Statistica Sinica Preprint No: SS-2023-0159	
Title	VALISE: A Robust Vertex Hunting Algorithm
Manuscript ID	SS-2023-0159
URL	http://www.stat.sinica.edu.tw/statistica/
DOI	10.5705/ss.202023.0159
Complete List of Authors	Dieyi Chen, Tracy Ke and Shuyi Zhang
Corresponding Authors	Shuyi Zhang
E-mails	syzhang@fem.ecnu.edu.cn

Statistica Sinica

VALISE: A Robust Vertex Hunting Algorithm

Dieyi Chen*, Tracy Zheng Ke* and Shuyi Zhang†

*Department of Statistics, Harvard University, Cambridge, MA 02138, USA

KLATASDS-MOE, School of Statistics,

Academy of Statistics and Interdisciplinary Sciences,

East China Normal University, Shanghai 200062, China

Abstract: Given data vectors $X_1, \dots X_n \in \mathbb{R}^r$, where X_i is a noisy observation of X_i^*, and $X_1^*, \dots X_n^*$ are contained in an unknown simplex with K vertices, vertex hunting (VH) is the problem of estimating the vertices of the true simplex. VH is a building block of several algorithms in hyperspectral remote sensing, soft clustering, topic modeling, and network mixed membership estimation. The popular VH algorithms are susceptible to outliers, whose estimation errors are governed by $\max_i \|X_i - X_i^*\|$. We propose a robust VH algorithm that properly shrinks estimated vertices towards the interior of data cloud, so as to mitigate the effect of outliers. The level of shrinkage is determined by maximizing a pseudo likelihood and has no tuning parameter. We show that, when the barycentric coordinates of $X_1^*, \dots, X_n^*$ come from a Dirichlet distribution, the proposed method has a faster rate of convergence than several popular VH algorithms.

Key words and phrases: Archetypal analysis, admixture, Dirichlet distribution, endmember extraction, gradient descent, linear unmixing, mixed membership estimation, nonnegative matrix factorization, pseudo likelihood, topic modeling.

Statistica Sinica: Preprint
doi:10.5705/ss.202023.0159

1. Introduction

Let $\mathcal{S} \subset \mathbb{R}^r$ be a K-vertex simplex, whose vertices are denoted by $V_1, V_2, \ldots, V_K$. Suppose $X_1^*, X_2^*, \ldots, X_n^*$ are non-stochastic vectors in this simplex. Equivalently, each X_i^* is a convex combination of the vertices:

$$X_i^* = \sum_{k=1}^{K} \pi_i(k) V_k, \qquad 1 \leq i \leq n. \tag{1.1}$$

Here $\pi_i = (\pi_i(1), \ldots, \pi_i(K))' \in \mathbb{R}_+^K$ is a vector in the standard simplex such that $0 \leq \pi_i(k) \leq 1$ for $0 \leq k \leq K$ and $\sum_{k=1}^K \pi_i(k) = 1$. The entries of π_i are called the barycentric coordinates of X_i^*. We call an algorithm a Vertex Hunting (VH) algorithm if it does the following job: Given input $X_1, X_2, \ldots X_n \in \mathbb{R}^r$, where each X_i is a noisy observation of X_i^*, the algorithm outputs the estimated vertices $\hat{V}_1, \hat{V}_2, \ldots, \hat{V}_K$.

Vertex hunting has a lot of applications in hyperspectral unmixing (Bioucas-Dias et al., 2012). A hyperspectral image is a mixture of signals from different pure materials, and vertex hunting algorithms are used to find the spectral signature of each pure material. Another application of vertex hunting is archetypal analysis (Cutler and Breiman, 1994), which targets to represent each data vector as a mixture of *archetypes* and is a popular tool for learning the admixture structure on biological data (Van Dijk et al., 2018). Vertex hunting is also a building block of algorithms for network mixed membership estimation (Jin et al., 2023), topic modeling (Arora et al., 2012; Ke and Wang, 2022), and nonnegative matrix factorization (Javadi and Montanari, 2019).

Some popular vertex hunting algorithms include the minimum volume transform (MVT) (Craig, 1994), N-FINDER (Winter, 1999), successive projection (SP) (Araújo et al., 2001), and archetypal analysis (AA) (Cutler and Breiman, 1994). SP is a greedy

Statistica Sinica: Preprint
doi:10.5705/ss.202023.0159

algorithm that finds one vertex at a time; at each iteration, it projects data vectors using previously found vertices and finds the next vertex by maximizing the Euclidean norm. The other three methods solve constrained optimizations. N-FINDER restricts the vertices of the simplex to be placed on data points and maximizes the volume of the simplex. MVT restricts that the simplex contains all the data points in the interior and minimizes the volume of the simplex. AA restricts that the vertices are in the convex hull of the data cloud and minimizes the sum of squared Euclidean distances from data vectors to the simplex.

However, these methods are unsatisfactory with the presence of strong noise or outliers. Take SP for example. If there is one X_i located far away from X_i^* such that $\|X_i - X_i^*\|$ is large, this X_i is likely to be picked by the algorithm as a vertex, resulting in a large error. This is also confirmed by theory. The estimation errors of these methods are at the same order as $\max_{1\leq i\leq n} \|X_i - X_i^*\|$ (Jin et al., 2023). Consider an example where $X_i - X_i^*$ are generated independently from $\mathcal{N}(0, \sigma^2)$. The error is $O_{\mathbb{P}}\big\{\sigma\sqrt{\log(n)}\big\}$. There is even no consistency.

Jin et al. (2023) observed empirically that using the k-means clustering to "sketch" the data cloud can significantly improve the performance of vertex hunting. They first applied k-means assuming L clusters for some $L > K$ and then conducted vertex hunting using the L cluster centers. Since each cluster center is an *average* of a large number of data vectors, they conjectured that the error can be reduced to $O(1/\sqrt{n})$ when the noise vectors $X_i - X_i^*$ are independent. However, they did not prove this conjecture. Furthermore, their method requires a tuning integer L. How to choose L in practice is unclear.

In this paper, we prove the conjecture by Jin et al. (2023) and show that the error of vertex hunting can indeed be reduced to $O(1/\sqrt{n})$, under some additional assumptions. We also propose a new vertex hunting algorithm that attains this error rate and does not need any tuning parameter.

In (1.1), write $Z_i = X_i - X_i^*$, $Z = (Z_1, Z_2, \ldots, Z_n) \in \mathbb{R}^{r\times n}$, $\Pi = (\pi_1, \pi_2, \ldots, \pi_n) \in \mathbb{R}^{K\times n}$ and $V = (V_1, V_2, \ldots, V_K) \in \mathbb{R}^{r\times K}$. Then $X_i = V\pi_i + Z_i$. For ease of reading, we provide a summary of major notations used in the paper in Table S1 in the supplementary material. We consider a running model where

$$\pi_i \overset{iid}{\sim} \text{Dirichlet}(\alpha), \quad Z_i \overset{iid}{\sim} \mathcal{N}_r(0, \sigma^2 I_r), \quad \Pi \text{ and } Z \text{ are independent.} \tag{1.2}$$

The pseudo likelihood is defined by

$$L(V, \sigma^2, \alpha) = \prod_{i=1}^{n} \int f(X_i|\pi_i; V, \sigma^2) f(\pi_i; \alpha) d\pi_i, \tag{1.3}$$

where $f(X_i|\pi_i; V, \sigma^2)$ is the density of $\mathcal{N}_r(V\pi_i, \sigma^2 I_r)$ and $f(\pi_i; \alpha)$ is the density of a Dirichlet distribution (supported on the standard simplex of $\mathbb{R}^K$) with parameters $\alpha = (\alpha_1, \alpha_2, \ldots, \alpha_K)'$. Let

$$(\hat{V}, \hat{\sigma}^2, \hat{\alpha}) \in \text{argmax}_{(V,\sigma^2,\alpha)}\big\{L(V, \sigma^2, \alpha)\big\}. \tag{1.4}$$

We use $\hat{V} = (\hat{V}_1, \hat{V}_2, \ldots, \hat{V}_K)$ as the estimates of vertices. The optimization (1.4) is solved by an alternating gradient ascent algorithm, which updates $\hat{V}$, $\hat{\sigma}^2$, and $\hat{\alpha}$ successively, each via a gradient ascent step, until the pseudo-likelihood converges. We call the method the Vertex Analysis via LIkelihood-assisted ShrinkagE (VALISE). Details will be given in Section 2. A simulated model is illustrated in Figure 1.

There are two main differences between VALISE and popular vertex hunting algorithms. First, our method encourages a shrinkage of the simplex towards the interior of

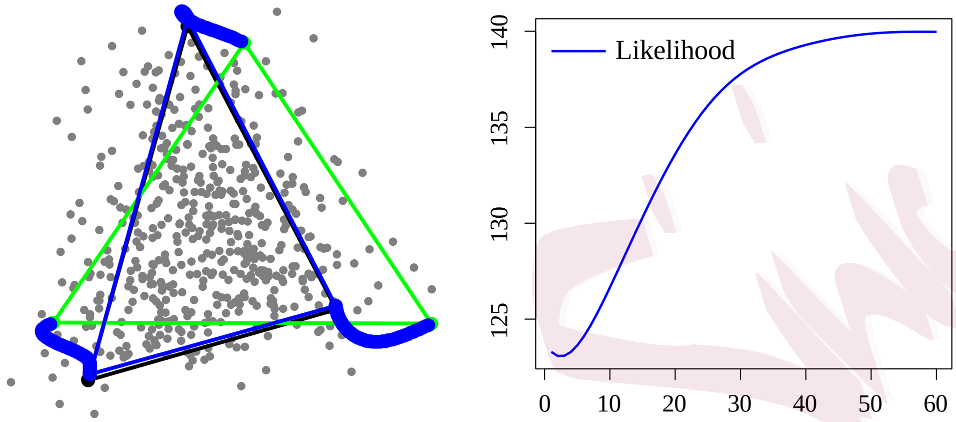

Figure 1: A simulation example. Left: SP (green) and VALISE (blue); right: Likelihood. The true simplex is marked in black but highly overlapped with the blue VALISE simplex. In this example, the data are generated from Models (1.1)-(1.2). VALISE is initialized by SP. All three vertices are outside the true simplex at the beginning, and they move towards the true ones during the iterations. When the likelihood $L(V, \sigma^2, \alpha)$ converges, the vertices stop moving and stay close to the true ones.

data cloud. The level of shrinkage is determined by $\hat{\sigma}^2$. In contrast, other algorithms heavily penalize any data point outside the simplex; as a result, when the data are noisy or there are outliers, the estimated simplex can be much larger than the true one. Second, our method efficiently uses those data points in the interior. For most other methods, data deep into the interior have little effect on the output, and these data vectors are wasted. While the interior data points carry little information of the vertices, they do carry information of the noise level and can be used to decide the proper level of shrinkage of the simplex (e.g., for estimating σ^2). This is why our method can be tuning free and achieve a faster error rate.

We investigate the theoretical properties of VALISE under the correctly specified model. In a correctly specified model, (1.2) is the true data generating process, and the

Statistica Sinica: Preprint
doi:10.5705/ss.202023.0159

pseudo likelihood becomes the true likelihood. We measure the performance of vertex hunting by $D(\hat{V}, V) \equiv \min_\tau \max_{1\le k\le K} \|\hat{V}_k - V_{\tau(k)}\|$, where the first minimum is taken over all permutations of $\{1, 2, \ldots, K\}$. We show that $D(\hat{V}, V) = O_{\mathbb{P}}(\sigma/\sqrt{n})$ and derive the asymptotic normality of $\hat{V}$.

The running model (1.2) is reminiscent of the latent Dirichlet allocation (Blei et al., 2003; Airoldi et al., 2008) and variational EM algorithms for topic modeling and network mixed membership estimation. However, they are restricted to particular applications, but our method is suitable for any application that uses vertex hunting as a module. For example, in Section 5, we show that our method can be combined with various spectral methods for different applications. We maximize (1.3) by gradient ascent instead of variational EM. One reason is that the Dirichelt distribution is not a conjugate prior of the Gaussian distribution, and it remains unclear how to choose the variational family.

The remaining of this paper is organized as follows. In Section 2, we describe the VALISE algorithm and explain its rationale. Section 3 contains the main theoretical results. Section 4 contains simulations and Section 5 contains real data results. Discussions can be found in Section 6.

2. The Method

2.1 The pseudo likelihood and its interpretation

Let $\phi_r(x; \mu, \Sigma)$ be the density of an r-variate normal distribution $\mathcal{N}_r(\mu, \Sigma)$, and let $\mathbb{E}_{\pi\sim\mathrm{Dir}(\alpha)}$ denote the expectation with respect to π that has a Dirichlet distribution. Let $\ell(V, \sigma^2, \alpha) = -\log L(V, \sigma^2, \alpha)$ be the minus pseudo log-likelihood. Then, (1.3) can be re-written as

$$\ell(V,\sigma^2,\alpha) = -\sum_{i=1}^{n} \log \mathbb{E}_{\pi_i\sim \mathrm{Dir}(\alpha)}\{\phi_r(X_i; V\pi_i, \sigma^2 I_r)\}. \tag{2.1}$$

We treat (2.1) as an empirical loss function to minimize. Here is an illustrating example.

Proposition 1. Suppose $V_1, V_2, \ldots, V_K$ are affinely independent. When $r = K-1$ and $\alpha = (1,1,\ldots,1)'$,

$$\ell(V,\sigma^2,\alpha) = \sum_{i=1}^{n} -\log\Big\{\int_{x\in\mathcal{S}} \exp\Big(-\frac{1}{2\sigma^2}\|X_i - x\|^2\Big)dx\Big\} + n\log \mathrm{Vol}(\mathcal{S}) + \frac{nr}{2}\log(\sigma^2) + C,$$

where $\mathcal{S}$ is the simplex spanned by V and C does not depend on (V,σ^2,α).

In $\ell(V,\sigma^2,\alpha)$, the first term is a measure of goodness-of-fit. If the simplex is too small so that some X_i are far outside, then $\|X_i - x\|$ is uniformly large for $x \in \mathcal{S}$, yielding a large value in this term. The second term is a penalty on the volume of $\mathcal{S}$, preventing the simplex to be too large. The trade-off between the first two terms is controlled by σ^2, and the third term is a penalty on σ^2.

In the above example, $\alpha = (1,1,\ldots,1)'$, and the Dirichlet density reduces to a uniform density on the simplex. We consider a modification of the empirical loss by changing $\int_{x\in\mathcal{S}} \exp\big(-\frac{1}{2\sigma^2}\|X_i - x\|^2\big)dx$ to $\mathrm{Vol}(\mathcal{S})\cdot \sup_{x\in\mathcal{S}}\big\{\exp\big(-\frac{1}{2\sigma^2}\|X_i - x\|^2\big)\big\}$. The latter becomes $\mathrm{Vol}(\mathcal{S})$ for any X_i in the interior of the simplex, i.e., the locations of the interior data vectors do not matter, and the loss is only determined by those data vectors outside the simplex. Specifically, for a outlier X_i and an interior X_j, $\|X_i - x\|$ is much larger than $\|X_j - x\|$ at any $x \in \mathcal{S}$, which makes the i-th summation term dominate the value of the first summation. The further the outlier is from the simplex, the greater its effect on empirical loss is. Such an empirical loss will be sensitive to outliers.

In the general scenario, the empirical loss is described in (2.5), and the Dirichlet density plays a key role in utilizing interior data. By choosing α, we adjust the weight of data in the empirical loss by its distance to each vertex. For example, if outliers occur more around the k-th vertex than other vertices, α_k could be chosen larger than $\alpha_{k'}$, $k' \neq k$. A data-driven strategy to choose α is provided in VALISE+ as shown in the following. Overall, the current form takes advantage of the large number of interior data vectors, thus more robust to outliers.

2.2 VALISE: an alternating gradient descent algorithm

We propose an algorithm for minimizing $\ell(V, \sigma^2, \alpha)$. First, we update σ^2 given (V, α). Note that

$$\nabla_{\sigma^2}\ell(V, \sigma^2, \alpha) = -\sum_{i=1}^{n} \frac{\mathbb{E}_{\pi_i \sim \mathrm{Dir}(\alpha)}\{\phi_r(X_i; V\pi_i, \sigma^2 I_r)\|X_i - V\pi_i\|^2\}}{2(\sigma^2)^2\, \mathbb{E}_{\pi_i \sim \mathrm{Dir}(\alpha)}\{\phi_r(X_i; V\pi_i, \sigma^2 I_r)\}} + \frac{nr}{2\sigma^2}.$$

By letting $\nabla_{\sigma^2}\ell(V, \sigma^2) = 0$, we obtain an explicit formula:

$$\sigma^2_{\text{update}}(V, \sigma^2) = \frac{1}{nr}\sum_{i=1}^{n} \frac{\mathbb{E}_{\pi_i \sim \mathrm{Dir}(\alpha)}\{\phi_r(X_i; V\pi_i, \sigma^2 I_r)\|X_i - V\pi_i\|^2\}}{\mathbb{E}_{\pi_i \sim \mathrm{Dir}(\alpha)}\{\phi_r(X_i; V\pi_i, \sigma^2 I_r)\}}. \tag{2.2}$$

Next, we update $V = (V_1, V_2, \ldots, V_K)$ using the gradient descent. By direct calculations,

$$\nabla_V \ell(V, \sigma^2) = -\sum_{i=1}^{n} \frac{\mathbb{E}_{\pi_i \sim \mathrm{Dir}(\alpha)}\{\phi_r(X_i; V\pi_i, \sigma^2 I_r)(X_i - V\pi)\pi'\}}{\sigma^2\, \mathbb{E}_{\pi_i \sim \mathrm{Dir}(\alpha)}\{\phi_r(X_i; V\pi_i, \sigma^2 I_r)\}}. \tag{2.3}$$

Note that in (2.3) we have arranged the gradient as an $r \times K$ matrix. We apply Nadam (Dozat, 2016) for the update of V, which is an improvement of the gradient descent algorithm by combining the adaptive learning rate, momentum and Nesterov acceleration. At iteration t, it first updates m_t and v_t, where m_t is a convex combination of m_{t-1} and $\nabla_V \ell$, and v_t is a convex combination of v_{t-1} and $(\nabla_V \ell)^2$. It then updates

V by $V_t = V_{t-1} - \eta \cdot \hat{m}_t/(\sqrt{\hat{v}_t} + \epsilon)$, where $\hat{m}_t$ and $\hat{v}_t$ are the rescaled version of m_t and v_t. In the above, all the operations on matrices are element-wise. Details are in Algorithm 1. We set the algorithm parameters as the default ones in Dozat (2016).

This algorithm needs to calculate (2.2) and (2.3) at each iteration. To estimate the integrals in both expressions, we use the Monte Carlo (MC) approximation. For any π in the standard simplex of $\mathbb{R}^K$, define

$$\hat{w}_i(\pi) = \hat{w}_i(\pi; V, \sigma^2, \alpha) \equiv \phi_r(X_i; V\pi, \sigma^2 I_r) \cdot f_{\mathrm{Dir}(\alpha)}(\pi), \qquad 1 \le i \le n.$$

Let $\{\pi_j\}_{j=1}^N$ be independent and identically generated samples of size N from a density $q(\cdot)$ on the standard simplex. We approximate (2.2) and (2.3) by

$$\widehat{\sigma^2}_{\text{update}}(V, \sigma^2) = \frac{1}{nr} \sum_{i=1}^{n} \frac{\sum_{j=1}^{N} \hat{w}_i(\pi_j)/q(\pi_j) \cdot \|X_i - V\pi_j\|^2}{\sum_{j=1}^{N} \hat{w}_i(\pi_j)/q(\pi_j)} \quad \text{and}$$

$$\widehat{\nabla_V \ell}(V, \sigma^2) = -\sum_{i=1}^{n} \frac{\sum_{j=1}^{N} \hat{w}_i(\pi_j)/q(\pi_j) \cdot (X_i - V\pi_j)\pi_j'}{\sum_{j=1}^{N} \hat{w}_i(\pi_j)/q(\pi_j)},$$

respectively. The MC sample $\{\pi_j\}_{j=1}^N$ is shared across $1 \le i \le n$ to reduce the running time. It is possible to use different MC samples for different i, or even more advanced MC methods such as importance sampling, but we do not see a significant numerical advantage. We usually set $q(\cdot)$ as the uniform distribution on the standard simplex as an informative prior unless additional information is known.

We now have an algorithm for computing $(\hat{V}, \hat{\sigma}^2)$ for a given α. Then we optimize α by the grid search. Optimizing (V, σ^2, α) together is also tried in the Nadam algorithm, but the grid search has a better numerical performance.

Algorithm 1: *A likelihood based vertex hunting algorithm*, maximum likelihood estimation using Nadam. Good default settings for the tested machine learning problems are $\eta = 0.001$, $\beta_1 = 0.9$, $\beta_2 = 0.999$ and $\epsilon = 10^{-8}$. All operations on matrices are element-wise. With β_1^t and β_2^t we denote β_1 and β_2 to the power of t

Input: η: Step size
Input: $\beta_1, \beta_2 \in [0, 1)$: Exponential decay rates for the moment estimates
Input: $l^*(V, \sigma^2)$: Stochastic objective function with parameter V
Input: V_0, σ_0^2: Initial parameter
Initialize: $m_0 \leftarrow 0(1^{st}$ moment), $v_0 \leftarrow 0$ (2^{nd} moment), $t \leftarrow 0$ (time step), $\hat{m}_0 \leftarrow 0$, $\hat{v}_0 \leftarrow 0$.
while V_t *not converge* **do**
    $t \leftarrow t + 1$
    $g_t \leftarrow \nabla_V l_t^*(V_{t-1} - \eta \cdot \hat{m}_{t-1}/(\sqrt{\hat{v}_{t-1}} + \epsilon), \sigma_{t-1}^2)$
    $m_t \leftarrow \beta_1 \cdot m_{t-1} + (1 - \beta_1) \cdot g_t$
    $v_t \leftarrow \beta_2 \cdot v_{t-1} + (1 - \beta_2) \cdot g_t^2$
    $\hat{m}_t \leftarrow m_t/(1 - \beta_1^t)$
    $\hat{v}_t \leftarrow v_t/(1 - \beta_2^t)$
    $V_t \leftarrow V_{t-1} - \eta \cdot \hat{m}_t/(\sqrt{\hat{v}_t} + \epsilon)$ (Update V)
    $\sigma_t^2 \leftarrow \sigma_{\text{update}}^2(V_t, \sigma_{t-1}^2)$ (Update σ^2)
return (V_t, σ_t^2)

3. Theoretical Properties

In this section, we minimize the minus pseudo log-likelihood in (2.1) over (V, σ^2, α) and show the root-n consistency of the maximum likelihood estimator (MLE). The model that our algorithm implements is a special case where α is given. To simplify the notation, below, we assume $r = K - 1$.

Remark 1. To form a valid simplex in the r-dimensional sample space, the true K vertices should be affinely independent, which is allowable only when $r \geq K - 1$. If $r < K - 1$, the affine dependence between vertex vectors leads to non-identifiability in theory and non-convergence in optimization. In practice, a r-dimensional sample might be from a simplex spanned by $K-1, K-2$ or less vertices, among which the simplex spanned by $K-1$ vertices is largest, most complicated and most informative. This is the reason why we assume $r = K-1$ throughout the paper.

3.1 The case of a correctly-specified model

In this section, we study the theoretical properties of the estimators when the model is correctly-specified. Let $\theta = (\mathrm{vec}(V)', \sigma^2, \alpha)' \in \mathcal{H}$ be the parameter vector where $\mathrm{vec}(V) = (V_1', V_2', \ldots, V_K')'$ represents the vectorization of the matrix V and $\mathcal{H}$ is the parameter space. Let $\{f(\cdot, \theta), \theta \in \mathcal{H}\}$ be a family of distributions such that

$$f(x; \theta) = \mathbb{E}_{\pi \sim \mathrm{Dir}(\alpha)}\{\phi_r(x; V\pi, \sigma^2 I_r)\}.$$

Then the minus pseudo log-likelihood function in (2.1) can be written as $\ell(\theta) = -\sum_{i=1}^n \log f(X_i; \theta)$. Let $\ell^*(\theta) = -\mathbb{E}_{X \sim f(x;\theta^*)}\{\log f(X; \theta)\}$ where $f(x; \theta^*)$ is the unknown underlying model that X_i follows. Let $\hat{\theta}_n$ be the value which minimizes $\frac{1}{n}\ell(\theta)$. Then $\hat{\theta}_n$ is the MLE that maximizes the log-likelihood. To derive the theoretical properties, we introduce the following assumptions on parameters.

Assumption 1. The parameter space $\mathcal{H}$ is compact. There exist positive constants $\epsilon_0^\sigma, \epsilon_0^\alpha, M^\sigma$ such that $\epsilon_0^\sigma \le \sigma^{2,*} \le M^\sigma$ and $\min_{1\le k \le K} \alpha_k^* \ge \epsilon_0^\alpha$.

Assumption 2. Let $\mathbb{V} = (\mathbf{1}_K', (V^*)')'$ and $\lambda_{\min}(\mathbb{V})$ be its minimum absolute eigenvalue. There exists $\epsilon_0^v > 0$ such that $\lambda_{\min}(\mathbb{V}) > \epsilon_0^v$.

Assumption 1 requires positive lower bounds of σ^2 and α_k. This is essentially required by model identifiability. The small σ^2 makes it difficult to estimate the variance of the noise. If some α_k is close to zero, the induced Dirichlet distribution is degenerate and we could hardly identify the corresponding vertex. Under Assumption 2, the matrix $\mathbb{V}$ is invertible so that $V_1^*, V_2^*, \ldots, V_K^*$ can be served as vertices of a simplex. Assumptions 1 and 2 imply that the Fisher information matrix $I_\theta = -\mathbb{E}_{X\sim f(x;\theta)}\{\nabla_{\theta\theta} \log f(X;\theta)\}$ is invertible at the true parameter value θ^*. In fact, when $r = 1, K = 2$ and $\alpha_1 = \alpha_2 = 1$, it can be derived that $I_{\mathrm{vec}(\mathrm{V}),\sigma^2} = \mathrm{diag}\{(\mathbf{1}_2)^{\otimes 2}/(4\sigma^2), 1/(2\sigma^4)\}$ if $\lambda_{\min}(\mathbb{V}) = 0$. When $\lambda_{\min}(\mathbb{V})$ is close to zero, the true simplex tends to degenerate in the high-dimensional space. Thus we should reduce the dimension to observe the data. Assumption 2 actually implies that the true simplex should be large enough to be recognized in the observed space. Assumptions 1 and 2 can be easily checked and are mild technical conditions for deriving the consistency and asymptotic normality of the proposed estimators.

Let $W = (V', \alpha')'$. Then $f(x;\theta)$ is invariant under permutations of the columns of W. Let $d(\hat{\theta}_n, \theta^*) = \min_\tau \max_{1\le k\le K} \|\hat{W}_{n,k} - W^*_{\tau(k)}\| + |\hat{\sigma}_n^2 - \sigma^{2,*}|$, where the first minimum is taken over all permutations of $\{1, 2, \ldots, K\}$. Then we use $d(\cdot,\cdot)$ to measure the performance of vertex hunting. The following theorem shows the estimation error.

Theorem 1. Assume θ^* is an inner point of $\mathcal{H}$. Suppose Assumptions 1 and 2 hold. Then $d(\hat{\theta}_n, \theta^*) = O_{\mathbb{P}}(n^{-1/2})$. In particular, there exists a sequence of permutations τ_n of $\{1, 2, \ldots, K\}$ such that $\|\hat{\theta}_{n,\tau_n} - \theta^*\| = O_{\mathbb{P}}(n^{-1/2})$.

Remark 2. The traditional consistency theorem typically handles the L_2 distance over the real-valued parameter space. In our model, we consider the continuity of the density function $f(x;\theta)$ with respect to the equivalence class θ under the new distance $d(\cdot,\cdot)$. The resultant consistency theorem provides a sequence of representatives $\hat{\theta}_{n,\tau_n}$ that converges in probability to θ^* under the L_2 distance.

Theorem 1 can be used to derive the asymptotic normality. Before that, we provide the form of the Fisher information matrix. Let $\psi(\cdot)$ and $\psi^{(1)}(\cdot)$ be the digamma and trigamma functions, respectively. Let $G_1(\alpha) = (\psi(\alpha_1), \ldots, \psi(\alpha_K))'$, $G_2(\pi) = (\log \pi_1, \ldots, \log \pi_K)'$, $G_3(\alpha) = \mathrm{diag}\{\psi^{(1)}(\alpha_1), \psi^{(1)}(\alpha_2), \ldots, \psi^{(1)}(\alpha_K)\}$ be a diagonal matrix, $G_4(\alpha) = \mathbb{E}_{\pi\sim\mathrm{Dir}(\alpha)}\{G_2(\pi)G_2'(\pi)\}$ with the (k_1, k_2)-th coordinate being $G_{4,k_1k_2}(\alpha) = \psi^{(1)}(\alpha_{k_1})\delta_{k_1k_2} - \psi^{(1)}(\|\alpha\|_1) + \{\psi(\alpha_{k_1}) - \psi(\|\alpha\|_1)\}\{\psi(\alpha_{k_2}) - \psi(\|\alpha\|_1)\}$ and $\delta_{k_1k_2} = \mathbb{I}(k_1 = k_2)$. Let $M(x;\theta) = (M_1'(x;\theta), M_2'(x;\theta), M_3'(x;\theta))'$ where

$$M_1(x;\theta) = (\sigma^2)^{-1}\mathbb{E}_{\pi\sim\mathrm{Dir}(\alpha)}\{\pi \otimes (x - V\pi)\phi_r(x; V\pi, \sigma^2 I_r)\},$$

$$M_2(x;\theta) = 2^{-1}(\sigma^2)^{-2}\mathbb{E}_{\pi\sim\mathrm{Dir}(\alpha)}\{\|x - V\pi\|^2\phi_r(x; V\pi, \sigma^2 I_r)\} \quad \text{and}$$

$$M_3(x;\theta) = \mathbb{E}_{\pi\sim\mathrm{Dir}(\alpha)}\{G_2(\pi)\phi_r(x; V\pi, \sigma^2 I_r)\}.$$

Furthermore, define $J(\theta) = \mathrm{diag}\{0, J_0(\theta)\}$ where

$$J_0(\theta) = \begin{pmatrix} -r^2(2\sigma^2)^{-2} & r(2\sigma^2)^{-1}\{\psi(\|\alpha\|_1) - G_1'(\alpha)\} \\ r(2\sigma^2)^{-1}\{\psi(\|\alpha\|_1) - G_1(\alpha)\} & G_3(\alpha) - G_4(\alpha) - \psi^{(1)}(\|\alpha\|_1) \end{pmatrix},$$

Then the Fisher information matrix can be formulated as

$$I_\theta = J(\theta) + \int f(x;\theta)^{-1}M(x;\theta)M'(x;\theta)dx.$$

Now we provide the asymptotic normality of $\hat{\theta}_{n,\tau_n}$ as follows.

Theorem 2. Assume θ^* is an inner point of $\mathcal{H}$. Suppose Assumptions 1 and 2 hold. Let τ_n be the sequence of permutations as in Theorem 1. Then $\sqrt{n}(\hat{\theta}_{n,\tau_n} - \theta^*)$ is asymptotically normal with mean zero and covariance matrix $I_{\theta^*}^{-1}$.

Remark 3. Theorem 2 shows that the error of the proposed vertex hunting algorithm can be reduced to $O_{\mathbb{P}}(n^{-1/2})$ under mild conditions. The root-n consistency is an appealing property by using the likelihood-based approach, which is unavailable for the SP

Statistica Sinica: Preprint
doi:10.5705/ss.202023.0159

and AA approaches. By efficiently using the interior data points, our proposed method can achieve a faster convergence rate of all parameters of interests, and the asymptotic variance-covariance matrix attains the Cramér-Rao lower bound. This shows the significant theoretical advantages of VALISE.

Remark 4. For our model, the likelihood is a Gaussian density taken expectation with respect to the barycentric coordinates π, which follows a Dirichlet distribution. We can regard the Dirichlet distribution as a prior to the mean vector of the Gaussian distribution, but it is not a conjugate prior. Thus the resultant density has an integral form that hardly obtains an analytic expression and is hard to analyze. Therefore, many straightforward properties such as continuity, bounded score function and Fisher information matrix need to be carefully examined.

Remark 5. The computations of the first and second derivatives of the log-likelihood are rather tedious due to the integral form of $f(x;\theta)$ and the high parameter dimensionality. The asymptotic variance-covariance matrix of the MLE can be derived by the formula of the inverse of the partitioned matrix, but the form is very complicated and we will not present in the main paper.

In this following, we propose a proposition of the effect of α on the Fisher information matrix I_θ and hence the asymptotic variance-covariance matrix $I_{\theta^*}^{-1}$.

Corollary 1. Suppose the conditions in Theorem 2 hold. Consider the case when $\alpha_1 = \alpha_2 = \cdots = \alpha_K$. If $\|\alpha\|_1 \to \infty$, the Fisher information matrix I_θ satisfies

$$-\mathbb{E}_{X\sim f(x;\theta)}\{\nabla_{\mathrm{vec}(V),\mathrm{vec}(V)} \log f(X;\theta)\} \to \frac{1}{K^2\sigma^{2,*}}(\mathbf{1}_K\mathbf{1}_K') \otimes I_r,$$
$$-\mathbb{E}_{X\sim f(x;\theta)}\{\nabla_{\sigma^2,\sigma^2} \log f(X;\theta)\} \to \frac{r}{2(\sigma^{2,*})^2},$$

and all the other elements of I_θ converge to zero. The asymptotic Fisher information matrix has rank $r+1$ and hence is not invertible, which makes the elements of the asymptotic variance-covariance matrix $I_{\theta^*}^{-1}$ go to infinity.

Remark 6. Corollary 1 shows that as α goes to infinity, the inverse of the asymptotic variance-covariance matrix of the MLE for V goes to $(\sigma^{2,*})^{-1}K^{-2}(\mathbf{1}_K\mathbf{1}_K')\otimes I_r$ which is not invertible. Thus the variance goes to infinity and hence the vertices are not estimable. In fact, when $\alpha_1=\alpha_2=\cdots=\alpha_K$ and $\|\alpha\|_1\to\infty$, $f(\pi;\alpha)$ goes to a point mass concentrating at $\mathbf{1}_K/K$. The rate of convergence is very fast. For example, when $r=1$ and $K=2$, if $\alpha_1=\alpha_2=m\in\mathbb{Z}_+$, the rates of $f(\mathbf{1}_2/2;\alpha)\to\infty$ and $f(\pi;\alpha)\to 0$ for $\pi\neq\mathbf{1}_2/2$ are faster than $O(m^{1/2})$ and $O(m^{-1/2})$, respectively. For large α, the variance can be extremely large, leading to instability of the vertex estimation. This is concordant with the intuition. When α is large, most of the observed data are in the interior of the simplex and there are few data points around each vertex, which makes the vertices hard to estimate.

Thus in the simulation and real data application, we only consider the case when $\alpha_k\leq 1$ for $k=1,2,\ldots,K$. If $\alpha_k<1$, there are more observed data points concentrating around the vertex V_k than in the interior. If $\alpha_1=\alpha_2=\cdots=\alpha_K=1$, the Dirichlet distribution is uniform over the simplex which can be regarded as an informative prior.

3.2 The case of a misspecified model

In this section, we study the effect of misspecification of α. In (1.2), we assume π_i's are *iid* generated from a Dirichlet distribution with parameter α. However, in practice, the true α is unknown and even π_i's do not follow a Dirichlet distribution. In particular,

Statistica Sinica: Preprint
doi:10.5705/ss.202023.0159

we consider the case when $\alpha = \mathbf{1}_K$ is used to estimate V and σ^2. Below, we generalize the definition of efficient vertex hunting in Jin et al. (2023) and show our proposed estimators are still efficient under the misspecification of α.

Definition 1. (Stochastically efficient vertex hunting). A Vertex Hunting algorithm is stochastically efficient if for any $\epsilon > 0$, there exists $C > 0$ such that $\mathbb{P}(\max_k \|\hat{V}_k - V_k^*\| \leq C \max_i \|X_i - X_i^*\|) \geq 1 - \epsilon$.

The theoretical analysis under the misspecified model is tough and complicated, since the objective function as given in Proposition 1 has an integral form. The key of the analysis is to derive a sharp bound of the log-likelihood function by using the pure nodes, whose definition is provided as follows.

Definition 2. (Pure nodes and mixed nodes). A node i is called a pure node of community k if $\pi_i(k) = 1$ and $\pi_i(k') = 0$, $k' \neq k$, and is called a mixed node if $\max_{1 \leq k \leq K} \pi_i(k) < 1$.

Enlightened by the work of Jin et al. (2023), two settings are considered:

- Setting 1: The true π_i's are fixed and each vertex corresponds to sufficient pure nodes which form some clusters.
- Setting 2: The true π_i's are random and have no clustering structure.

First, we discuss Setting 1. Let $\mathcal{M} = \{1 \leq i \leq n : \max_{1 \leq k \leq K} \pi_i(k) < 1\}$ be the set of all mixed nodes and $|\mathcal{M}|$ be its cardinality. We make the following assumptions.

Assumption 3. Let $\mathcal{N}_k(\eta) = \{1 \leq i \leq n : \|X_i^* - V_k\| \leq \eta\}$ be the set of "nearly" pure nodes and $|\mathcal{N}_k(\eta)|$ be its cardinality. There exists $c_1 > 0$ such that $\min_{1\leq k\leq K} |\mathcal{N}_k(\eta_n)| \geq c_1 r_n$ where $\eta_n = o(\max_{k\neq l} \|V_k^* - V_l^*\|)$ and $r_n^{-1} = o(1)$.

Assumption 4. Define $(\tilde{V}, \tilde{\sigma}^2) \in \mathrm{argmax}_{(V,\sigma^2)}\big\{L(V, \sigma^2, \mathbf{1}_K)\big\}$. There exists $c_2 > 0$ such that $\tilde{\sigma}^2/\sigma^{2,*} \xrightarrow{\mathbb{P}} c_2$ as $n \to \infty$.

It is noticeable that r_n in Assumption 3 is allowed to be at a smaller order of n, and c_2 in Assumption 4 is unnecessarily equal to one, so as to allow over-estimates ($c_2 > 1$) or under-estimates ($c_2 < 1$) of $\sigma^{2,*}$.

Theorem 3. (Stochastic efficiency under misspecification, Setting 1). Assume $r = K - 1$ is fixed. Suppose the true vertices $V_1^*, V_2^*, \ldots, V_K^*$ and the estimated vertices $\tilde{V}_1, \tilde{V}_2, \ldots, \tilde{V}_K$ are affinely independent. Let $\mathcal{S}^*$ and $\tilde{\mathcal{S}}$ be the simplex spanned by V^* and $\tilde{V}$ respectively. Consider the case where $\mathcal{S}^* \subset \tilde{\mathcal{S}}$. Additionally, suppose Assumptions 3 and 4 hold. Then for any $\epsilon > 0$, there exists a positive constant C_0 satisfying $\log^{-1} C_0 = o(r_n/n)$ such that

$$\mathbb{P}\left(\max_k \|\tilde{V}_k - V_k^*\|^2 \leq C_0 \sum_{i=1}^n \|X_i - X_i^*\|^2/n\right) \geq 1 - \epsilon.$$

Next, we consider Setting 2. Let $\mathcal{S}_0$ be the simplex spanned by standard basis vectors of $\mathbb{R}^K$. Under the misspecification case, Assumption 3 is replaced by the following Assumption 5.

Assumption 5. Assume π_i's are *iid* sampled from a mixture

$$f_n(\pi) = \sum_{k=1}^K \rho_{n,k} \cdot \delta\{\pi; \mathcal{B}(V_k, \eta_n)\} + \left(1 - \sum_{k=1}^K \rho_{n,k}\right) \cdot g_n(\pi),$$

where $\rho_{n,k}$ are positive constants such that $\min_{1\le k\le K}\rho_{n,k} > 0$, $\sum_{k=1}^{K}\rho_{n,k} < 1$ and $\max\rho_{n,k}^{-1} = o(n)$, $\eta_n = o(\max_{k\neq l}\|V_k^* - V_l^*\|)$, $\delta\{\pi;\mathcal{B}(V_k,\eta_n)\}$ is the density function of the uniform distribution over $\mathcal{B}(V_k,\eta_n) = \{x : \|V_k - x\| \le \eta_n\}$ for $k = 1, 2, \ldots, K$, and $g_n(\pi)$ is some probability density function over $\mathcal{S}_0/\cup_{k=1}^{K}\mathcal{B}(V_k,\eta_n)$.

Theorem 4. (Stochastic efficiency under misspecification, Setting 2). Under conditions of Theorem 3 except that Assumption 3 is replaced by Assumption 5, for any $\epsilon > 0$, there exists a positive constant C_0 satisfying $\log^{-1} C_0 = o(\max_k \rho_{n,k})$ such that

$$\mathbb{P}\left(\max_k \|\tilde{V}_k - V_k^*\|^2 \le C_0 \sum_{i=1}^{n} \|X_i - X_i^*\|^2/n\right) \ge 1 - \epsilon.$$

Remark 7. Theorems 3 and 4 provide stronger conclusions than stochastic efficiency that the maximum error bound of the estimated vertex can be chosen as the root mean squared error of observed data, say $n^{-1}\sum_{i=1}^{n}\|Z_i\|^2$. In fact, the large deviation bound of the ℓ^2-error is at the order of the maximum error bound divided by $\sqrt{\log(n)}$. Thus if considering the ℓ^2-error, we could obtain a sharper bound and faster convergence rate.

Remark 8. Theorems 3 and 4 both consider the most difficult case when $\mathcal{S}^* \subset \tilde{\mathcal{S}}$. It can be seen that our proposed approach can still shrinkage the estimated simplex towards the interior of the data cloud in this case. Since the error is $O\{\sigma\sqrt{\log(n)}\}$, the estimated simplex obtained by all the other existing methods could explode when we use the wrong distribution of α. Theorems 3 and 4 show our proposed algorithm could control the error bound at least as good as SP and AA even when the model is misspecified. Thus our proposed method has considerably nice performances against model misspecification.

4. Simulations

To assess the numerical performance of VALISE, we study the behavior of the algorithm when n and K grows. We also compare VALISE with other vertex hunting methods: sketched vertex search (SVS) (Jin et al., 2023), archetypal analysis (AA) (Cutler and Breiman, 1994), successive projection (SP) (Araújo et al., 2001) and K-means. We compare these methods under the setting when α is given, α is jointly optimized with (V, σ^2) and when the model is misspecified. For all the experiments below, we denote *VALISE* the version with given α and *VALISE+* the version that jointly optimizes (V, σ^2, α).

To implement SVS and AA, we use the R packages *ScorePlus* and *archetypes* (Eugster and Leisch, 2009) with the default algorithm parameters. For each parameter setting, the π_is are *iid* drawn from Dirichlet(α), the simulated data points X_i^*s are *iid* drawn from $\mathcal{N}_r\big(\sum_{k=1}^K \pi_i(k)V_k, \sigma^2 I_r\big)$, we report the estimation error $\min_\tau \max_{1\le k\le K} \|\hat{V}_k - V_{\tau(k)}\|$ for 50 repetitions.

4.1 Comparison with other methods

In this experiment, we compare the performance of VALISE and VALISE+ with SVS, AA, SP and K-means.

First we consider VALISE, where α is given. Consider the simple case when $K = 3, r = 2$. Let $n = 500$, fix V, $(\alpha', \sigma^2) = (\mathbf{1}_K', 1)$. We simulate n data points according to this setting, and run VALISE with given $\alpha = \mathbf{1}_K$, set $\eta = 0.02$ and let the initial parameters V_0 be the result of SVS and $\sigma_0^2 = 1$. The results are presented in Figure 2.

Statistica Sinica: Preprint
doi:10.5705/ss.202023.0159

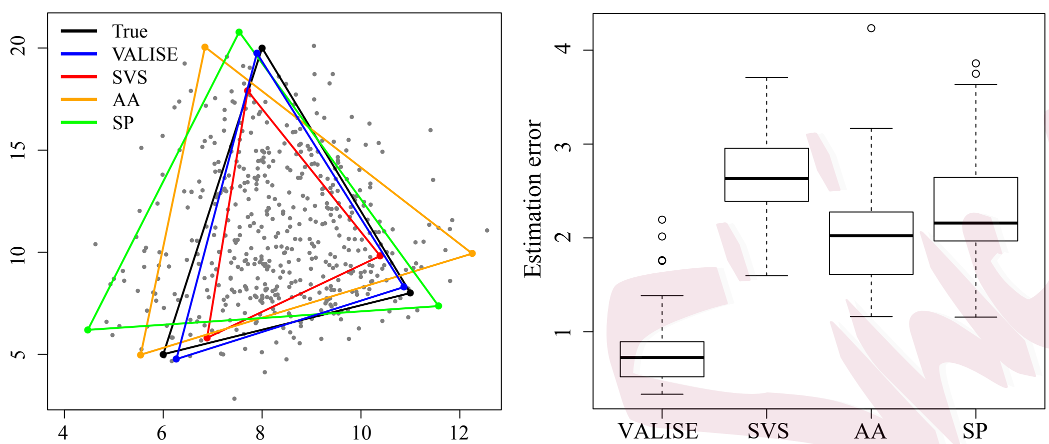

Figure 2: Comparison of VALISE (given $\alpha = \mathbf{1}_K$) to other methods. Left: true vertices (black), VALISE (blue) and other methods. Right: boxplot of estimation errors for different methods. Results are based on 50 repetitions.

It suggests that VALISE outperforms other methods in terms of the estimation error. VALISE is initialized by SP, where all three vertices are inside of the true simplex, then they gradually move outside towards the true vertices during the iterations, thus achieving a better performance in estimating the true vertices.

Next, we consider VALISE+, which is the case when α is jointly optimized with (V, σ^2). Let $(K, r) = (3, 2)$, fix V, $\sigma^2 = 1$ and $\alpha = (0.5, 0.6, 0.7)'$, simulate $n = 1000$ data points according to this setting and run VALISE+ that optimizes (V, σ^2, α) together. Let the initial vertices V_0 be SVS, $\sigma_0^2 = 1$ and $\eta = 0.02$. The results are shown in Figure 3. We see that when the data points are not uniformly distributed within the simplex, VALISE+ still gives a better estimation of the true vertices than other methods and can report an estimated α.

To further measure the performance of VALISE+ under different distributions of the data points, we conduct simulation settings with various α. Let $(K, r) = (3, 2)$, fix

Statistica Sinica: Preprint
doi:10.5705/ss.202023.0159

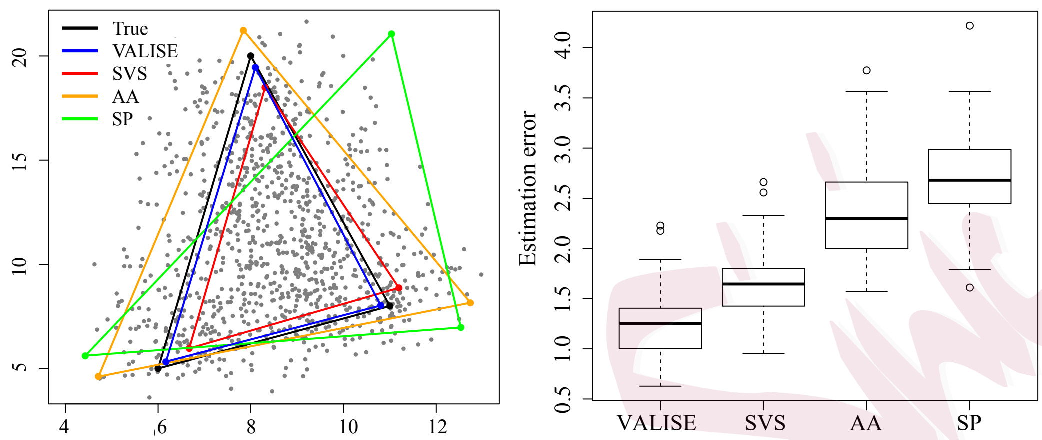

Figure 3: Comparison of VALISE+ (true $\alpha = (0.5, 0.6, 0.7)'$) to other methods. Left: true vertices (black), VALISE+ (blue) and other methods. Right: boxplot of estimation errors for different methods. Results are based on 50 repetitions.

V, $\sigma^2 = 0.1$ and $n = 1000$, let α takes on value of $(0.1, 0.2, 0.3)'$, $(0.5, 1, 1.5)'$ and $(1, 2, 3)'$ respectively. For each setting, we run VALISE+ that jointly optimizes (V, σ^2, α). Set $\eta = 0.06$, let the initial parameters V_0 be SVS for $\alpha = (0.1, 0.2, 0.3)'$ and SP for the others; let $\sigma_0^2 = 0.1$. The estimation errors of VALISE+ and other methods are displayed in Figure 4.

It suggests that VALISE+ outperforms other methods in all of the three settings. We find that the performance of VALISE+ improves as α decreases; the reason is that, when α is small ($\alpha_i < 1$), the simulated data points tend to cluster around the vertices of the simplex, making it easy for VALISE+ to identify the different vertices, we could even have a rough guess by eyes; however, when α is large, the simulated data points all cluster around the center of the simplex, thus the performance of VALISE+ deteriorates.

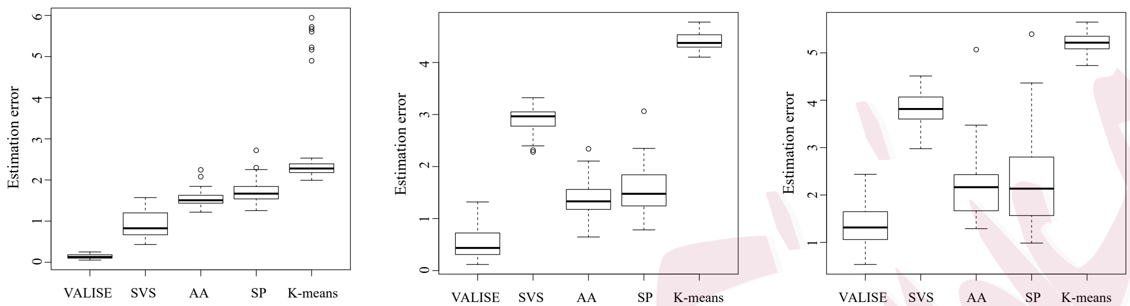

Figure 4: Estimation error of VALISE+ and other methods. The data is generated with three different α's. Left: $\alpha = (0.1, 0.2, 0.3)'$. Middle: $\alpha = (0.5, 1, 1.5)'$. Right: $\alpha = (1, 2, 3)'$. Results are based on 50 repetitions.

4.2 VALISE under different settings

In this section, we study the behavior of VALISE when n, K, σ^2 and α change. We design 9 different settings to see how the estimation error changes when one of the parameters in (n, K, σ^2, α) changes. Let Setting 1: $(n, K, \sigma^2, \alpha) = (1000, 3, 1, \mathbf{1}_K)$ be the baseline. In Setting 2 and 3, we let n to be smaller and larger. In Setting 4 and 5, we let K grow, and the dimensionality $r = K - 1$ grows accordingly. In Setting 6 and 7, we choose smaller and larger σ^2. In Setting 8 and 9, we generate data with smaller and larger α and run VALISE given the same α. For all of the settings above, let $\eta = 0.06$, and the initial parameters V_0 be the result of SP, $\sigma_0^2 = 1$. The results are presented in Table 1. It suggests that fixing the rest of the parameters, as n grows, the estimation error decreases; as K grows, the estimation error increases; smaller σ^2 and α in the data generating process give a better vertex estimation for VALISE, this is because when the data points tend to cluster around the vertices of the simplex that they are generated from, and they don't have too much variation, then it's easier for

Table 1: Estimation error of VALISE under different settings. Results are averaged over 50 repetitions.

Setting	n	K	σ^2	α	Error
1	1000	3	1	$\mathbf{1}_K$	0.4399231
2	**500**	3	1	$\mathbf{1}_K$	0.5461461
3	**3000**	3	1	$\mathbf{1}_K$	0.3237662
4	1000	**4**	1	$\mathbf{1}_K$	2.3565339
5	1000	**5**	1	$\mathbf{1}_K$	4.0654453
6	1000	3	**0.1**	$\mathbf{1}_K$	0.2800923
7	1000	3	**2**	$\mathbf{1}_K$	0.5030107
8	1000	3	1	$\mathbf{0.1} \cdot \mathbf{1}_K$	0.1523620
9	1000	3	1	$\mathbf{2} \cdot \mathbf{1}_K$	0.9163023

VALISE to correctly identify the vertices.

To further study the behavior of VALISE when n grows. Fix V, let $(K, r) = (3, 2)$, $(\alpha, \sigma^2) = (\mathbf{1}_K, 1)$ and let n range in $\{500, 1000, 1500, 2000, 3000, 5000\}$. For each setting, we run VALISE given $\alpha = \mathbf{1}_K$, let $\eta = 0.06$ and let the initial parameters V_0 be the result of SVS and $\sigma_0^2 = 1$. The result is presented in left panel of Figure 5. It suggests that as n increases, the estimation of VALISE is more accurate, and VALISE can have a perfect performance when n is large enough.

To further study the behavior of VALISE when K grows. Let $n = 1000$, $(\alpha, \sigma^2) = (\mathbf{1}_K, 1)$ and let K range in $\{3, 4, 5, 6, 7, 8\}$, fix V for a given K. For each setting, we run VALISE with given $\alpha = \mathbf{1}_K$, set $\eta = 0.02$ and let the initial parameters V_0 be the result of SP and $\sigma_0^2 = 1$, we also run archetypal analysis (AA) and successive projection (SP) for comparison. The result is displayed in right panel of Figure 5, where

Statistica Sinica: Preprint
doi:10.5705/ss.202023.0159

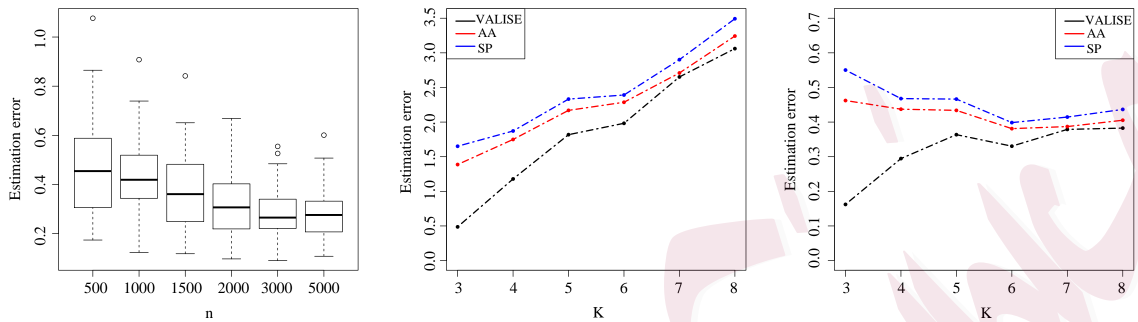

Figure 5: Estimation error of VALISE as n grows (left) and K grows (middle and right). Results are based on 50 repetitions.

the estimation error is scaled by $\frac{1}{\sqrt{K}}$ (middle) and $\frac{1}{K\sqrt{K}}$ (right). It shows that VALISE performs reasonably well when K is large, and it outperforms AA and SP in all settings, especially when K is small.

Finally, we numerically demonstrate the robustness of VALISE to outliers. We first generate clean data with $n = 1000, K = 3, \sigma^2 = 1, \alpha = \mathbf{1}_K$, then we construct three contaminated scenarios with different percentage and magnitude of outliers: (1) 0.1% of samples is generated with $\sigma^2 = 100$; (2) 1% of samples is generated with $\sigma^2 = 10$; and (3) 1% of samples is generated with $\sigma^2 = 20$. We run VALISE on the clean data and three contaminated data. Comparison of estimation errors is presented in Table 2. It shows that VALISE is robust under different portions and magnitudes of outliers.

4.3 Misspecified model

In this experiment, we compare the performance of VALISE with other methods when the model is misspecified. We aims to find the situation when VALISE still outperforms other methods under a misspecified setting. Let $(K, r) = (3, 2)$, fix V, $\sigma^2 = 0.1$ and

Statistica Sinica: Preprint
doi:10.5705/ss.202023.0159

Table 2: Estimation error of VALISE under contaminated settings. Results are based on averages over 50 repetitions.

Setting	n	K	σ^2	α	Error
Clean	1000	3	1	$\mathbf{1}_K$	0.4399231
Contaminate(1)	1000	3	1	$\mathbf{1}_K$	0.5318753
Contaminate(2)	1000	3	1	$\mathbf{1}_K$	0.7238348
Contaminate(3)	1000	3	1	$\mathbf{1}_K$	1.1277980

$n = 1000$, let α takes on value of $(0.6, 0.8, 1)'$, $(0.8, 1, 1.2)'$ and $(1, 1.2, 1.4)'$ respectively. For each setting, we generate simulated data and run VALISE with given $\alpha = \mathbf{1}_K$. Set $\eta = 0.06$, let the initial parameters V_0 be SP and $\sigma_0^2 = 0.1$. We also run VALISE+ with $\eta = 0.06, \sigma_0^2 = 0.1$ and the initial vertices be SP for comparison. The estimation errors are displayed in Figure 6.

Note that VALISE with given α is based on a misspecified model as the given $\alpha = \mathbf{1}_K$ is different from that in the data generating process. The results reveal that VALISE with given α outperforms other methods in all three misspecified settings above. We notice that under the misspecified setting, VALISE+ always achieves better performance than VALISE with given α; the reason is obvious, VALISE+ is actually based on a well-specified model. We also observe as the given α in VALISE be further from the true value in the data generating process, the performance deteriorates; when α is far enough, the performance can be worse than other methods. Remove this sentence: Thus, the given α in VALISE should be carefully chosen before we run the algorithm.

Remark 9 (Computational cost)**.** To compare the computational cost of VALISE to other methods, we present runtime of VALISE under different settings in Table 3, all let

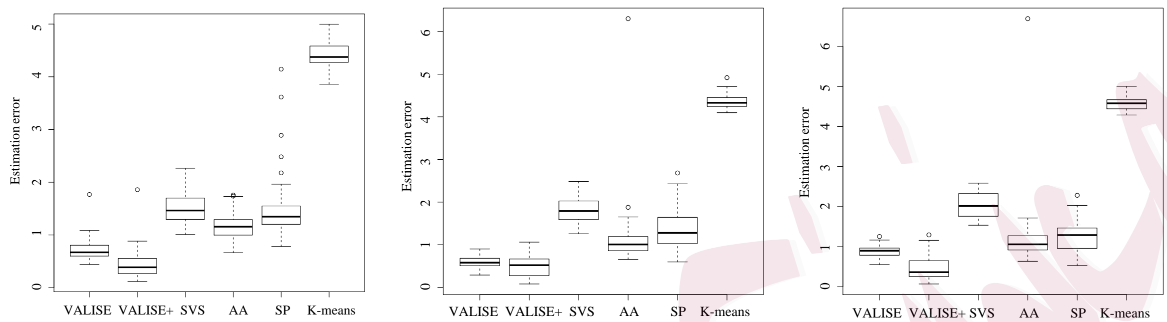

Figure 6: Misspecified case: estimation error of VALISE (given $\alpha = \mathbf{1}_K$), VALISE+ and other methods. The data is generated with three different α's. Left: $\alpha = (0.6, 0.8, 1)'$. Middle: $\alpha = (0.8, 1, 1.2)'$. Right: $\alpha = (1, 1.2, 1.4)'$. Results are based on 50 repetitions.

the initial parameters V_0 be the result of SVS. Table 3 shows that runtime of VALISE is moderate under large n and K. Compared with other vertex hunting methods, VALISE has comparable runtime with nonnegative matrix factorization (NMF) (Javadi and Montanari, 2019) as they both require iterative updates. Runtime for SVS, AA, SP and K-means is within a few seconds as they don't rely on iterative updates.

Table 3: Runtime of VALISE under different settings. Results are based on averages over 50 repetitions.

Setting	n	K	σ^2	α	Runtime(sec)
1	500	3	1	$\mathbf{1}_K$	15.4
2	1000	3	1	$\mathbf{1}_K$	20.2
3	2000	3	1	$\mathbf{1}_K$	48.0
4	1000	4	1	$\mathbf{1}_K$	23.6
5	1000	5	1	$\mathbf{1}_K$	30.9

Statistica Sinica: Preprint
doi:10.5705/ss.202023.0159

Remark 10 (Initial values of VALISE)**.** A good initial value would reduce the convergence time of VALISE. We provide a rule of thumb to select initial values of V, σ^2 and α. For initial values of V, we recommend using the result of sketched vertex search (SVS) (Jin et al., 2023) or archetypal analysis (AA) (Cutler and Breiman, 1994). Section 4 shows that SVS and AA generate moderately good estimation with less runtime. For initial values of σ^2, we recommend taking $\sigma_0 = \max_{1\leq i\leq n} \|X_i - \bar{X}\| \times 0.01$. For initial values of α, we recommend first running VALISE+ to obtain an estimate of α, denoted as $\hat{\alpha}$, then run VALISE with the initial value $\hat{\alpha}$.

5. Real Data Applications

We evaluate the performance of our algorithm on a wide range of real-world applications.

5.1 Citee network

In Ji and Jin (2016)'s paper, they have collected a network data set for statisticians, based on all published papers in Annals of Statistics, Biometrika, JASA, and JRSS-B, from 2003 to the first half of 2012. The data set allows us to construct many networks. We focus our study on a citee network, where each node is an author, and there is an edge between two authors if they have been cited at least once by the same author (other than themselves). We focus on the giant component ($n = 1790$) for our study.

Ji and Jin (2016) suggested that the network has three meaningful communities: 'Large Scale Multiple Testing'(MulTest), 'Spatial and Nonparametric Statistics' (SpatNon) and 'Variable Selection' (VarSelect). In light of this, we use a DCMM model with K $= 3$, and apply the SCORE method to get the data matrix $R \in \mathbb{R}^{n\times 2}$. We aim to identify three vertices among the data cloud formed by rows of R. Thus we run VALISE given $\alpha = (0.75, 0.3, 0.75)'$, let $\eta = 0.01, \sigma_0^2 = 0.7$ and let the initial parameters V_0 be the result of SVS. Initial α value $(0.75, 0.3, 0.75)'$ is the estimated α from VALISE+. The estimated $\hat{\sigma}^2$ of VALISE is 0.67. Figure 7 presents the rows of R, where a 2-simplex (i.e., triangle) identified by VALISE and other methods are clearly visible in the cloud. It suggests that comparing with other methods, VALISE more precisely estimate the underlying vertices of the data.

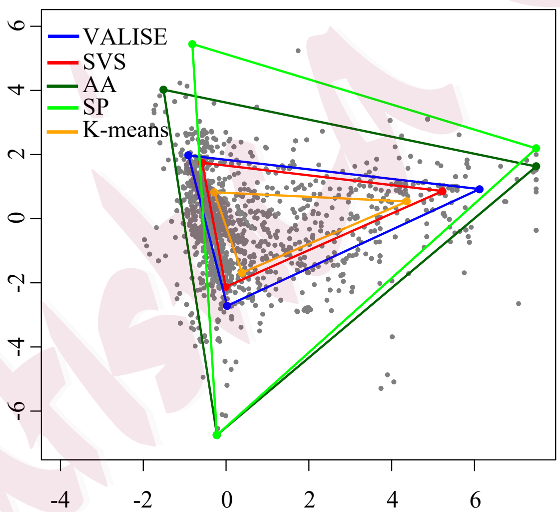

Figure 7: Citee network: Comparison of VALISE (blue) to other methods.

5.2 Topic model

Vertex hunting is a crucial step in topic model estimation. We consider the data set consisting of the abstract of 56500 papers published in 36 statistical journals, from

Statistica Sinica: Preprint
doi:10.5705/ss.202023.0159

1990 to 2015. After text stemming, removing stop words and low-frequency words, the vocabulary contains $n = 2106$ words.

We first properly scale the word-document matrix and obtain the corresponding first K left singular vectors $\hat{\Xi} \in \mathbb{R}^{n\times K}$. In order to recover the word-topic matrix, it is desirable to further normalize $\hat{\Xi}$ to a entry-wise ratio matrix $\hat{R} \in \mathbb{R}^{n\times(K-1)}$, where rows of $\hat{R}$ generate a point cloud with the silhouette of a simplex. We aim to use rows of $\hat{R}$ and the simplex structure to locate all vertices $v_1, v_2, \ldots, v_K$.

We find that $K = 11$ yields the most meaningful results, so 11 topics are picked out in our study. We run VALISE given $\alpha = (0.58, 0.24, 0.61, 1.35, 0.50, 0.68, 0.68, 1.47, 0.86, 0.94, 1.51)'$, this α value is the estimated α from VALISE+. Let $\eta = 0.01$, $\sigma_0^2 = 0.3$ and V_0 be the result of SVS. Black dots in Figure 8 shows the projection of the rows of $\hat{R} \in \mathbb{R}^{n\times 10}$ into the first four principal components, and the estimated 11 vertices marked in different colors. We can see that the estimated vertices roughly capture all the corners of the data cloud in the projection plots, suggesting satisfactory performance of VALISE in identifying the vertices. Table 4 shows the top 10 representative words in each of the 11 estimated topics. These topics can be interpreted as 'Inference', 'Exp.Design', Time Series', 'Machi.Learn', 'Bayes', 'Latent.Var', 'Clinic', 'Math.Stats', 'Regression', 'Hypo.Test' and 'Bio/Med'.

5.3 Single cell RNA-seq data

The single cell RNA-sequence data set (Deng et al., 2014) contains RNA-sequence read counts for single cells at different stages of mouse embryo development, from zygote to blastocyst. The data set consists of $n = 259$ stages. A single cell at certain stage belongs

Statistica Sinica: Preprint
doi:10.5705/ss.202023.0159

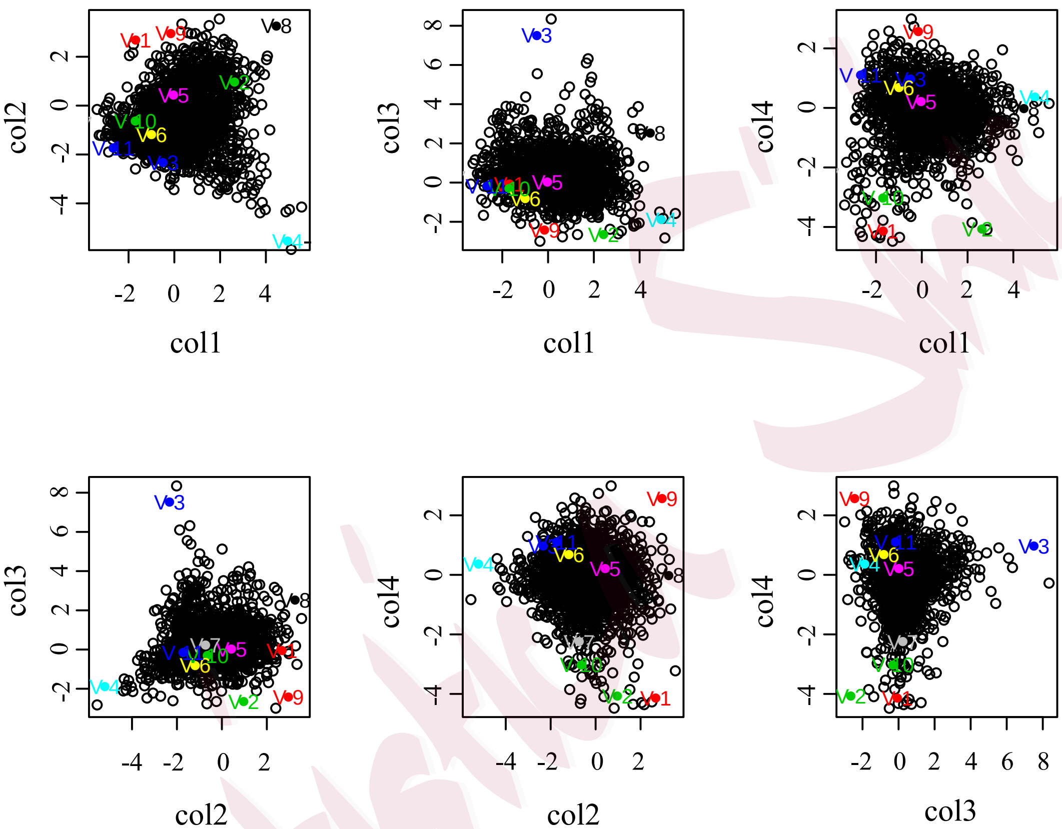

Figure 8: Topic model: Projection of the rows of $\hat{R}$ into the first four principal components with the estimated 11 vertices marked in different colors.

to a mixture of cell types with different probabilities. We aim to identify K different cell types across different stages of mouse development and estimate the membership probability for this cell at each stage.

We applied our methods with $K = 3, 6$. For $K = 3$, we run VALISE given $\alpha = (0.07, 0.04, 0.11)'$, this α value is the estimated α from VALISE+. Let $\eta = 0.1$, $\sigma = 1000$ and the result from K-means as V_0. For $K = 6$, we run VALISE given $\alpha = \mathbf{1}_K$, $\eta = 0.5$,

Table 4: Topic Model: top 10 representative words for each estimated topic ($K = 11$).

'Inference'	*confid, coverag, width, interv, twosid, pivot, bootstrap, stepdown, onesid, edgeworth*
'Exp.Design'	*aoptim, doptim, latin, aberr, factori, twofactor, twolevel, design, nonregular, block*
'Time Series'	*wait, time, semimarkov, seri, hit, failur, queue, intervalcensor, event, repair*
'Machi.Learn'	*kmean, metropoli, algorithm, learn, scalabl, stateoftheart, svm, supervis, machin, text*
'Bayes'	*jeffrey, improp, nuisanc, prior, conjug, paramet, noninform, posterior, default, hyperparamet*
'Latent.Var'	*variabl, manifest, explanatori, categor, select, latent, forest, proxi, ordin, exogen*
'Clinic'	*placebo, treatment, noncompli, complianc, clinician, trial, arm, therapi, clinic, therapeut*
'Math.Stats'	*probab, corollari, levi, walk, ergod, ddimension, theorem, infin, epsilon, convolut*
'Regression'	*ridg, regress, regressor, cook, singleindex, backfit, varyingcoeffici, spline, smoother, isoton*
'Hypo.Test'	*fals, regulatori, discoveri, bonferroni, decisionmak, fdr, familywis, agenc, prespecifi, control*
'Bio/Med'	*casecontrol, genomewid, populationbas, polymorph, alzheim, epidemiolog, phenotyp, exposur, genotyp, ascertain*

$\sigma = 1000$ and the result from K-means as V_0.

The results are shown in Figure 9, the first row for $K = 3$ and the second row for $K = 6$. For each setting, the left panel is the structure bar plot indicating cell-type membership for each development stage. Each row represents a stage containing bars of different colors, each color represents a cell type, the lengths of the bar measure the probability of the stage belonging to different cell types. All lengths sum up to 1. In the right panel, we visualize the clustering result of the data in the first two principal components. For both setting, we see that VALISE gives reasonable estimation of the vertices in the sense that the resulting clustering of the development stages roughly follows the chronological order.

6. Discussion

This paper introduces a new vertex hunting algorithm based on the maximum likelihood estimation. In the real data application, we make two approximations: (i) Clustering

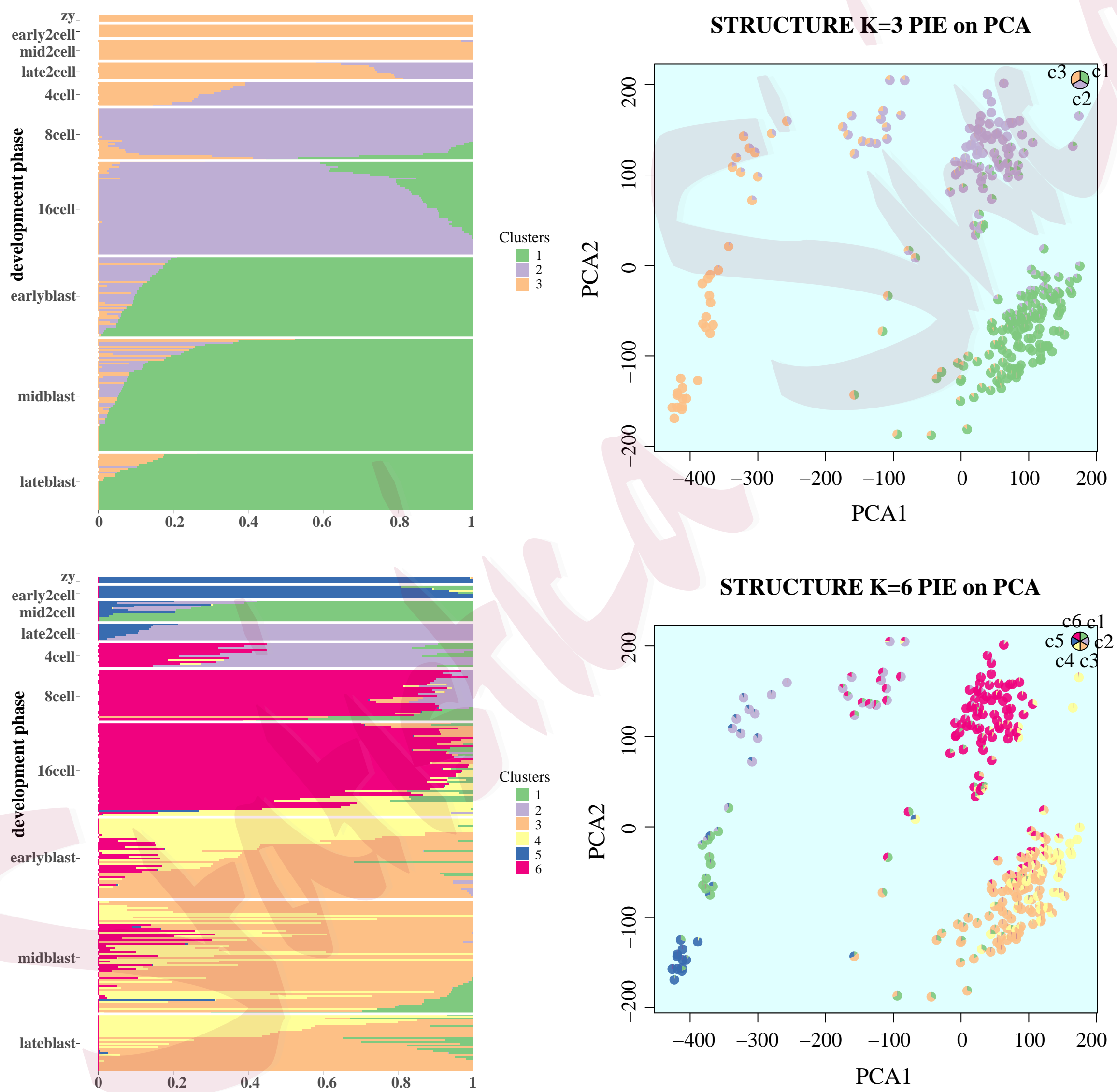

Figure 9: RNA-Seq: plots of cell type membership (left) and PCA projection (right) for $K = 3$ (first row) and $K = 6$ (second row).

membership follows a Dirichlet distribution $\pi \sim \text{Dir}(\alpha)$. This is a common practice and has been widely used when we model a discrete probability density; (ii) Noises are independently and identically generated from a normal distribution $Z_i \overset{i.i.d}{\sim} N(0, \sigma^2 I)$. As the nodes are usually correlated with each other (rows of $\hat{R}$ are not independent), we are actually maximizing a quasi-likelihood function where the model is mis-specified. However, our algorithm does offer an appropriate way to shrink parameters and it's completely tuning free.

We could further improve our model by assuming $Z_i \sim N(0, \sigma^2 w_i I)$, where w_i represent the weight of node i and is a function of the degree of this node. As a node with higher degree obtains less noise in our setting, we could give this node a smaller weight and thus a smaller variance before run our algorithm. By doing this, our model accommodates more information and explains more variance among the data, thus leads to a better model performance.

Supplementary Materials

Supplementary Materials are available in the attached file which contains a summary of major notations, useful lemmas, proofs of Proposition 1, Theorems 1 – 4 and Corollary 1.

Acknowledgements

The authors are grateful to the anonymous referees, the associate editor and the editor for their helpful comments and suggestions. Zhang's research is partially supported by National Natural Science Foundation of China Grant 92358303, National Key R&D

Program of China Grants 2021YFA1000100, 2021YFA1000101 and 2021YFA1000104, Science and Technology Commission of Shanghai Municipality Grant 23JS1400500, and National Natural Science Foundation of China Grants 72331005, 72571102 and 72201101. Ke's research is partially supported by Sloan Research Grant FG-2023-19970.

References

Airoldi, E. M., D. M. Blei, S. E. Fienberg, and E. P. Xing (2008). Mixed membership stochastic blockmodels. *Journal of Machine Learning Research 9*, 1981–2014.

Araújo, M. C. U., T. C. B. Saldanha, R. K. H. Galvao, T. Yoneyama, H. C. Chame, and V. Visani (2001). The successive projections algorithm for variable selection in spectroscopic multicomponent analysis. *Chemometrics and Intelligent Laboratory Systems 57*(2), 65–73.

Arora, S., R. Ge, and A. Moitra (2012). Learning topic models – going beyond SVD. In *2012 IEEE 53rd Annual Symposium on Foundations of Computer Science*, pp. 1–10.

Bioucas-Dias, J. M., A. Plaza, N. Dobigeon, M. Parente, Q. Du, P. Gader, and J. Chanussot (2012). Hyperspectral unmixing overview: Geometrical, statistical, and sparse regression-based approaches. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing 5*(2), 354–379.

Blei, D. M., A. Y. Ng, and M. I. Jordan (2003). Latent dirichlet allocation. *Journal of Machine Learning Research 3*, 993–1022.

Craig, M. D. (1994). Minimum-volume transforms for remotely sensed data. *IEEE Transactions on Geoscience and Remote Sensing 32*(3), 542–552.

Cutler, A. and L. Breiman (1994). Archetypal analysis. *Technometrics 36*(4), 338–347.

Deng, Q., D. Ramsköld, B. Reinius, and R. Sandberg (2014). Single-cell rna-seq reveals dynamic, random monoallelic gene expression in mammalian cells. *Science 343*(6167), 193–196.

Dozat, T. (2016). Incorporating nesterov momentum into Adam. In *Proceedings of the 4th International Conference on Learning Representations*, pp. 1–4.

Eugster, M. J. A. and F. Leisch (2009). From spider-man to hero — archetypal analysis in R. *Journal of Statistical Software 30*(8), 1–23.

Javadi, H. and A. Montanari (2019). Nonnegative matrix factorization via archetypal analysis. *Journal of the American Statistical Association*, 1–22.

Ji, P. and J. Jin (2016). Coauthorship and citation networks for statisticians (with discussion). *The Annals of Applied Statistics 10*, 1779–1812.

Jin, J., Z. T. Ke, and S. Luo (2023). Mixed membership estimation for social networks. *Journal of Econometrics*.

Ke, Z. T. and M. Wang (2022). Using svd for topic modeling. *Journal of the American Statistical Association*, 1–16.

Van Dijk, D., R. Sharma, J. Nainys, K. Yim, P. Kathail, A. J. Carr, C. Burdziak, K. R. Moon, C. L. Chaffer, D. Pattabiraman, et al. (2018). Recovering gene interactions from single-cell data using data diffusion. *Cell 174*(3), 716–729.

Winter, M. E. (1999). N-FINDR: An algorithm for fast autonomous spectral end-member determination in hyperspectral data. In *SPIE's International Symposium on Optical Science, Engineering, and Instrumentation*, pp. 266–275.

Harvard University

E-mail: dieyi.chen@g.harvard.edu

Harvard University

E-mail: zke@fas.harvard.edu

East China Normal University

E-mail: syzhang@fem.ecnu.edu.cn