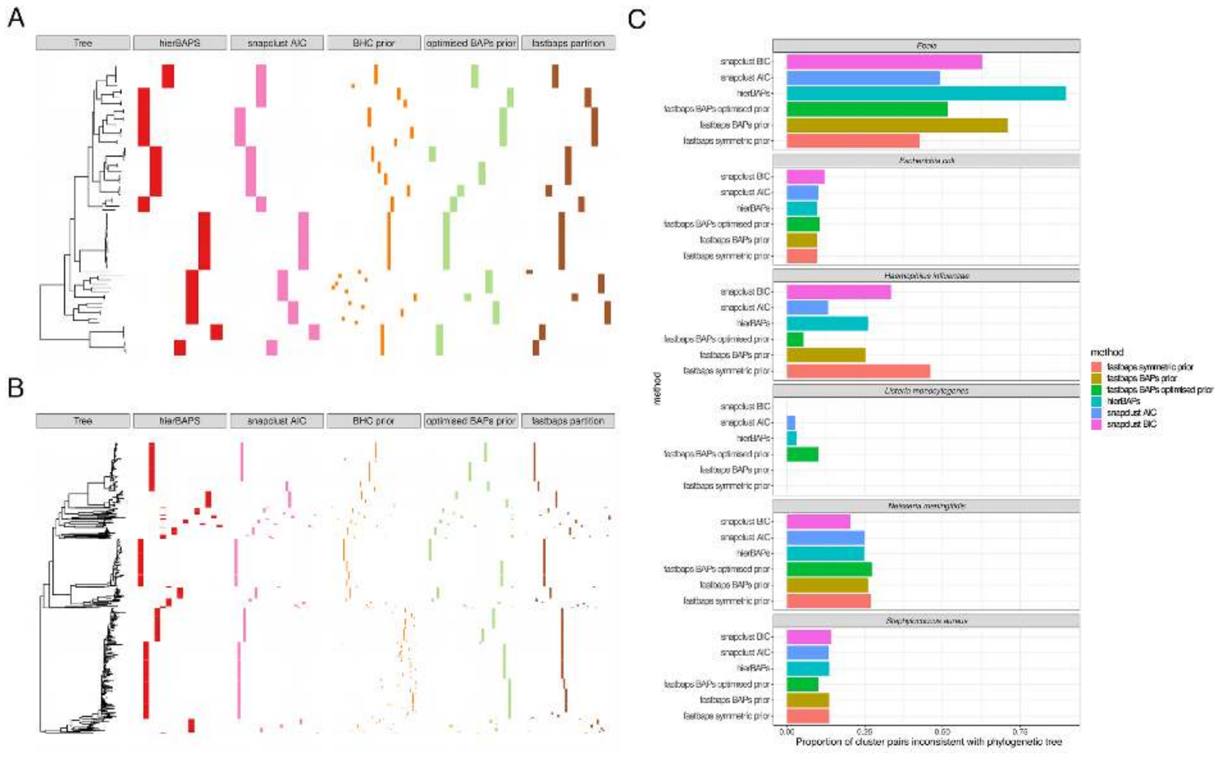

Figure 4. (**A** and **B**) The resulting clustering inferred by hierBAPS, snapclust and fastbaps with both the original BHC prior and the optimized BAPS prior on both *Staphylococcus aureus* (A) and *Neisseria meningitidis* (B). The clusters are shown in comparison to a phylogeny generated using Fasttree. The final clustering indicates a partition of the Fasttree phylogeny using the fastbaps algorithm and thus is constrained to be consistent with the phylogeny. (**C**) The proportion of isolate pairs that appear in the same inferred cluster but have isolates from a separate cluster in the clade represented by their most recent common ancestor in the Fasttree phylogeny. This provides an indication of the error of each algorithm assuming the phylogeny generated by Fasttree is correct. The fastbaps algorithm using either the optimized BAPS or symmetric prior outperforms the other methods on five out of six datasets.

the phylogeny. Assuming the Fasttree phylogeny is accurate, this provides a measurement of error for each of the runs. The fastbaps algorithm with either the optimized symmetric or optimized BAPS priors achieves a superior clustering on five out of six of the datasets. On the *N. meningitidis* dataset the snapclust algorithm using the BIC model selection criteria outperforms the other methods. The BIC model selection technique more heavily penalizes solutions with a larger number of clusters and was in the simulations shown to frequently result in too few clusters. It is unlikely that the other methods are over-partitioning these data relative to snapclust with BIC. Hence, the outlying result for *N. meningitidis* could rather be due to inaccuracies in the Fasttree phylogeny, since *N. meningitidis* has a high recombination rate.

The computational run times and maximum memory requirements of each of the algorithms are shown in Tables 1 and 2, respectively. The tables indicate that fastbaps often performs more than an order of magnitude faster than snapclust and hierBAPS which themselves significantly outperform Structure in computational efficiency (6,51). Part of this speed is due to the use of highly optimized sparse matrix libraries in fastbaps. As snapclust must be run for all values of K that are to be considered, its computational performance is directly tied to the maximum K chosen. Here, snapclust was run for a maximum $K = 50$ as this corresponded to a sensible upper bound for the *E. coli* dataset. Reducing the maximum K to use would improve the timing of the snapclust algorithm at the expense of failing to explore higher dimension solutions. However, Table 1 indicates that the fastbaps algorithm achieves a superior runtime compared with snapclust even for a single value of K, while automatically determining an appropriate number of clusters. Depending on the dataset, the prior optimization step in fastbaps can take longer than running the complete algorithm. Thus if a short run-time is a necessity, running the method with a fixed prior is available as an option. At last, if a hierarchy is already available, either through the use of a traditional phylogeny reconstruction algorithm such as Fasttree or via agglomerative clustering a partition using the fastbaps algorithm can be achieved with a complexity linear in the number of sequences. Fastbaps was able to partition a phylogeny of the full HIV dataset built using Fasttree (46) including reading in the data in 45.8 s. Supplementary Figure S8 indicates the running time of the different modes of fastbaps excluding loading each dataset. The phylogeny conditioned mode can be seen to scale linearly, while the full fastbaps mode scales quadratically with the number of samples. Genie is a sub-quadratic hierarchical clustering method and by fixing the number of initial clusters Supplementary Figure S8 indicates we are able to achieve a sub-quadratic clustering algorithm. However, as the initial number of sub-clusters is fixed this mode would be most

Downloaded from https://academic.oup.com/nar/article/47/11/5539/5488018 by guest on 21 August 2022

Table 1. Total CPU time (seconds) of the different algorithms

Dataset	N	L	fastbaps BAPS prior	fastbaps optimized BAPS prior	fastbaps optimized symmetric prior	rhierbaps	snapclust	snapclust single K
Ebola	1610	2279	9.93	16.52	16.24	11556.36	686.18	45.8
Escherichia coli	1508	241 750	282.97	1210.3	1155.3	515913.93	498433.4	15007.32
Haemophilus influenzae	75	113605	9.36	30.85	33.4	9018.08	13472.79	277.64
HIV	118 091	1497	39718.89	55012.17	55067.16	NA	NA	NA
Listeria monocytogenes	128	150 759	22.86	78.39	68.36	18787.03	29257.58	392.6
Neisseria meningitidis	882	87 730	77.71	278.62	283.58	110312.24	76361.75	1900.13
Staphylococcus aureus	284	50 104	12.18	39.82	41.26	6514.21	35322.14	212.02
Streptococcus pneumoniae	3156	392 524	794.99	3822.11	3998.63	NA	NA	NA

Only fastbaps was able to run in a week for the Pneumococcal and HIV datasets.

Table 2. Maximum memory usage of the different algorithms (Mb)

Dataset	N	L	fastbaps BAPS prior	fastbaps optimized BAPS prior	fastbaps optimized symmetric prior	rhierBAPS	snapclust	snapclust single K
Ebola	1610	2279	411.64	468.64	487.32	1102.51	519.71	498.44
Escherichia coli	1508	241 750	4290.98	4559.68	5178.42	15194.38	54714.39	45680.39
Haemophilus influenzae	75	113 605	394.56	375.84	382.09	852.5	4148.99	2113.79
HIV	118091	1497	3465.45	6511.35	7409.47	NA	NA	NA
Listeria monocytogenes	128	150 759	791.76	842.88	823.48	1400.69	5814.35	5194.14
Neisseria meningitidis	882	87 730	1439.4	1477.06	1382.16	4231.05	13779.11	10822.19
Staphylococcus aureus	284	50 104	384.85	387.04	396.44	742	3152.37	2423.9
Streptococcus pneumoniae	3156	392 524	3915.3	5790.93	5710.96	NA	NA	NA

useful in cases where the user expected a large amount of redundancy in the dataset.

DISCUSSION

Finding clusters of related sequences present in a genetic alignment is a critical first step for many genetic and ecological analyses, allowing targeted sub-analysis and determination of the structure of the population when external classifications are not present. Model-based clustering methods are attractive due to their ability to hierarchically find highly specific clusters, estimate meaningful clustering parameters incorporating uncertainties, compare different model fits and produce probabilities of cluster assignment. Typical cohort sizes now range between thousands and tens of thousands of samples, growing even larger if combined with previous cohorts. However, the ability of methods to fit a clustering model to these alignments has not increased in step with these increases in alignment size (6,37), with even modestly sized datasets requiring gene-by-gene or distance based approaches to determine clusters (5,52,53). Model-based approaches have also proved impractical in surveillance settings, where the continuous addition of small numbers of new sequences would require refitting the entire model.

By leveraging ideas from both the BHC (26) and the hierBAPS (15) algorithms, along with a fast sparse matrix based implementation, fastbaps enables efficient model-based clustering of large alignments that were previously infeasible to analyze using existing methods. Our approach enjoys all of the advantages inherent in model-based approaches and can produce comparable or higher quality clusters than previous methods. Additionally, our algorithm is able to rapidly partition pre-computed phylogenies. This provides an attractive alternative approach for clustering genetic data when a phylogeny is available. The significant acceleration of inference provided by fastbaps enables the use of bootstrap replicates, allowing for the sensitivity of the resulting clusters to the alignment to be investigated in manner that has traditionally not been possible for model-based clustering methods on large alignments.

We verified our new approach by comparing its performance with other algorithms on both simulated data and eight real datasets. We show that as well as a considerable increase in the speed of the algorithm, we are able to achieve comparable accuracy, often outperforming previous model-based approaches. The speed of our new approach allowed us to cluster a large HIV sequence dataset containing over 100 000 sequences. The resulting clusters have higher concordance with HIV subtypes and geography than an alternative approach using *k*-means, while providing a principled method for selecting the underlying number of clusters; a major limitation of *k*-means.

While our method enables the clustering of very large alignments, its complexity is still tied to the initial hierarchy generation and thus is $O(n^2)$ or possibly $O(nlog(n))$ if single linkage hierarchical clustering is used. After the initial hierarchy is generated, the remaining BHC is of the order $O(lm^2)$ where m is the number of clusters generated in the first stage and l is the number of variable sites. Thus, in the future, as alignments begin to comprise tens of millions of sequences, additional improvements will need to be explored. Possible options include exploring the randomized extension to the BHC algorithm (54) or other machine

Downloaded from https://academic.oup.com/nar/article/47/11/5539/5488018 by guest on 21 August 2022

learning based approaches such as those based on matrix decomposition (55).

Fastbaps therefore offers two new ways to rapidly find high-quality clusters from a genetic alignment or phylogeny. The significant speed increase our software provides over previous approaches enables fitting a clustering model to previously intractable large alignments, and has the potential to allow continuous model refitting in surveillance settings. Fastbaps is provided as an open source package with a clearly documented interface.

SOFTWARE AVAILABILITY

Source code available from: https://github.com/gtonkinhill/fastbaps

Code for reproducing figures from: https://github.com/gtonkinhill/fastbaps_manuscript

Archived source code at time of publication to bioRxiv: https://doi.org/10.5281/zenodo.1472299

Archived code for figures at time of publication to bioRxiv: https://zenodo.org/badge/latestdoi/142294532

SUPPLEMENTARY DATA

Supplementary Data are available at NAR Online.

ACKNOWLEDGEMENTS

We thank Mathew Beale for helping to test the software and the three anonymous referees for helpful comments and suggestions.

FUNDING

Wellcome Trust [206194 to S.D.B]; Wellcome Trust PhD Scholarship Grant [204016 to G.T.H]; ERC [742158 to J.C.]; The Alan Turing Institute via an Engineering and Physical Sciences Research Council Grant [EP/510129/1 to S.D.W.F]; U.S. National Institutes of Health [R01AI135970 to S.D.W.F.]. Funding for open access charge: Wellcome Trust.
Conflict of interest statement. None declared.

REFERENCES

1. Kosakovsky Pond,S.L. and Frost,S. D.W. (2005) Not so different after all: a comparison of methods for detecting amino acid sites under selection. *Mol. Biol. Evol.*, **22**, 1208–1222.
2. Lees,J.A., Vehkala,M., Välimäki,N., Harris,S.R., Chewapreecha,C., Croucher,N.J., Marttinen,P., Davies,M.R., Steer,A.C., Tong,S. *et al.* (2016) Sequence element enrichment analysis to determine the genetic basis of bacterial phenotypes. *Nat. Commun.*, **7**, 12797–12805.
3. Earle,S.G., Wu,C.-H., Charlesworth,J., Stoesser,N., Gordon,N.C., Walker,T.M., Spencer,C.C.A., Iqbal,Z., Clifton,D.A., Hopkins,K.L. *et al.* (2016) Identifying lineage effects when controlling for population structure improves power in bacterial association studies. *Nat. Microbiol.*, 16041–16049.
4. Pritchard,J.K., Stephens,M. and Donnelly,P. (2000) Inference of population structure using multilocus genotype data. *Genetics*, **155**, 945–959.
5. Jombart,T., Devillard,S. and Balloux,F. (2010) Discriminant analysis of principal components: a new method for the analysis of genetically structured populations. *BMC Genet.*, **11**, 94–109.
6. Beugin,M.-P., Gayet,T., Pontier,D., Devillard,S. and Jombart,T. (2018) A fast likelihood solution to the genetic clustering problem. *Methods Ecol. Evol.*, **9**, 1006–1016.
7. Falush,D., Stephens,M. and Pritchard,J.K. (2003) Inference of population structure using multilocus genotype data: linked loci and correlated allele frequencies. *Genetics*, **164**, 1567–1587.
8. Corander,J. and Marttinen,P. (2006) Bayesian identification of admixture events using multilocus molecular markers. *Mol. Ecol.*, **15**, 2833–2843.
9. Corander,J., Marttinen,P., Sirén,J. and Tang,J. (2008) Enhanced Bayesian modelling in BAPS software for learning genetic structures of populations. *BMC Bioinformatics*, **9**, 539–553.
10. Corander,J., Waldmann,P. and Sillanpää,M.J. (2003) Bayesian analysis of genetic differentiation between populations. *Genetics*, **163**, 367–374.
11. Alexander,D.H., Novembre,J. and Lange,K. (2009) Fast model-based estimation of ancestry in unrelated individuals. *Genome Res.*, **19**, 1655–1664.
12. Raj,A., Stephens,M. and Pritchard,J.K. (2014) fastSTRUCTURE: variational inference of population structure in large SNP data sets. *Genetics*, **197**, 573–589.
13. Frichot,E., Mathieu,F., Trouillon,T., Bouchard,G. and François,O. (2014) Fast and efficient estimation of individual ancestry coefficients. *Genetics*, **196**, 973–983.
14. Anderson,E.C. and Thompson,E.A. (2002) A model-based method for identifying species hybrids using multilocus genetic data. *Genetics*, **160**, 1217–1229.
15. Cheng,L., Connor,T.R., Sirén,J., Aanensen,D.M. and Corander,J. (2013) Hierarchical and spatially explicit clustering of DNA sequences with BAPS software. *Mol. Biol. Evol.*, **30**, 1224–1228.
16. Evanno,G., Regnaut,S. and Goudet,J. (2005) Detecting the number of clusters of individuals using the software STRUCTURE: a simulation study. *Mol. Ecol.*, **14**, 2611–2620.
17. Akaike,H. (1974) A new look at the statistical model identification. *IEEE Trans. Automat. Contr.*, **19**, 716–723.
18. Pella,J. and Masuda,M. (2011) The Gibbs and split-merge sampler for population mixture analysis from genetic data with incomplete baselines. *Can. J. Fish. Aquat. Sci.*, **63**, 576–596.
19. Huelsenbeck,J.P., Andolfatto,P. and Huelsenbeck,E.T. (2011) Structurama: bayesian inference of population structure. *Evol. Bioinform. Online*, **7**, 55–59.
20. Huelsenbeck,J.P. and Andolfatto,P. (2007) Inference of population structure under a Dirichlet process model. *Genetics*, **175**, 1787–1802.
21. Blei,D.M. and Jordan,M.I. (2006) Variational inference for Dirichlet process mixtures. *Bayesian Anal.*, **1**, 121–143.
22. Iii,H.D. (2009) Fast search for Dirichlet process mixture models. In: *Proceedings of Machine Learning Research*. San Juan, pp. 83–90.
23. Wang,L. and Dunson,D.B. (2011) Fast Bayesian inference in Dirichlet process mixture models. *J. Comput. Graph. Stat.*, **20**, 296–216.
24. Crook,O.M., Gatto,L. and Kirk,P.D.W. (2018) Fast approximate inference for variable selection in Dirichlet process mixtures, with an application to pan-cancer proteomics. arXiv doi: https://arxiv.org/abs/1810.05450, 12 October 2018, preprint: not peer reviewed.
25. Minka,T. and Ghahramani,Z. (2003) Expectation propagation for infinite mixtures. *NIPS Workshop on Nonparametric Bayesian Methods and Infinite Models*, **19**, https://tminka.github.io/papers/infinite/.
26. Heller,K.A. and Ghahramani,Z. (2005) Bayesian Hierarchical Clustering. In: *Proceedings of the 22Nd International Conference on Machine Learning*. ACM, NY, pp. 297–304.
27. Florita,A.R., Brackney,L.J., Otanicar,T.P. and Robertson,J. (2013) Classification of commercial building electrical demand profiles for energy storage applications. *J. Sol. Energy Eng.*, **135**, 031020.
28. Savage,R.S., Heller,K., Xu,Y., Ghahramani,Z., Truman,W.M., Grant,M., Denby,K.J. and Wild,D.L. (2009) R/BHC: fast Bayesian hierarchical clustering for microarray data. *BMC Bioinformatics*, **10**, 242–251.
29. Murtagh,F. and Legendre,P. (2014) Ward's Hierarchical agglomerative clustering Method: Which algorithms implement Ward's criterion? *J. Classification*, **31**, 274–295.
30. Gagolewski,M., Bartoszuk,M. and Cena,A. (2016) Genie: a new, fast, and outlier-resistant hierarchical clustering algorithm. *Inf. Sci.*, **363**, 8–23.
31. Müllner,D. *et al.* (2013) fastcluster: Fast hierarchical, agglomerative clustering routines for R and Python. *J. Stat. Softw.*, **53**, 1–18.

Downloaded from https://academic.oup.com/nar/article/47/11/5539/5488018 by guest on 21 August 2022

32. Ragonnet-Cronin,M., Hodcroft,E., Hué,S., Fearnhill,E., Delpech,V., Brown,A. J.L., Lycett,S. and UK HIV Drug Resistance Database (2013) Automated analysis of phylogenetic clusters. *BMC Bioinformatics*, **14**, 317–326.
33. Strehl,A. and Ghosh,J. (2002) Cluster Ensembles—a knowledge reuse framework for combining multiple partitions. *J. Mach. Learn. Res.*, **3**, 583–617.
34. Richardson,S. and Green,P.J. (1997) On Bayesian analysis of mixtures with an unknown number of components (with discussion). *J. R. Stat. Soc.*, **59**, 731–792.
35. Staab,P.R., Zhu,S., Metzler,D. and Lunter,G. (2015) scrm: efficiently simulating long sequences using the approximated coalescent with recombination. *Bioinformatics*, **31**, 1680–1682.
36. Staab,P.R. and Metzler,D. (2016) Coala: an R framework for coalescent simulation. *Bioinformatics*, **32**, 1903–1904.
37. Tonkin-Hill,G., Lees,J.A., Bentley,S.D., Frost,S.D.W. and Corander,J. (2018) RhierBAPS: an R implementation of the population clustering algorithm hierBAPS. *Wellcome Open Res.*, **3**, 93–98.
38. Fowlkes,E.B. and Mallows,C.L. (1983) A method for comparing two hierarchical clusterings. *J. Am. Stat. Assoc.*, **78**, 553–569.
39. Chewapreecha,C., Harris,S.R., Croucher,N.J., Turner,C., Marttinen,P., Cheng,L., Pessia,A., Aanensen,D.M., Mather,A.E., Page,A.J. *et al.* (2014) Dense genomic sampling identifies highways of pneumococcal recombination. *Nat. Genet.*, **46**, 305–309.
40. Aanensen,D.M., Feil,E.J., Holden,M.T.G., Dordel,J., Yeats,C.A., Fedosejev,A., Goater,R., Castillo-Ramírez,S., Corander,J., Colijn,C., European SRL Working Group *et al.* (2016) Whole-Genome sequencing for routine pathogen surveillance in public Health: a population snapshot of invasive staphylococcus aureus in Europe. *Mbio*, **7**, e00444-16.
41. Lees,J.A., Kremer,P.H.C., Manso,A.S., Croucher,N.J., Ferwerda,B., Serón,M.V., Oggioni,M.R., Parkhill,J., Brouwer,M.C., van der Ende,A. *et al.* (2017) Large scale genomic analysis shows no evidence for pathogen adaptation between the blood and cerebrospinal fluid niches during bacterial meningitis. *Microb. Genome*, **3**, e000103.
42. Kremer,P.H.C., Lees,J.A., Koopmans,M.M., Ferwerda,B., Arends,A. W.M., Feller,M.M., Schipper,K., Valls Seron,M., van der Ende,A., Brouwer,M.C. *et al.* (2017) Benzalkonium tolerance genes and outcome in Listeria monocytogenes meningitis. *Clin. Microbiol. Infect.*, **23**, 265.e1–265.e7.
43. Koelman,D., Kremer,P., Lees,J., Brouwer,M., Bentley,S. and van de Beek,D. (2017) Bacterial hypervirulence in haemophilus influenzae meningitis identified by whole genome sequencing. *J. Neurol. Sci.*, **381**, 181–182.
44. HIV Sequence Compendium, Foley,B., Leitner,T., Apetrei,C., Hahn,B., Mizrachi,I., Mullins,J., Rambaut,A., Wolinksy,S. and Korber,B. (2018) In: *Theoretical Biology and Biophysics Group*. Los Alamos National Laboratory, NM.
45. Dudas,G., Carvalho,L.M., Bedford,T., Tatem,A.J., Baele,G., Faria,N.R., Park,D.J., Ladner,J.T., Arias,A., Asogun,D. *et al.* (2017) Virus genomes reveal factors that spread and sustained the Ebola epidemic. *Nature*, **544**, 309–315.
46. Price,M.N., Dehal,P.S. and Arkin,A.P. (2010) FastTree 2—approximately maximum-likelihood trees for large alignments. *PLoS One*, **5**, e9490.
47. McInnes,L. and Healy,J. (2018) UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction. arXiv doi: https://arxiv.org/abs/1802.03426, 09 February 2018, preprint: not peer reviewed.
48. Diaz-Papkovich,A., Anderson-Trocme,L. and Gravel,S. (2018) Revealing multi-scale population structure in large cohorts. bioRxiv doi: https://doi.org/10.1101/423632, 23 September 2018, preprint: not peer reviewed.
49. Becht,E., McInnes,L., Healy,J., Dutertre,C.-A., Kwok,I. W.H., Ng,L.G., Ginhoux,F. and Newell,E.W. (2018) Dimensionality reduction for visualizing single-cell data using UMAP. *Nat. Biotechnol.*, **37**, 38–48.
50. Kassambara,A. (2017) *Practical Guide to Cluster Analysis in R: Unsupervised Machine Learning*, STHDA, Vol. **1**, pp. 129–130.
51. Latch,E.K., Dharmarajan,G., Glaubitz,J.C. and Rhodes,O.E. (2006) Relative performance of Bayesian clustering software for inferringpopulation substructure and individual assignment at low levels of population differentiation. *Conserv. Genet.*, **7**, 295–302.
52. Zhang,J., Halkilahti,J., Hänninen,M.-L. and Rossi,M. (2015) Refinement of whole-genome multilocus sequence typing analysis by addressing gene paralogy. *J. Clin. Microbiol.*, **53**, 1765–1767.
53. Lees,J.A., Harris,S.R., Tonkin-Hill,G., Gladstone,R.A., Lo,S., Weiser,J.N., Corander,J., Bentley,S.D. and Croucher,N.J. (2018) Fast and flexible bacterial genomic epidemiology with PopPUNK. *Genome Res.*, **29**, 304–316.
54. Katherine,A. and Heller,Z.G. (2005) Randomized algorithms for fast bayesian hierarchical clustering. *PASCAL Workshop on Statistics and Optimization of Clustering*, **25**, 1–22.
55. Yang,Z., Corander,J. and Oja,E. (2016) Low-rank doubly stochastic matrix decomposition for cluster analysis. *J. Mach. Learn. Res.*, **17**, 1–25.

Downloaded from https://academic.oup.com/nar/article/47/11/5539/5488018 by guest on 21 August 2022