Random Projection Based Clustering for Population Genomics

Sotiris Tasoulis*, Lu Cheng‡ , Niko Välimäki§, Nicholas J Croucher¶ , Simon R Harris**
William P Hanage‖ Teemu Roos* and Jukka Corander†,
* Helsinki Institute for Information Technology HIIT, Department of Computer Science
University of Helsinki, Finland. Email: tasoulis.cs.helsinki.fi, teemu.roos@cs.helsinki.fi
‡ Helsinki Institute for Information Technology HIIT, Department of Information and Computer Science
Aalto University, Finland. Email: lu.cheng@helsinki.fi
§ Department of Computer Science, University of Helsinki, Finland. Email: niko.valimaki@helsinki.fi
¶ Department of Infectious Disease Epidemiology, Imperial College, London UK. Email: n.croucher@imperial.ac.uk
** Wellcome Trust Sanger Institute, Hinxton, Cambridge, UK. Email: sh16@sanger.ac.uk
‖ Center for Communicable Disease Dynamics, Department of Epidemiology,
Harvard School of Public Health, Boston, Massachusetts, USA. Email: bill.hanage@gmail.com
† Helsinki Institute for Information Technology HIIT, Department of Mathematics and Statistics
University of Helsinki, Finland. Email: jukka.corander@helsinki.fi

***Abstract*—Recent data revolution in population genomics for bacteria has increased the size of aligned sequence data sets by two-to-three orders of magnitude. This trend is expected to continue in the near future, putting an emphasis on applicability of big data techniques to leverage biologically important insights. Moreover, with the increasing density of sampling, it may also be necessary to consider alignment-free sequence analysis techniques combined with clustering to yield a sufficient insight to data. This leads to ultra high-dimensional data with tens of millions of variables, which can no longer be handled by the existing population genomic methods. Using the largest bacterial sequence data sets published to date, we demonstrate that random projection based clustering provides a highly accurate and several orders of magnitude faster approach to the analysis of both alignment-based and alignment-free genome data sets, compared with the Bayesian model-based analysis that is currently considered as the state-of-the-art. Hence, clustering methods for big data harbor considerable potential for important applications in genomics and could pave way for novel analysis pipelines even in the online setting when executed in a massively parallel computing environment.**

***Keywords*—Clustering; Random Projection; Population Genomics; High Dimensionality;**

I. Introduction

Recent rapid advances in sequencing technology have enabled a data revolution in population genomics, in particular for prokaryotic organisms such as bacteria. Since most bacteria are fast-evolving and can have highly cryptic population structure due to horizontal transfer of DNA, clustering is an important tool for the analysis of whole-genome bacterial sequence data, offering the potential to reveal evolutionary patterns that are not discoverable with standard phylogenetic methods. Very recently, sizes of the cutting-edge bacterial genome data sets have increased one to two orders of magnitude [1]–[3], which poses a considerable challenge even for the most advanced model-based clustering methods developed for population genomics [4], [5]. Currently, the largest published bacterial data set [3] harbors hundreds of thousands of variable genome positions and fitting of the Bayesian population genomic model to the data with intelligent non-reversible stochastic optimization algorithm [4], [5] is a matter of several weeks in a parallel computing environment. As data sets of comparable size and beyond will soon become widespread routine for the study of bacterial evolution, there is an urgent need for developing faster clustering methods that are able to maintain high level of accuracy compared with the model-based gold standard approach. Moreover, with the increasing density of sampling, it will also be necessary to use alignment-free sequence analysis techniques combined with clustering to yield sufficient insight to data. Already with the afore-mentioned genome studies, this leads to ultra high-dimensional data with tens of millions of variables, which can no longer be handled by the existing Bayesian population genomic methods. Here we demonstrate that random projection based clustering provides a highly accurate and several orders of magnitude faster approach to the analysis of both alignment-based and alignment-free genome data sets. Hence, clustering methods for big data harbor considerable potential for important applications in genomics and could pave way for novel analysis pipelines even in the online setting when executed in a massively parallel computing environment.

The rest of the paper is organized as follows. In Section II we give a brief description of the Random Projection (RP) method. In Sections III and IV the RP based clustering algorithms are presented while Section V is devoted to the experimental evaluation of the clustering approaches.

II. Background

Clustering is a very important problem in data mining and statistical pattern recognition, which has been studied extensively by several research communities the past decades. More recently, high-dimensional clustering has been a very active research field, due to its relevance in real-world applications.

978-1-4799-5666-1/14/$31.00 ©2014 IEEE

Along with the high computational requirements a fundamental limitation posed by increasing dimensionality is that concepts like proximity, distance, or neighbourhood, which are intrinsically related to similarity, become less meaningful.

Projection methods for dimension reduction have enabled the discovery of otherwise unattainable structure in ultra high dimensional data [6]–[9]. More recently, a particular method, namely Random Projection [10]–[12], has been shown to have the advantage of high quality data representations with minimal computation effort, even for data dimensions in the range of hundreds of thousands or even millions. In [13] this dimension reduction technique was coupled with a hierarchical data clustering algorithm that is specially designed for high dimensional cases while it was shown that the theoretical properties of both components can be combined in a promising and effective clustering framework that achieves high quality data partitions, orders of magnitude faster. In addition, in [14] it was shown that by applying the random projection method prior to the k-means clustering algorithm the real "clusters" are not distorted significantly.

A. The Random Projection Method

Random Projection (RP) for dimensionality reduction has been used widely in several domains. Using RP in the context of a system for organizing textual documents, Kaski [15] presented experimental results that were as good as those obtained using PCA. Papadimitriou et al. [12] use random projection as a prepossessing step in an attempt to speed up Linear Discriminant Analysis for document categorization. Bingham and Mannila [11] have shown that RP preserves distances and has performance comparable to that of PCA on image and text data, while being much faster. Dasgupta [16], [17] also concludes that RP results in more spherical cluster than those in the original dimension. RP also performs better than PCA on eccentric data, where actually PCA might fail completely. Based on the principles of RP in [18] Schneider and Vlachos combine random projections onto randomly chosen lines with hierarchical clustering to offer expedient construction of dendrograms with provable quality guarantees, while in [19] through the use of such random projections they extend density-based clustering techniques by presenting algorithms that significantly improve runtime.

RP is motivated by the Johnson–Lindenstrauss lemma which states that a set of n points in a high dimension Euclidean space can be mapped down onto an $r < O(\log n/\varepsilon^2)$ dimension space such that the distances between the points are approximately preserved. In other words, the distances are not distorted more than a factor of $1 \pm \varepsilon$, for any $0 < \varepsilon < 1$.

B. Technical details of RP

In RP the original a-dimensional data is projected to an r-dimensional subspace ($r < a$), using a random $a \times r$ orthogonal matrix R, whose rows have unit lengths. Using matrix notation where $D_{n\times a}$ is the original set of n a-dimensional observations,

$$D^{RP}_{n\times r} = D_{n\times a} R_{a\times r},$$

is the projection of the data onto the lower r-dimensional subspace spanned by $\mathbb{R}$.

The orthogonalization of R is computationally expensive, but necessary in order to preserve similarities between the original vectors in the low dimension space and avoid distortions. However, in some cases, we can avoid orthogonalization. As shown in [20], in high dimension spaces, there exists a much larger number of almost orthogonal vectors than orthogonal directions. Thus, high-dimensional vectors having random directions are very likely to be close to orthogonal.

The selection of the elements of the matrix R is a matter of interest. In most cases the elements are drawn from a Normal distribution, but this is not always necessary. In [10] Achlioptas has proposed a much simpler algorithm for approximating the random matrix. Random projection is computationally very simple, forming the random matrix R and projection the $n \times a$ data matrix D onto r dimensions is of order $O(arn)$, and if the data matrix D is sparse with about c non zero entries per column, the complexity is reduced to $O(crn)$ [12].

III. Random direction divisive clustering

Divisive hierarchical clustering algorithms build a hierarchy of clusters following a top to bottom procedure. Starting with a single all inclusive cluster at the top, clusters are split recursively into two parts until a stopping criterion is satisfied.

The main characteristic of the dePDDP (density enhanced Principal Direction Divisive Partitioning) algorithm is that it incorporates information from the density of the projected data onto the first principal component. The algorithm creates a top-down hierarchy of partitions by iteratively projecting on the first principal component and performing binary splits based on the density of the projected data.

Let us assume that the data at hand is represented by an $n \times a$ matrix D, in which each row represents a data sample $d_i, i = 1, \ldots, n$, and a denotes the dimensionality. We define the vector b and matrix Σ to represent the mean vector and the covariance of the data respectively:

$$b = \frac{1}{n}\sum_{i=1}^{n} d_i, \quad \Sigma = \frac{1}{n}(D - be)^\top (D - be),$$

where e is a column vector of ones. The covariance matrix Σ is symmetric and positive semi-definite, so all its eigenvalues are real and non-negative. The eigenvectors u_j, $j = 1, \ldots, k$, corresponding to the k largest eigenvalues, are called the principal components or principal directions. The dePDDP algorithm uses the projections p_i:

$$p_i = u_1(d_i - b), \quad i = 1, \ldots, n,$$

onto the first principal component u_1, to initially separate the entire data set into two partitions P_1 and P_2 based on the minimimum of all local minima of their univariate density estimation, namely *MinLocal* . More formally *MinLocal* defined in Definition 1 as follows:

Definition 1. *(**MinLocal**): A point $\mu \in \mathbb{R}$ is a local minimum of $\hat{f}_{1D}$ is there exists $\delta > 0$ s.t. $\hat{f}_{1D}(\mu) < \hat{f}_{1D}(x)$ for all $x \in (\mu - \delta, \mu + \delta)$. A point $\mu^* \in \mathbb{R}$ is said to be a MinLocal if $\hat{f}_{1D}(\mu^*) \leqslant \hat{f}_{1D}(\mu)$ for all local minima μ.*

In the above definition $\hat{f}_{1D} : \mathbb{R} \to [0, \infty)$ is the kernel density estimation of the density of the projected data onto

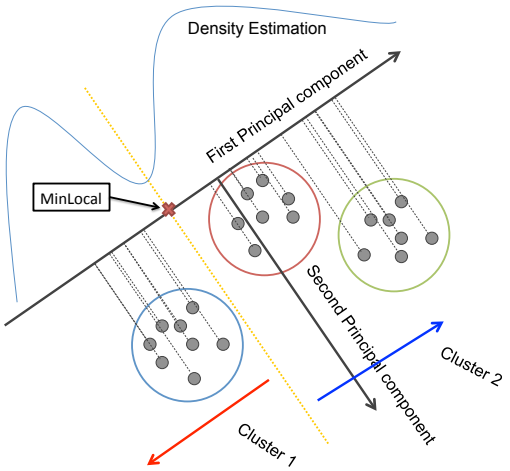

Fig. 1: An example of the dePDDP algorithm's splitting criterion.

the first principal component defined as

$$\hat{f}_{1D}(p_y) = \frac{1}{n} \sum_{x \in \mathcal{D}} K(|p_y - p_x|)$$

at any point $p_y \in \mathbb{R}$, where $h \in \mathbb{R}^+$ is a real positive number called the bandwidth and K is a *kernel* function which needs to be normalized. Several different kernel functions can be used, such as triangular, normal, etc., but typically, the quality of a kernel estimate depends less on the shape of K than on the value of its bandwidth h. Small values of h increase the variance and thus the resulting estimate $\hat{f}_{1D}$ seems "wiggly" with many spurious features if graphically checked. On the other hand, big values of h reduce the variance of $\hat{f}_{1D}$, but also increase the bias, probably "smoothing away" the features of the true density. In this work the bandwidth parameter is set by choosing a multiple of the h_{opt} bandwidth ("normal reference rule") as proposed in [21]. That is the bandwidth that minimizes the mean integrated squared error (MISE), given by:

$$h_{opt} = \sigma \left(\frac{4}{3n} \right)^{1/5},$$

where σ is the standard deviation of the data. That multiple was set to 3 for all experiments.

The computational complexity of the dePDDP implementation is mostly influenced by the computation of the principal vectors. To compute them, the Singular Value Decomposition of the data matrix is employed. This introduces a total worst case complexity of $\mathrm{O}\left(L(2 + k_{SVD})(s_{nz} n\ a)\right)$ where k_{SVD} are the iterations needed by the Lanczos SVD computation algorithm, s_{nz} is the fraction of non-zero entries in D and L is the number of clusters retrieved (for more details refer to [8]). In addition, using techniques like the Fast Gauss Transform we achieve linear running time for the Kernel Density Estimation while to find MinLocal we only need to evaluate the density at n positions, in between the projected data points, since those are the only places we can have valid splitting points.

The Random Projection dePDDP *(rp-dePDDP)* algorithm is a method that combines RP along with the dePDDP algorithmic structure by exploiting the relationship between the density of the true clusters in the data and the density of the one dimensional projections of the data that are priory projected onto a random frame. By expressing the Johnson and Lindenstrauss lemma in terms of the kernel density estimate of the data projection, it has been shown that using the dePDDP algorithm's splitting technique we can guarantee not to split any actual cluster up to an error term. For more information on the proof of the related theorems the interested reader should refer to [13]. A short description of the *(rp-dePDDP)* algorithm follows:

> The algorithm starts with a single cluster $C_0 = \mathcal{D}^{\mathcal{RP}}$ constituted by the projections of the input data $\mathcal{D}$ onto the Random Frame. Then iterative splits of clusters C_i are performed corresponding to the *MinLocal*. The selected cluster is replaced by two subclusters that are constituted by the points whose one dimensional projections lie at the left and right side of μ^* respectively. The procedure stops when there are no local minima of $\hat{f}_{1D}$ in any of the remaining clusters. This termination criterion allows the algorithm to provide estimates of the number of clusters that constitute the given dataset.

An algorithmic outline is presented at Algorithm 1. Note in particular that the first principal component is recalculated recursively on each of the subclusters. In addition, a graphical illustration of the method is given in Figure 2.

Algorithm 1 The rp-dePDDP algorithm summary.

1: Given the error parameter ε calculate $\mathcal{D}^{\mathcal{RP}}$
2: Set $\Pi = \{\mathcal{D}^{\mathcal{RP}}\}$
3: **repeat**
4: Select a set $\mathcal{C} \in \Pi$
5: Split $\mathcal{C}$ into two sub-sets $\mathcal{C}_1$ and $\mathcal{C}_2$
6: Remove $\mathcal{C}$ from Π and set $\Pi \rightarrow \Pi \cup \{\mathcal{C}_1, \mathcal{C}_2\}$
7: **until** Stopping Criterion is not satisfied
8: Return Π the partition of $\mathcal{D}$ into $|\Pi|$ clusters

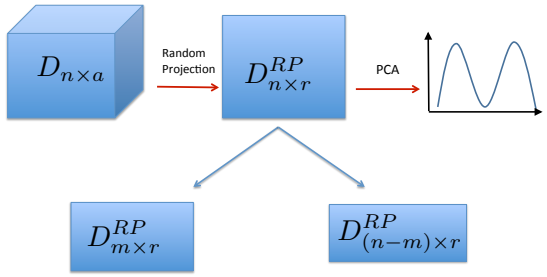

Fig. 2: An example of the rp-dePDDP algorithm's methodology.

Algorithm 2 The rp-$k$means algorithm summary.

1: Given the error parameter ε calculate $\mathcal{D}^{\mathcal{RP}}$
2: Set the desired number of clusters k
3: Run the k-means algorithm on $\mathcal{D}^{\mathcal{RP}}$
4: Return Π the partition of $\mathcal{D}$ into k clusters

IV. Random Projections for k-means

The k-means algorithm is simple and quite efficient in most cases; it was recently recognized as one of the top ten data mining tools of the last fifty years. As a partitioning clustering algorithm, starts from an initial clustering (that may be formed at random) and subsequently create flat partitionings by iteratively adjusting the clusters based on the distance of the data points from a representative member of each cluster. The authors in [14] discuss the topic of dimensionality reduction for k-means focusing on the application of the random projection method to the k-means clustering problem. In the proposed approach the original dataset $D_{n\times a}$ is projected onto an r dimensional subspace with $r \ll a$, and then k-means is applied on the projected data points. The size of the target subspace is defined by $r = ck/\varepsilon^2$ where c is a sufficient large constant. It is shown that the algorithm distorts the real "clusters" by a factor at most $2+\varepsilon$ for some $\varepsilon \in (0, 1/3)$. Although there are many variants of k-means that improve its performance and are less susceptible to initialization problems, here we focus only on the simple form of k-means. The algorithmic scheme of the aforementioned method on which we will refer as rp-$k$means is presented at Algorithm 2.

If the random frame R is constructed to be a sign matrix as defined in [10], then the mailman algorithm for matrix manipulation [22] can be used to compute the RP in $\mathrm{O}\left(na\lceil\varepsilon^2 k/\log(a)\rceil\right)$. Subsequently the total running time can be bounded by $\mathrm{O}\left(na\lceil\varepsilon^2 k/\log(a)\rceil + 2^{(k/\varepsilon)^{\mathrm{O}(1)}} kn/\varepsilon^2\right)$ using the γ-approximation algorithm presented in [23] for k-means with input parameter $\gamma = 1+\varepsilon$.

V. Experimental Results

To test the presented approaches we use the three largest bacterial genome data sets published to date: [1] (MA), [2] (TB), [3] (MAELA). The genome alignments in these studies have been analysed using the Bayesian model-based method (BAPS) [4], [5] and we use the clustering output as the gold standard against which the random projection -based clustering is compared. The BAPS clusterings used here as the reference are nested at either two or three levels, in the results below we generally refer to the cluster labels at the second level. In any of the case the level of clustering is explicitly mentioned. The data sets contain between 616-3085 bacterial genomes, with up to 385,000 variable positions (MAELA). The variable positions, single-nucleotide polymorphisms (SNPs), are stored in a data matrix with each sequence as a row. The largest alignment-based data matrix (MAELA) is of size 3085 $\times$ 385,000.

In addition, since producing a genome alignment is computationally very expensive in itself, we use in tandem an alignment-free approach, where DNA words of a given length (KMERs) are scanned from the genome assemblies using the distributed method introduced in [24]. By comparing the accuracy of the clustering based on KMER indicator variable data with the alignment-based clustering, it is possible to assess the potential of using methods for ultra high-dimensional data to bypass the alignment procedure in an analysis. The use of KMERs is computationally attractive since scanning of the DNA words scales linearly with respect to the size of the assemblies, whereas alignment requires methods that are at least polynomial in terms of computational complexity. However, it is an open research question how much information about population structure will be lost by the use of KMERs instead of an alignment. In our experiments we used the fixed KMER length of 21 to ensure sufficient within-species informativeness of the resulting indicator variables. Earlier, it has been shown that KMERs approximately of length 12-13 are generally sufficient for separating genomes of bacteria representing different species [25], but considerably longer words are still necessary for characterization of differences in within-population variation. Notably, even much longer KMERs (up to length 50) are used in the single genome assembly process where short DNA reads are assembled to sequence contigs [26].

To measure the degree of correspondence between the

TABLE I: Mean purity, V-measure and number of found clusters (with the observed standard deviation in parentheses) for the genome alignment datasets.

	Purity	V-measure	# of Clusters
	dePDDP		
TB (12)	0.8412	0.7051	25
MA (16)	0.9513	0.7924	43
	rp-dePDDP $\varepsilon = 0.1$		
TB (12)	0.8503 (0.06)	0.7138 (0.03)	28.09 (5.10)
MA (16)	0.8787 (0.11)	0.7631 (0.07)	34.36 (7.33)
MAELA (190)	0.7528 (0.03)	0.8688 (0.01)	151.60 (13.50)
	rp-$k$means $\varepsilon = 0.1$		
TB (12)	0.7666 (0.06)	0.7309 (0.05)	
MA (16)	0.6903 (0.05)	0.7297 (0.04)	
MAELA (190)	0.8317 (0.02)	0.9008 (0.01)	
	rp-dePDDP $\varepsilon = 0.05$		
TB (12)	0.8519 (0.05)	0.7141 (0.02)	27.17 (4.11)
MA (16)	0.9412 (0.02)	0.7932 (0.01)	40.01 (2.46)
MAELA (190)	0.7091 (0.11)	0.8349 (0.09)	139.90 (30.08)
	rp-$k$means $\varepsilon = 0.05$		
TB (12)	0.7504 (0.07)	0.7207 (0.06)	
MA (16)	0.6786 (0.05)	0.7200 (0.04)	
MAELA (190)	0.7651 (0.02)	0.8969 (0.01)	

TABLE II: Mean purity, V-measure and number of found clusters(with the observed standard deviation in parenthesis) with respect to the first level class labels for the genome alignment datasets.

	Purity	V-measure	# of Clusters
	rp-dePDDP $\varepsilon = 0.1$		
TB first level (3)	0.9897 (0.00)	0.4778 (0.04)	28.09 (5.10)
MAELA first level (34)	0.9400 (0.02)	0.7743 (0.01)	151.60 (13.50)
	rp-dePDDP $\varepsilon = 0.05$		
TB first level (3)	0.9917 (0.00)	0.4455 (0.01)	27.17 (4.11)
MAELA first level (34)	0.8888 (0.13)	0.7487 (0.06)	139.90 (30.08)

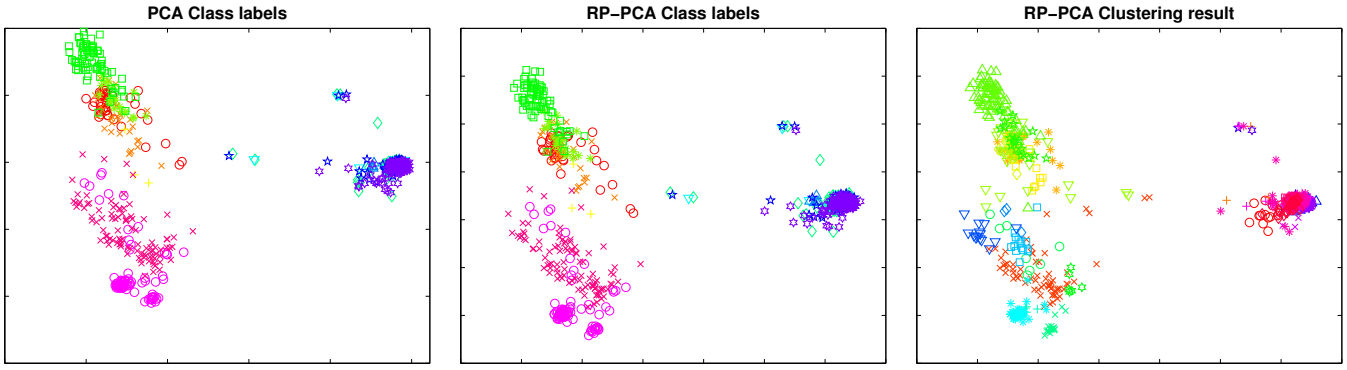

Fig. 3: The projected TB alignment data onto the first two principal components with respect to the class labels and the clustering result respectively.

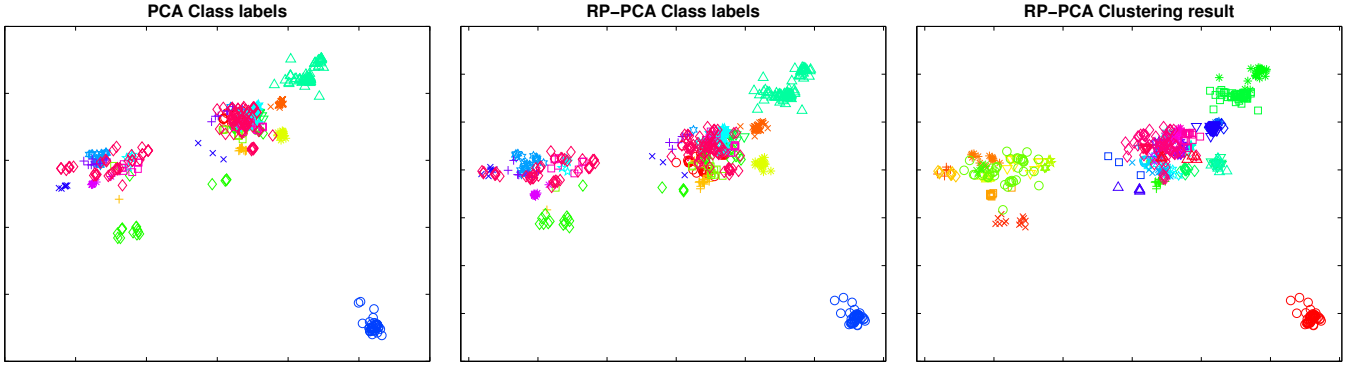

Fig. 4: The projected MA alignment data onto the first two principal components with respect to the class labels and the clustering result respectively.

resulting clusters and the assigned classes (coming from BABS clustering) for each object we use two measures namely Purity and V-measure [21]. Both of these scale from 0 to 1 with high values indicating better clusterings. Large Purity values indicate that the majority of vectors in each cluster come from the same class, so in essence the partitioning is "pure" with respect to class labels. V-measure tries to capture cluster homogeneity and completeness, by representing the degree to which classes are split to different clusters. For details on how these are calculated, see [27].

The size of the r-dimensional subspace used in the RP step is set based on the desired ε value (see Section II-A), since we have $r = O(\log n/\varepsilon^2)$ while the random frame R is a random sign matrix as defined in [10]. In what follows we present results for rp-dePDDP and rp-$k$means respectively where both algorithms use the same random projections in their processes for all cases. The predefined number of clusters is given as input to rp-$k$means while in the case of rp-dePDDP it is automatically determined based on the algorithm's termination criterion. Note that very often the initialization of rp-$k$means produces an empty cluster but these cases are excluded from the analysis.

A. *Genome Alignments*

Table I reports the clustering results of the algorithms for the genome alignment datasets with respect to the Purity, V-measure and number of found clusters when the ε error parameter is set to 0.1 and 0.05 respectively. For details regarding this parameter selection see the *Sensitivity Analysis* paragraph below. The mean values along with the standard deviation in the parenthesis are reported for 100 algorithm's runs of the RP based methods. For comparison purposes we also employ the basic dePDDP algorithm for TB and MA datasets where the direct application of PCA is feasible. For MAELA dataset this was not possible due to memory restrictions using the available resources (server computer with 32 gigabytes of memory). As shown there is not any significant difference in the results for the different ε values for both RP based algorithms while we notice that there is a slight decrease in the performance of rp-dePDDP for the MAELA dataset when $\varepsilon = 0.05$. With respect to V-measure the algorithms have similar performance for the TB and MA datasets, while rp-dePDDP and dePDDP achieve higher purity values in these cases due to the high number of retrieved clusters. rp-$k$means performs better for the MAELA dataset as rp-dePDDP retrieves less clusters than the number of classes. In Table II we also report the clustering results of the rp-dePDDP algorithm with respect to the first level class labels

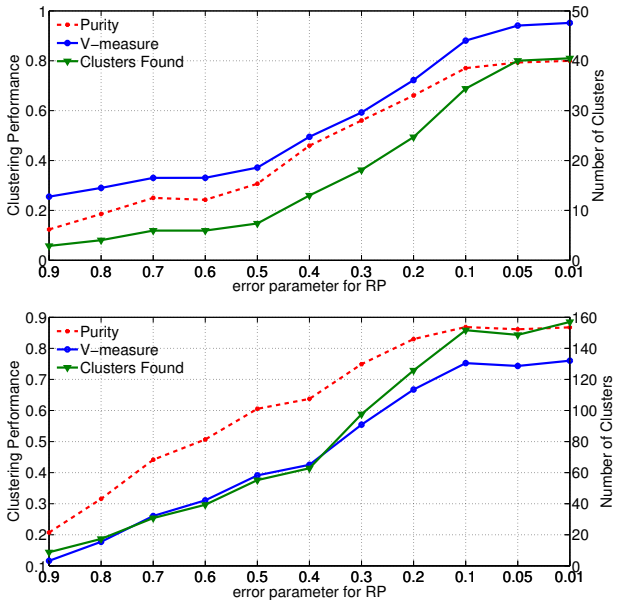

Fig. 5: Clustering result with respect to the Purity and V-measure and number of found clusters for different ε parameter values for the MA dataset (top) and the MAELA dataset (bottom).

when these are available. As shown by the high purity values the clustering result captures the first level clustering structure very well. The lower values of V-measure is a consequence of the low number of first level classes being partitioned to a much high number of clusters.

In an attempt to visually investigate the clustering result and the effect of RP in the datasets structure we employ 2-dimensional projections onto the first two principal components for the TB and MA dataset. In Figure 3 (left) PCA is applied directly to the TB dataset. The different symbols and colours of the points describe the class labels. In Figure 3 (middle) PCA is applied to the randomly projected data points and again the class labels are described by different symbols and colours, while in Figure 3 (right) the 2-dimensional projections of the randomly projected data points are plotted with respect to the clustering result. Similar plots for the MA dataset are illustrated in Figure 4. We notice that there is not any significant visual difference in the datasets structure after applying the RP. Also the clustering results match very well the class labels, for both cases. In addition, in Figure 3 (see bottom part of each subfigure) we visually identify points with the same class label that form clearly separate groups. These groups are in fact identified as separate clusters by the algorithm. Similar behaviour is observed in Figure 4 indicating that this phenomenon may occur in other cases as well. Such information could help the experts identify further structure in the data that is not captured by BABS.

Sensitivity Analysis: In what follows, we perform an extensive analysis of the ε parameter. For that purpose we use MA and MAELA datasets. In Figure 5 (top) we can see the clustering results for the MA dataset with respect to the mean values of Purity, V-measure and number of found clusters over 100 experiments for different ε values. As shown for

TABLE III: Mean purity, V-measure and number of found clusters(with the observed standard deviation in parenthesis) for the KMER datasets

	Purity	V-measure	# of Clusters
	rp-dePDDP $\varepsilon = 0.1$		
TB (12)	0.6495 (0.01)	0.6130 (0.02)	20.31 (3.56)
MA (16)	0.5617 (0.18)	0.4520 (0.16)	24.87 (15.21)
	rp-$k$means $\varepsilon = 0.1$		
TB (12)	0.6111 (0.03)	0.6268 (0.05)	
MA (16)	0.7798 (0.05)	0.7637 (0.04)	
	rp-dePDDP $\varepsilon = 0.05$		
TB (12)	0.6622 (0.02)	0.5802 (0.01)	24.46 (2.82)
MA (16)	0.8142 (0.15)	0.6551 (0.14)	44.82 (11.99)
	rp-$k$means $\varepsilon = 0.05$		
TB (12)	0.6169 (0.03)	0.6233 (0.04)	
MA (16)	0.7828 (0.06)	0.7720 (0.05)	

error values less than 0.2 the clustering results are quite good while the performance is stabilizing for values less than 0.05. The computational time for a single run when $\varepsilon = 0.1$ is approximately 10 minutes for the MA dataset, using Maltab on OS X with the following computer specifications: processor 1, 3 GHz Intel Core i5 and memory 8 GB 1600 MHz DDR3. In this case it was possible to load the full data matrix into memory as well as the constructed random matrix thus the computation of the random projection talking advantage of Matlab's efficient matrix computations is very fast. However due to the memory limitations it is impossible to load the dataset and the random matrix at once in the case of the MAELA dataset. The simplicity of the RP method allow us to apply it iteratively by generating only a column of the random matrix at each step and then discard it. This modification extended the computational time of a single run to approximately 1 hour. It is important to notice that in all cases the total computational time is mainly determined by the computation of RP. After reducing the dimensionality and thus surpassing the memory limitations, the execution time of the clustering algorithms is minimal, which is expected considering the relatively small number of samples. In general, the computational challenges in this work regard the computation RP. Figure 5 (bottom) displays the same kind of a parameter analysis with respect to the bigger MAELA dataset. The difference in size and dimensionality between the two datasets does not alter the results significantly. The clustering algorithm seems to achieve its maximum performance again for values close to 0.1. It is also important to notice that very small ε values results into high dimensional random projections. For example in the case of $\varepsilon = 0.01$ the resulting dimensionality for the MAELA dataset is 80,407. This justifies the use of a high dimensional clustering algorithm like dePDDP even after the RP method is applied.

B. KMER datasets

Even if we manage to bypass the memory restrictions using the aforementioned techniques, the computation of RP becomes extensively slow as the size of the random matrix grows. In the case where loading the data matrix into memory is not possible we will have to pass over it for each generated part of the random matrix. Thankfully, it is possible to avoid multiple passes over the data by saving the random generator

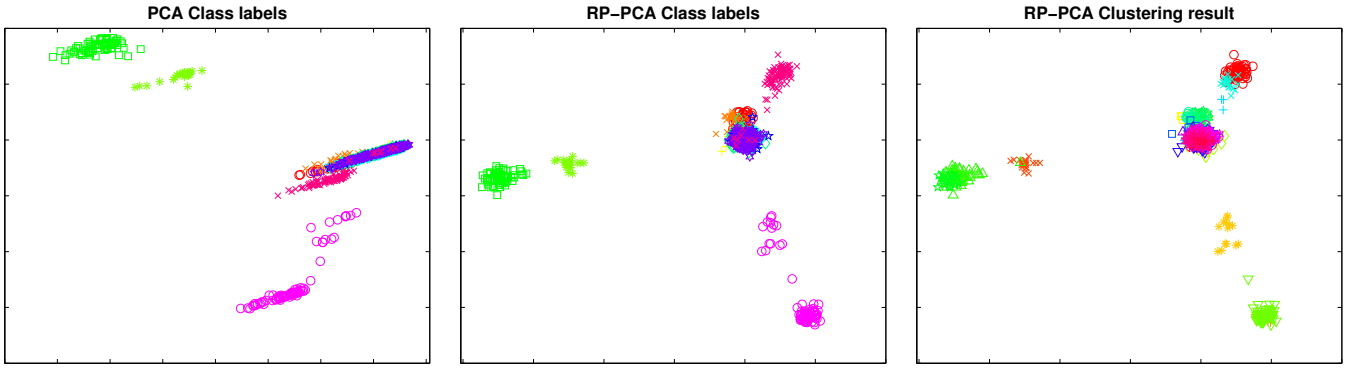

Fig. 6: The projected TB KMER data onto the first two principal components with respect to the class labels and the clustering result respectively.

TABLE IV: Mean purity and V-measure (with the observed standard deviation in parenthesis) with respect to the first level class labels for the KMER datasets.

	Purity	V-measure	# of Clusters
	rp-dePDDP $\varepsilon = 0.1$		
TB first level (3)	0.9867 (0.00)	0.4780 (0.02)	20.31 (3.56)
	rp-dePDDP $\varepsilon = 0.05$		
TB first level (3)	0.9845 (0.00)	0.5333 (0.04)	24.46 (2.82)

seed when generating parts of the random matrix. Still the computation of RP would require several hours strained by the regular usage of the hard disk. To efficiently apply RP to the much bigger KMER datasets (MA in this case is characterized by 9,469,696 attributes) we introduce an efficient and simple approach talking advantage of parallel computations. It is possible to generate the same random matrix in individual nodes of a cluster computer by storing the random seed that the random generation mechanism uses. Then to compute the random projection of the dataset we just need to distribute parts of the data matrix amongst the cluster nodes. This makes the problem perfectly parallel so calculating the random projection of the MA dataset on the Computer Science Department's high performance cluster constituted by 192 nodes with 32GB of RAM and 2 Intel Xeon E5540 2.53GHz CPUs takes only a few minutes.

Table III reports the clustering results of the algorithms for the KMER datasets. Again these are mean values and the corresponding standard deviation in the parenthesis for 100 algorithms' runs. In the case of the MA dataset the rp-dePDDP algorithm in its default settings only split a few data points, probably outliers, and then based on the termination criterion stops the procedure having found just a small number of clusters. This behaviour is due to the inappropriate bandwidth parameter for the kernel density estimation. By reducing the multiplier value of the bandwidth parameter by 0.5 the performance of the algorithm is increased significantly. However, this phenomenon still occurs in some cases, mostly when $\varepsilon = 0.1$, which explains the large variation in the clustering results. That is the reason why rp-$k$means performs better in this case. Table IV again reports the results of rp-dePDDP with respect to the first level class labels for the TB dataset.

In Figure 6 we can visually investigate the clustering result and the effect of RP in the structure of the TB dataset following the procedure explained in the previous Section. Here we observe the more spherical shapes of the clusters when PCA is applied to the randomly projected dataset (Figure 6 (middle-right)). It has been shown [17], that the randomly projected counterparts of even highly skewed clusters will be more spherical an thus more easy to detect by most clustering algorithms. Finally, the clustering result again matches very well the class labels.

VI. Conclusion

We have shown that Random Projection -based clustering offers a promising approach to population genomic analysis of bacteria, where Bayesian model-based analysis is currently considered as the gold standard. The rapidly increasing dimensionality of genome data sets is calling for novel approaches that would be more suitable for massive parallelization and online type analysis. For the largest published bacterial genome data set (MAELA), the model-based analysis takes approximately four weeks of CPU time with fast intelligent stochastic optimization. Hence, the RP approach offers a several order of magnitude faster clustering, while maintaining high levels of homogeneity and completeness. The clustering results were relatively insensitive with respect to the ε parameter, as long as the values remain bounded below a certain threshold. Moreover, we showed that reasonably high accuracy clusterings results could also be obtained with alignment-free input based on high-order DNA word (KMER) occurrences in the genome assemblies. This suggests that it would also be useful to examine the potential of KMERs to population genomic analysis in more depth.

Acknowledgment

This research was partially supported by the Academy of Finland under the Finnish Centre of Excellence in Computational Inference Research (COIN) and by ERC grant no. 239784.

References

[1] N. J. Croucher, J. A. Finkelstein, S. I. Pelton, P. K. Mitchell, G. M. Lee, J. Parkhill, S. D. Bentley, W. P. Hanage, and M. Lipsitch, "Population

genomics of post-vaccine changes in pneumococcal epidemiology," *Nat Genet*, vol. 45, no. 6, pp. 656–663, Jun. 2013.

[2] N. Casali, V. Nikolayevskyy, Y. Balabanova, S. R. Harris, O. Ignatyeva, I. Kontsevaya, J. Corander, J. Bryant, J. Parkhill, S. Nejentsev, R. D. Horstmann, T. Brown, and F. Drobniewski, "Evolution and transmission of drug-resistant tuberculosis in a Russian population," *Nat Genet*, vol. 46, no. 3, pp. 279–286, Mar. 2014.

[3] C. Chewapreecha, S. R. Harris, N. J. Croucher, C. Turner, P. Marttinen, L. Cheng, A. Pessia, D. M. Aanensen, A. E. Mather, A. J. Page, S. J. Salter, D. Harris, F. Nosten, D. Goldblatt, J. Corander, J. Parkhill, P. Turner, and S. D. Bentley, "Dense genomic sampling identifies highways of pneumococcal recombination," *Nature Genetics*, vol. advance online publication, Feb. 2014.

[4] J. Tang, W. P. Hanage, C. Fraser, and J. Corander, "Identifying currents in the gene pool for bacterial populations using an integrative approach." *PLoS Computational Biology*, vol. 5, no. 8, 2009.

[5] L. Cheng, T. R. Connor, J. Sirén, D. M. Aanensen, and J. Corander, "Hierarchical and Spatially Explicit Clustering of DNA Sequences with BAPS Software," *Molecular Biology and Evolution*, vol. 30, no. 5, pp. 1224–1228, Feb. 2013.

[6] C. Aggarwal, J. Wolf, P. Yu, C. Procopiuc, and J. Park, "Fast algorithms for projected clustering," in *SIGMOD '99: Proceedings of the 1999 ACM SIGMOD international conference on Management of data*, 1999, pp. 61–72.

[7] C. Aggarwal and P. Yu, "Finding generalized projected clusters in high dimensional spaces," in *Proc. ACM SIGMOD '00*. New York: ACM Press, 2000, pp. 70–81.

[8] D. Boley, "Principal direction divisive partitioning," *Data Mining and Knowledge Discovery*, vol. 2, no. 4, pp. 325–344, 1998.

[9] E. Müller, S. Günnemann, I. Assent, and T. Seidl, "Evaluating clustering in subspace projections of high dimensional data." *PVLDB*, vol. 2, no. 1, pp. 1270–1281, 2009.

[10] D. Achlioptas, "Database-friendly random projections," in *Proceedings of the Twentieth ACM Symposium on Principles of Database Systems, ACM Press*, 2001, pp. 274–281.

[11] E. Bingham and H. Mannila, "Random projection in dimensionality reduction: Applications to image and text data," in *Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM Press*, 2001, pp. 245–250.

[12] C. H. Papadimitriou, P. Raghavan, H. Tamaki, and S. Vempala, "Latent semantic indexing: A probabilistic analysis," *Proc. 17th ACM Symp. on the Principles of Database Systems*, pp. 159–168, 1998.

[13] S. K. Tasoulis, D. K. Tasoulis, and V. P. Plagianakos, "Random direction divisive clustering," *Pattern Recogn. Lett.*, vol. 34, no. 2, pp. 131–139, Jan. 2013.

[14] C. Boutsidis, A. Zouzias, and P. Drineas, "Random projections for k-means clustering," *CoRR*, vol. abs/1011.4632, 2010.

[15] S. Kaski, "Data exploration using self-organizing maps," *Acta Polytechnica Scandinavica, Mathematics, Computing and Management in Engineering Series*, no. 82, 1997.

[16] S. Dasgupta, "Learning mixtures of gaussians," *Foundations of Computer Science, Annual IEEE Symposium on*, vol. 0, p. 634, 1999.

[17] S. Dasgupta, "Experiments with random projection," in *Proceedings of the 16th Conference on Uncertainty in Artificial Intelligence*, ser. UAI '00. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2000, pp. 143–151.

[18] J. Schneider and M. Vlachos, "On randomly projected hierarchical clustering with guarantees," *CoRR*, vol. abs/1401.5814, 2014.

[19] J. Schneider and M. Vlachos, "Fast parameterless density-based clustering via random projections," in *Proceedings of the 22nd ACM international conference on Conference on information knowledge management*, ser. CIKM '13. New York, NY, USA: ACM, 2013, pp. 861–866.

[20] R. Hecht-Nielsen, "Context vectors: general purpose approximate meaning representations self-organized from raw data," *Computational Intelligence: Imitating Life, IEEE Press*, pp. 43–56, 1994.

[21] S. Tasoulis, D. Tasoulis, and V. Plagianakos, "Enhancing principal direction divisive clustering," *Pattern Recognition*, vol. 43, pp. 3391–3411, 2010.

[22] E. Liberty and S. W. Zucker, "The mailman algorithm: A note on matrix–vector multiplication," *Inf. Process. Lett.*, vol. 109, no. 3, pp. 179–182, Jan. 2009.

[23] A. Kumar, Y. Sabharwal, and S. Sen, "A simple linear time $(1+\epsilon)$ -approximation algorithm for k-means clustering in any dimensions," *2013 IEEE 54th Annual Symposium on Foundations of Computer Science*, vol. 0, pp. 454–462, 2004.

[24] N. Vlimki and S. J. Puglisi, "Distributed string mining for high-throughput sequencing data." in *WABI*, ser. Lecture Notes in Computer Science, B. J. Raphael and J. Tang, Eds., vol. 7534. Springer, 2012, pp. 441–452.

[25] S. Chumakov, C. Belapurkar, C. Putonti, T.-B. Li, B. M. Pettitt, G. E. Fox, R. C. Willson, and Y. Fofanov.

[26] D. R. Zerbino and E. Birney, "Velvet: Algorithms for de novo short read assembly using de Bruijn graphs," *Genome Research*, vol. 18, no. 5, pp. 821–829, May 2008.

[27] A. Rosenberg and J. Hirschberg, "V-measure: A conditional entropy-based external cluster evaluation measure," in *2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, 2007, pp. 410–420.