arXiv:2601.01757v1 [stat.ML] 5 Jan 2026

稀疏凸双聚类

Jiakun Jiang、Dewei Xiang、Chenliang Gu，北京师范大学文理学院，中国珠海；Wei Liu，四川大学数学学院，中国成都；Binhuan Wang*，艾伯维公司数据与统计科学部门，美国新泽西州弗洛勒姆帕克

摘要

双聚类是一种重要的无监督机器学习技术，可同时对数据矩阵的行和列进行聚类，广泛应用于基因组学、转录组学及其他高维组学数据领域。尽管双聚类十分重要，现有方法仍难以满足现代大规模数据集的需求。其挑战源于高维特征中的噪声累积、非凸优化形式的局限性，以及识别有意义簇的计算复杂度。随着数据集规模增大，这些问题往往会导致准确率和稳定性下降。为应对这些挑战，我们提出稀疏凸双聚类（SpaCoBi），这是一种在双聚类过程中对噪声进行惩罚、以提高准确率和稳健性的新方法。通过采用凸优化框架并引入基于稳定性的调参准则，SpaCoBi 在聚类保真度与稀疏性之间实现了最佳平衡。全面的数值研究（包括模拟以及对小鼠嗅球数据的应用）表明，SpaCoBi 的准确率显著优于最先进的方法。这些结果显示，SpaCoBi 是一种稳健且高效的解决方案，适用于高维和大规模数据集的双聚类。

关键词： 凸双聚类，基因表达数据，高维性，稀疏性，*Sylvester*方程

1 引言

在数据驱动研究快速发展的背景下，双聚类已成为分析复杂数据矩阵的一项关键技术，能够同时对行和列进行聚类。

*通讯作者：wang.binhuan@gmail.com

这种双重划分能力使双聚类有别于传统聚类方法（通常称为单向聚类）：后者通常根据所有可用特征对观测进行聚类，或根据所有观测对特征进行聚类。双聚类能够揭示观测和特征呈现同步模式的子矩阵，从而洞察上下文特异性关系；若采用全局分析，这些关系往往会被掩盖。这一特点在众多应用领域中尤为有用，尤其是在生物和生物医学数据领域（Xie et al., 2019）。这些应用涉及由单细胞RNA测序等技术生成的复杂数据集，能够深入揭示细胞异质性、疾病相关变异识别以及调控程序推断。例如，在基因表达研究中，某些基因子集可能仅在特定细胞类型或实验条件下表现出共表达。双聚类能够高效地揭示这些模式，因此相比传统方法可提供更精确的生物学解释（Busygin et al., 2008; Madeira and Oliveira, 2004）。

在我们的示例中，我们分析了通过 10x Chromium 单细胞RNA测序获得的小鼠嗅球（MOB）数据。该数据集包含 305 个观测，每个观测包含 1,250 个基因表达值。这项案例研究至关重要，因为了解细胞类型的异质性并识别标记基因，是阐明这一神经结构功能组织的重要步骤，而该结构在处理嗅觉信息方面发挥着基础作用。

尽管传统双聚类方法具有实用价值，但仍面临诸多挑战，尤其是在现代大规模科学研究中常见的高维数据集上应用时。早期方法通常基于混合模型或经典算法，并对数据结构作出特定假设，因此灵活性不足，也难以适用于真实世界的数据。基于奇异值分解（SVD）的方法（Bergmann 等人，2003年；Lazzeroni 和 Owen，2002年；Turner 等人，2005年）以及采用基于图的划分策略的方法颇具代表性。然而，这些方法往往依赖贪心优化算法，因此常常只能达到

局部最优解，从而限制了它们在复杂数据集上的效能（Chi 等人，2017年；Wang 等人，2023年）。高维数据的出现进一步加剧了这些挑战，因为传统的非凸优化形式难以满足现代数据集带来的计算和分析需求。近期方法开始通过在双聚类过程中引入稀疏性来应对这些问题。基于稀疏奇异值分解（SVD）的技术（Chen 等人，2013年；Lee 等人，2010年；Sill 等人，2011年）尝试通过对奇异值施加惩罚来改善结果，从而实现更好的特征选择。然而，由于其准则函数具有非凸性，这些方法通常难以解释，也无法保证达到全局最优解（Helgeson 等人，2020年）。

针对这些挑战，我们提出了稀疏凸双聚类（SpaCoBi）——一种创新框架，将稀疏性融入凸优化方法，从而有效缓解高维噪声累积并实现精准的特征选择。受稀疏凸聚类算法（Wang 等人，2018年）的启发，该算法在凸优化框架下同时对观测值进行聚类并执行特征选择，并保证获得全局最优值；所提出的 SpaCoBi算法在其双聚类模型中引入了促稀疏的套索惩罚，通过抑制无关特征来增强真实信号的检测能力。该计算策略将问题分解为易处理的子问题，并通过融入 Sylvester型更新的伪回归方案求解，同时严格保证收敛。这些创新贡献有针对性地应对了固有的高维挑战，在准确率和稳健性方面带来了显著提升。我们的方法具有多项优势：首先，在准确率和稳定性方面，SpaCoBi 的凸形式允许存在唯一的全局最小化解，从而显著提升不同维度下的聚类精度。其次，SpaCoBi 通过同时估计双聚类组并选择信息特征，提高了可解释性，从而揭示细胞亚群及其特征基因集等生物学信息。最后，通过采用快速 Sylvester求解器和热启动的简洁迭代算法，实现了计算效率，确保

可扩展性。

本文其余部分的结构如下：第 2 节详细介绍 SpaCoBi 框架及其优化算法。第 3 节讨论实际实现方面的考量。第 4 节通过大量模拟以及一项基于 MOB 数据的案例研究，展示该方法的卓越性能。第 5 节总结全文并讨论未来研究方向。技术细节见附录。

2 稀疏凸双聚类

2.1 模型

设 $\mathbf{X} \in \mathbb{R}^{n\times p}$ 为一个数据矩阵，包含 n 个观测 $X_{i\cdot} = (X_{i1}, X_{i2}, \ldots, X_{ip})^{\mathrm{T}}$ 和 p 个特征 $i = 1, \cdots, n$。我们假设这些 n 观测属于 K 个未知且互不重叠的类别 $C_1, \ldots, C_K$，而这些 p 特征属于 R 个未知且互不重叠的类别 $D_1, \ldots, D_R$。为便于后续推导，也可以在特征层面将数据矩阵 $\mathbf{X}$ 表示为列向量 $\mathbf{X} = (\mathbf{x}1, \cdots, \mathbf{x}p)$，其中 $\mathbf{x}j = (X_{1j}, \cdots, X_{nj})^{\mathrm{T}}$, $j = 1, \ldots, p$。类似地，我们将 $\mathbf{A}$ 在特征层面表示为列向量 $\mathbf{A} = (\mathbf{a}1, \cdots, \mathbf{a}p)$，在观测层面表示为 $(A_{1\cdot}, \ldots, A_{n\cdot})^{\mathrm{T}}$。定义 $\mathcal{E}_1 = \{l = (l_1, l_2) : 1 \le l_1 < l_2 \le n\}$ 以及 $\mathcal{E}_2 = \{k = (k_1, k_2) : 0 \le k_1 < k_2 \le p\}$。然后分别将 $|\mathcal{E}_1|$ 和 $|\mathcal{E}_2|$ 记为 $\mathcal{E}_1$ 和 $\mathcal{E}_2$的分量数。我们将稀疏凸双聚类问题表述如下，

$$
\begin{aligned}
\min_{\mathbf{A}\in\mathbb{R}^{n\times p}} \quad & \frac{1}{2}\sum_{i=1}^{n} \|X_{i\cdot} - A_{i\cdot}\|_2^2 \\
\text{s.t.} \quad & \sum_{l\in\mathcal{E}_1} w_l \|A_{l_1\cdot} - A_{l_2\cdot}\|_q \le t \\
& \sum_{k\in\mathcal{E}_2} \widetilde{w}_k \|\mathbf{a}_{k_1} - \mathbf{a}_{k_2}\|_q \le s \\
& \sum_{j=1}^{p} u_j \|\mathbf{a}_j\|_2 \le r,
\end{aligned}
\tag{1}
$$

其中权重为 $w_l \geq 0$, $\widetilde{w}_k \geq 0$和 $u_j \geq 0$。约束条件 (1) 中的第一项和第二项分别旨在促进观测和特征的整合，以实现双聚类。第三项强调特征稀疏性的重要性。这里采用组LASSO或自适应组LASSO惩罚来选择特征，因为同一特征通常会被所有观测共享。通过引入额外的松弛变量$\mathbf{v}_l$、$\mathbf{z}_k$和$\mathbf{g}^j$（它们是应用 ADMM算法的关键组成部分），我们可以将上述问题改写为以下等价的约束优化问题,

$$\begin{aligned}
\min_{\mathbf{A}\in\mathbb{R}^{n\times p}} \quad & \frac{1}{2}\sum_{i=1}^{n}\|X_{i\cdot}-A_{i\cdot}\|_2^2+\gamma_1\sum_{l\in\mathcal{E}_1}w_l\|\mathbf{v}_l\|_q+\gamma_2\sum_{k\in\mathcal{E}_2}\widetilde{w}_k\|\mathbf{z}_k\|_q+\gamma_3\sum_{j=1}^{p}u_j\|\mathbf{g}_j\|_2 \qquad (2)\\
\text{s.t.} \quad & A_{l_1\cdot}-A_{l_2\cdot}-\mathbf{v}_l=\mathbf{0},\ \forall l\in\mathcal{E}_1\\
& \mathbf{a}_{k_1}-\mathbf{a}_{k_2}-\mathbf{z}_k=\mathbf{0},\ \forall k\in\mathcal{E}_2\\
& \mathbf{a}_j-\mathbf{g}_j=\mathbf{0},\ j=1,\ldots,p.
\end{aligned}$$

于是，增广拉格朗日问题可表示为

$$\begin{aligned}
& \mathcal{L}_{\nu_1,\nu_2,\nu_3}(\mathbf{A},\mathbf{v},\mathbf{z},\mathbf{g},\mathbf{\Lambda}_1,\mathbf{\Lambda}_2,\mathbf{\Lambda}_3)\\
= \ & \frac{1}{2}\sum_{i=1}^{n}\|X_{i\cdot}-A_{i\cdot}\|_2^2+\gamma_1\sum_{l\in\mathcal{E}_1}w_l\|\mathbf{v}_l\|_q+\gamma_2\sum_{k\in\mathcal{E}_2}\widetilde{w}_k\|\mathbf{z}_k\|_q+\gamma_3\sum_{j=1}^{p}u_j\|\mathbf{g}_j\|_2\\
& +\sum_{l\in\mathcal{E}_1}\langle\boldsymbol{\lambda}_{1l},\mathbf{v}_l-A_{l_1\cdot}+A_{l_2\cdot}\rangle+\frac{\nu_1}{2}\sum_{l\in\mathcal{E}_1}\|\mathbf{v}_l-A_{l_1\cdot}+A_{l_2\cdot}\|_2^2\\
& +\sum_{k\in\mathcal{E}_2}\langle\boldsymbol{\lambda}_{2k},\mathbf{z}_k-\mathbf{a}_{k_1}+\mathbf{a}_{k_2}\rangle+\frac{\nu_2}{2}\sum_{k\in\mathcal{E}_2}\|\mathbf{z}_k-\mathbf{a}_{k_1}+\mathbf{a}_{k_2}\|_2^2\\
& +\sum_{j=1}^{p}\langle\boldsymbol{\lambda}_{3j},\mathbf{g}_j-\mathbf{a}_j\rangle+\frac{\nu_3}{2}\sum_{j=1}^{p}\|\mathbf{g}_j-\mathbf{a}_j\|_2^2.
\end{aligned}$$

2.2 SpaCoBi算法

最小化上述增广拉格朗日问题 $\mathcal{L}_{\nu_1,\ \nu_2,\ \nu_3}$(**A**、**v**、**z**、**g** $\mathbf{\Lambda}_1$、 $\mathbf{\Lambda}_2$、 $\mathbf{\Lambda}_3$)颇具挑战，但ADMM算法使我们能够按照以下方案迭代更新**A**、**v**、**z**、**g** $\mathbf{\Lambda}_1$、 $\mathbf{\Lambda}_2$，以及 $\mathbf{\Lambda}_3$：

$$\mathbf{A}^{m+1} \ = \ \underset{\mathbf{A}}{\operatorname{argmin}}\,\mathcal{L}_{\nu_1,\nu_2,\nu_3}(\mathbf{A},\mathbf{V}^m,\mathbf{Z}^m,\mathbf{G}^m,\mathbf{\Lambda}_1^m,\mathbf{\Lambda}_2^m,\mathbf{\Lambda}_3^m),$$

$$
\begin{aligned}
\mathbf{V}^{m+1} &= \underset{\mathbf{V}}{\operatorname{argmin}}\, \mathcal{L}_{\nu_1,\nu_2,\nu_3}(\mathbf{A}^{m+1}, \mathbf{V}, \mathbf{Z}^m, \mathbf{G}^m, \boldsymbol{\Lambda}_1^m, \boldsymbol{\Lambda}_2^m, \boldsymbol{\Lambda}_3^m), \\
\mathbf{Z}^{m+1} &= \underset{\mathbf{Z}}{\operatorname{argmin}}\, \mathcal{L}_{\nu_1,\nu_2,\nu_3}(\mathbf{A}^{m+1}, \mathbf{V}^{m+1}, \mathbf{Z}, \mathbf{G}^m, \boldsymbol{\Lambda}_1^m, \boldsymbol{\Lambda}_2^m, \boldsymbol{\Lambda}_3^m), \\
\mathbf{G}^{m+1} &= \underset{\mathbf{G}}{\operatorname{argmin}}\, \mathcal{L}_{\nu_1,\nu_2,\nu_3}(\mathbf{A}^{m+1}, \mathbf{V}^{m+1}, \mathbf{Z}^{m+1}, \mathbf{G}, \boldsymbol{\Lambda}_1^m, \boldsymbol{\Lambda}_2^m, \boldsymbol{\Lambda}_3^m), \\
\boldsymbol{\lambda}_{1l}^{m+1} &= \boldsymbol{\lambda}_{1l}^m + \nu_1(\mathbf{v}_l^{m+1} - A_{l_1\cdot}^{m+1} + A_{l_2\cdot}^{m+1}),\ l \in \mathcal{E}_1, \\
\boldsymbol{\lambda}_{2k}^{m+1} &= \boldsymbol{\lambda}_{2k}^m + \nu_2(\mathbf{z}_k^{m+1} - \mathbf{a}_{k_1}^{m+1} + \mathbf{a}_{k_2}^{m+1}),\ k \in \mathcal{E}_2, \\
\boldsymbol{\lambda}_{3j}^{m+1} &= \boldsymbol{\lambda}_{3j}^m + \nu_3(\mathbf{g}_j^{m+1} - \mathbf{a}_j^{m+1}),\ j = 1, \ldots, p.
\end{aligned}
$$

接下来，我们分三个步骤详细介绍$\mathbf{A}$、$\mathbf{V}$、$\mathbf{Z}$、$\mathbf{G}$ $\boldsymbol{\Lambda}_1$、 $\boldsymbol{\Lambda}_2$、 $\boldsymbol{\Lambda}_3$的更新实现。

SpaCoBi算法的概要如算法1所示。

步骤1（更新$\mathbf{A}$）：我们需要最小化

$$
\begin{aligned}
f(\mathbf{A}) &= \frac{1}{2}\sum_{i=1}^{n}\|X_{i\cdot} - A_{i\cdot}\|_2^2 + \frac{\nu_1}{2}\sum_{l\in\mathcal{E}_1}\|\widetilde{\mathbf{v}}_l - A_{l_1\cdot} + A_{l_2\cdot}\|_2^2 \\
&\quad + \frac{\nu_2}{2}\sum_{k\in\mathcal{E}_2}\|\widetilde{\mathbf{z}}_k - \mathbf{a}_{k_1} + \mathbf{a}_{k_2}\|_2^2 + \frac{\nu_3}{2}\sum_{j=1}^{p}\|\widetilde{\mathbf{g}}_j - \mathbf{a}_j\|_2^2,
\end{aligned}
$$

其中 $\widetilde{\ }\mathbf{v}_1 = \mathbf{v}_l + \frac{1}{\nu_1}\boldsymbol{\lambda}_{1l}$、 $\widetilde{\ }\mathbf{z}_k = \mathbf{z}_k + \frac{1}{\nu_2}\boldsymbol{\lambda}_{2k}$ 和 $\widetilde{\ }\mathbf{g}_j = \mathbf{g}_j + \frac{1}{\nu_3}\boldsymbol{\lambda}_{3j}$。

这一步是 SpaCoBi 算法的关键组成部分。通过应用矩阵技术，可以通过求解以下方程得到$\mathbf{A}$ 的估计值，详细内容见附录：

$$
\mathbf{M}\mathbf{A} + \mathbf{A}\mathbf{N} = \mathbf{H}, \tag{3}
$$

其中

$$
\begin{aligned}
\mathbf{M} &= \mathbf{I}_n + \nu_1\sum_{l\in\mathcal{E}_1}(\mathbf{e}_{l_1} - \mathbf{e}_{l_2})(\mathbf{e}_{l_1} - \mathbf{e}_{l_2})^{\mathrm{T}} \\
\mathbf{N} &= \nu_2\sum_{k\in\mathcal{E}_2}(\mathbf{e}_{k_1}^* - \mathbf{e}_{k_2}^*)(\mathbf{e}_{k_1}^* - \mathbf{e}_{k_2}^*)^{\mathrm{T}} + \nu_3\sum_{j=1}^{p}\mathbf{e}_j^*(\mathbf{e}_j^*)^{\mathrm{T}} \\
\mathbf{H} &= \mathbf{X} + \sum_{l\in\mathcal{E}_1}(\mathbf{e}_{l_1} - \mathbf{e}_{l_2})(\boldsymbol{\lambda}_{1l} + \nu_1\mathbf{v}_l)^{\mathrm{T}} + \sum_{k\in\mathcal{E}_2}(\boldsymbol{\lambda}_{2k} + \nu_2\mathbf{z}_k)(\mathbf{e}_{k_1}^* - \mathbf{e}_{k_2}^*)^{\mathrm{T}} + \sum_{j=1}^{p}(\boldsymbol{\lambda}_{3j} + \nu_3\mathbf{g}_j)(\mathbf{e}_j^*)^{\mathrm{T}}.
\end{aligned}
$$

如果边集 $\mathcal{E}_1$ 和 $\mathcal{E}_2$ 包含所有可能的边，则不难验证

$$\begin{aligned}\sum_{l\in\mathcal{E}_1}(\mathbf{e}_{l_1}-\mathbf{e}_{l_2})(\mathbf{e}_{l_1}-\mathbf{e}_{l_2})^{\mathrm{T}} &= n\mathbf{I}_n-\mathbf{1}_n\mathbf{1}_n^{\mathrm{T}}\\ \sum_{k\in\mathcal{E}_2}(\mathbf{e}^*_{k_1}-\mathbf{e}^*_{k_2})(\mathbf{e}^*_{k_1}-\mathbf{e}^*_{k_2})^{\mathrm{T}} &= p\mathbf{I}_p-\mathbf{1}_p\mathbf{1}_p^{\mathrm{T}}.\end{aligned}$$

Then

$$\begin{aligned}\mathbf{M} &= (1+n\nu_1)\mathbf{I}_n-\nu_1\mathbf{1}_n\mathbf{1}_n^{\mathrm{T}}\\ \mathbf{N} &= p\nu_2\mathbf{I}_p-\nu_2\mathbf{1}_p\mathbf{1}_p^{\mathrm{T}}+\nu_3\mathbf{I}_p.\end{aligned}$$

方程（3）是一个标准的*Sylvester*方程，在控制理论和许多其他工程分支中发挥着重要作用。其理论解基于下文所示的特征向量和特征值分解（Jameson，1968），但计算成本很高。

假设$\mathbf{M}$的特征值为 $\lambda_i, i=1,\ldots,n$，而$\mathbf{N}$的特征值为 $\mu_j, j=1,\ldots,p$。那么，已知方程（3）有解当且仅当

$$\lambda_i+\mu_j\neq 0 \quad for\ \ all\ \ i,j.$$

这里，$\mathbf{M}$ 是正定的，$\mathbf{N}$ 是半正定的，这意味着$\mathbf{A}$ 是可解的。假设$\mathbf{M}$ 和$\mathbf{N}$ 可以通过正交变换对角化：

$$\mathbf{T}^{\mathrm{T}}\mathbf{M}\mathbf{T}=\begin{bmatrix}\lambda_1 & & & \\ & \lambda_2 & & \\ & & \ddots & \\ & & & \lambda_n\end{bmatrix}\quad \mathbf{S}^{\mathrm{T}}\mathbf{N}\mathbf{S}=\begin{bmatrix}\mu_1 & & & \\ & \mu_2 & & \\ & & \ddots & \\ & & & \mu_p\end{bmatrix}.$$

于是，解可表示为

$$\mathbf{A}=\mathbf{S}\widetilde{\mathbf{A}}\mathbf{T}^{\mathrm{T}},$$

其中$\widetilde{\mathbf{A}}=(\widetilde{a}_{ij})$,

$$\widetilde{a}_{ij}=\frac{\widetilde{c}_{ij}}{\mu_i+\lambda_j},\ \widetilde{\mathbf{C}}=(\widetilde{c}_{ij})=\mathbf{S}^{\mathrm{T}}\mathbf{C}\mathbf{T}.$$

或者，Bartels-Stewart算法（Bartels 和 Stewart，1972）是一种标准数值解法，它通过 Schur 分解将 *Sylvester* 方程 转换为三角系统，然后通过前代或回代求解。本文实现了 Sorensen等人（2003）提出的一种改进的 Bartels-Stewart算法，其计算效率更高。

步骤2（更新V、Z 和G）： 显然，向量 $\tilde{}\mathbf{v}_l$、$\tilde{}\mathbf{z}_k$ 和 $\tilde{}\mathbf{g}_j$ 在目标函数中是可分离的，因此 $\tilde{}\mathbf{v}_l$ 和 $\tilde{}\mathbf{z}_k$ 可以通过近端映射求解：

$$
\begin{aligned}
\mathbf{v}_l &= \underset{\mathbf{v}_l}{\operatorname{argmin}} \frac{1}{2}\|\mathbf{v}_l - (A_{l_1\cdot} - A_{l_2\cdot} - \nu_1^{-1}\boldsymbol{\lambda}_{1l})\|_2^2 + \frac{\gamma_1 w_l}{\nu_1}\|\mathbf{v}_l\|_q \\
&= \operatorname{prox}_{\sigma_{1l}\|\cdot\|_q}(A_{l_1\cdot} - A_{l_2\cdot} - \nu_1^{-1}\boldsymbol{\lambda}_{1l}) \\
\mathbf{z}_k &= \underset{\mathbf{z}_k}{\operatorname{argmin}} \frac{1}{2}\|\mathbf{z}_k - (\mathbf{a}_{k_1} - \mathbf{a}_{k_2} - \nu_2^{-1}\boldsymbol{\lambda}_{2k})\|_2^2 + \frac{\gamma_2 \widetilde{w}_k}{\nu_2}\|\mathbf{z}_k\|_q \\
&= \operatorname{prox}_{\sigma_{2k}\|\cdot\|_q}(\mathbf{a}_{k_1} - \mathbf{a}_{k_2} - \nu_2^{-1}\boldsymbol{\lambda}_{2k}) \\
\mathbf{g}_j &= \underset{\mathbf{g}_j}{\operatorname{argmin}} \frac{1}{2}\|\mathbf{g}_j - (\mathbf{a}_j - \nu_3^{-1}\boldsymbol{\lambda}_{3j})\|_2^2 + \frac{\gamma_3 u_j}{\nu_3}\|\mathbf{g}_j\|_2 \\
&= \operatorname{prox}_{\sigma_{3j}\|\cdot\|_2}(\mathbf{a}_j - \nu_3^{-1}\boldsymbol{\lambda}_{3j})
\end{aligned}
$$

其中 $\sigma_{1l} = \gamma_1 w_l/\nu_1$、$\sigma_{2k} = \gamma_2 \widetilde{w}_k/\nu_2$ 和 $\sigma_{3j} = \gamma_3 u_j/\nu_3$。关于 L_q-范数在 $q = 1$、2 和 ∞ 情况下的近端映射解，请参阅 Chi 和 Lange （ 2015 ）中的表 1。本文主要使用 L_2- 范数。

步骤 3（更新 $\boldsymbol{\Lambda}_1$，$\boldsymbol{\Lambda}_2$ 和 $\boldsymbol{\Lambda}_3$）： 我们通过以下方式更新$\boldsymbol{\lambda}_{1l}$，$\boldsymbol{\lambda}_{2k}$ 和 $\boldsymbol{\lambda}_{3j}$

$$
\begin{aligned}
\boldsymbol{\lambda}_{1l} &\leftarrow \boldsymbol{\lambda}_{1l} + \nu_1(\mathbf{v}_l - A_{l_1\cdot} + A_{l_2\cdot}), \\
\boldsymbol{\lambda}_{2k} &\leftarrow \boldsymbol{\lambda}_{2k} + \nu_2(\mathbf{z}_k - \mathbf{a}_{k_1} + \mathbf{a}_{k_2}), \\
\boldsymbol{\lambda}_{3j} &\leftarrow \boldsymbol{\lambda}_{3j} + \nu_3(\mathbf{g}_j - \mathbf{a}_j).
\end{aligned}
$$

算法1 SpaCoBi

1. 初始化 $\mathbf{V}^0$,$\mathbf{Z}^0, \mathbf{G}^0$， $\mathbf{\Lambda}_1^0$， $\mathbf{\Lambda}_2^0$和 $\mathbf{\Lambda}_3^0$。计算

$$\begin{aligned}\mathbf{M} &= (1+n\nu_1)\mathbf{I}_n - \nu_1 \mathbf{1}_n \mathbf{1}_n^{\mathrm{T}} \\ \mathbf{N} &= p\nu_2 \mathbf{I}_p - \nu_2 \mathbf{1}_p \mathbf{1}_p^{\mathrm{T}} + \nu_3 \mathbf{I}_p.\end{aligned}$$

对于 $m = 1$、2、.....

2. 求解*Sylvester*方程 $\mathbf{MA} + \mathbf{AN} = \mathbf{H}^{m-1}$，得到$\mathbf{A}^m$，其中

$$\begin{aligned}\mathbf{H}^{m-1} &= \mathbf{X} + \sum_{l\in\mathcal{E}_1} (\mathbf{e}_{l_1} - \mathbf{e}_{l_2})(\boldsymbol{\lambda}_{1l}^{m-1} + \nu_1 \mathbf{v}_l^{m-1})^{\mathrm{T}} + \\ &\quad \sum_{k\in\mathcal{E}_2} (\boldsymbol{\lambda}_{2k}^{m-1} + \nu_2 \mathbf{z}_k^{m-1})(\mathbf{e}_{k_1}^* - \mathbf{e}_{k_2}^*)^{\mathrm{T}} + \sum_{j=1}^{p} (\boldsymbol{\lambda}_{3j}^{m-1} + \nu_3 \mathbf{g}_j^{m-1})(\mathbf{e}_j^*)^{\mathrm{T}}.\end{aligned}$$

3. 对于 $l \in \mathcal{E}_1$，执行

$$\mathbf{v}_l^m = \mathrm{prox}_{\sigma_{1l}\|\cdot\|_q}(A_{l_1\cdot}^m - A_{l_2\cdot}^m - \nu_1^{-1}\boldsymbol{\lambda}_{1l}^{m-1}).$$

4. 对于 $k \in \mathcal{E}_2$，执行

$$\mathbf{z}_l^m = \mathrm{prox}_{\sigma_{2k}\|\cdot\|_q}(\mathbf{a}_{k_1}^m - \mathbf{a}_{k_2}^m - \nu_2^{-1}\boldsymbol{\lambda}_{2k}^{m-1}).$$

5. 对于 $j = 1, \ldots, p$，执行

$$\mathbf{g}_j^m = \mathrm{prox}_{\sigma_{3j}\|\cdot\|_2}(\mathbf{a}_j^m - \nu_3^{-1}\boldsymbol{\lambda}_{3j}^m).$$

6. 对于 $l \in \mathcal{E}_1$、 $k \in \mathcal{E}_2$ 和 $j = 1, \ldots, p$，执行

$$\begin{aligned}\boldsymbol{\lambda}_{1l}^m &= \boldsymbol{\lambda}_{1l}^{m-1} + \nu_1(\mathbf{v}_l^m - A_{l_1\cdot}^m + A_{l_2\cdot}^m) \\ \boldsymbol{\lambda}_{2k}^m &= \boldsymbol{\lambda}_{2k}^{m-1} + \nu_2(\mathbf{z}_k^m - \mathbf{a}_{k_1}^m + \mathbf{a}_{k_2}^m) \\ \boldsymbol{\lambda}_{3j}^m &= \boldsymbol{\lambda}_{3j}^{m-1} + \nu_3(\mathbf{g}_j^m - \mathbf{a}_j^m).\end{aligned}$$

7. 重复步骤 2–6，直到收敛。

3 实现

本节讨论实现所提出算法时的实际注意事项，包括算法收敛和调优参数的选择。

3.1 算法收敛

在凸聚类的背景下，Wang 等人（2023年）讨论了其凸双聚类算法的收敛性。式（2）中的目标函数与 Wang 等人（2023年）中的目标函数的主要区别，在于前者在特征层面的向量上额外加入了组 LASSO 惩罚项。根据 Chi 等人（2017年），这仍然是一个凸优化问题，并且在温和的

正则化条件下，SpaCoBi算法能够保证收敛。

3.2 权重的选择

本节介绍如何为融合LASSO惩罚选择权重 $w_l, l \in \mathcal{E}_1$ 和 $\widetilde{w}_k, l \in \mathcal{E}_2$ ，以及如何为组LASSO惩罚选择因子 $u_j, j = 1,\ldots,p$。按照 Chi和Lange（2015）的方法，我们结合 m 近邻法与高斯核来选择权重。具体而言，样本 (l_1, l_2) 之间的权重 w_l 定义如下:

$$w_l = t^m_{l_1,l_2} \exp\left(-\phi \|X_{l_1\cdot} - X_{l_2\cdot}\|_2^2\right), \tag{4}$$

其中，$t^m_{l_1,l_2}$ 在个体 l_2 位于个体 l_1的 m近邻范围内时取 1，否则取 0。同理，我们可以定义 $\widetilde{w}_k$。当 m 较小时，这种权重设定适用于广泛的 ϕ。在我们的数值结果中， m 固定为 5， ϕ 固定为 0.5。因子 u_j 可选为 $\frac{1}{\|\mathbf{a}_j^{(0)}\|_2}$，其中 $\mathbf{a}_j^{(0)}$ 是 $\gamma_3 = 0$ 时 (2) 中 $\mathbf{a}j$ 的估计值。与非自适应版本相比，这种因子选择方法对信息特征施加较小的惩罚，对非信息性特征施加较大的惩罚，从而提高聚类准确率和变量选择性能。最后，为确保最优调优参数 γ_1, γ_2 和 γ_3 无论维度和样本量如何都保持在相对稳定的范围内，我们将权重 w_l , $\widetilde{w}_k$以及因子 u_j 分别重新缩放，使其总和为 $\frac{1}{\sqrt{p}}, \frac{1}{\sqrt{n}}$和 $\frac{1}{\sqrt{n}}$。这种重新缩放便于在计算时同时考虑这两个参数，且不会影响最终的聚类路径。

3.3 调优参数的选择

本节讨论如何选择调优参数 γ_1、 γ_2和 γ_3。回顾一下， γ_1控制观测层级聚类的数量， γ_2 决定特征层级聚类的数量，而 γ_3 则调节特征向量的稀疏性。使用三个独立的调优参数，分别控制行列聚类的数量以及稀疏性，能够为算法的应用提供更大的

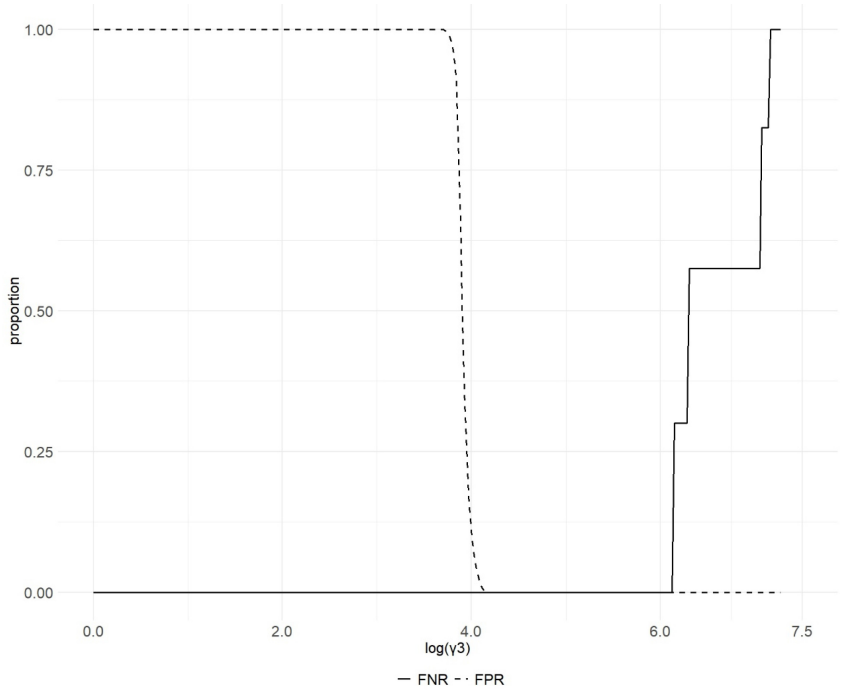


图 1：展示 γ_3 对变量选择准确性的影响。实线表示假阴性率（FNR）的变化曲线，虚线表示假阳性率（FPR）的变化曲线。

灵活性。

我们首先通过数值模拟，展示调优参数 γ_3 在控制变量选择准确性方面的效果。在此示例中，分别从 4 个类别生成了 60 个样本，每个样本包含 400 个特征。在这 400 个特征中，只有 40 个与聚类有关，其余 360 个均为噪声。详细的模拟设置请参阅数值模拟一节。将 γ_1 和 γ_2 固定为可能接近最优值的 50，并将 γ_3 从 e^0 逐步增大至 $e^{7.5}$，我们绘制了最终估计量的假阴性率（FNR）和假阳性率（FPR）变化曲线。如图（1）所示，当 γ_3 趋近于零时，所有特征都会被纳入。当 γ_3 增大到一定范围时，所有非信息性特征都会被排除，同时所有信息特征都会被完整保留，这意味着所有有用变量都被准确选出。这表明 γ_3 对最终估计量变量选择性能的影响程度。

一般而言，如果计算资源充足，三维网格搜索可以提供

一组在特定准则下的最优调优参数。考虑到调优参数 γ_1 和 γ_2 的特殊性，如果数据矩阵规模较大，三维网格搜索的计算负担会变得难以承受，此时我们可以采用类似于 Chi 等人（2017）提出的策略：将这两个调优参数与适当重新缩放的惩罚项结合起来。这种方法可以减轻计算负担，但需要按比例对行和列进行聚类。具体而言，我们将稀疏凸双聚类问题改写为如下仅含两个调优参数的最小化问题：

$$\min_{\mathbf{A}\in\mathbb{R}^{p\times n}} \quad \frac{1}{2}\sum_{i=1}^{n}\|X_{i\cdot}-A_{i\cdot}\|_2^2+\gamma\left\{\sum_{l\in\mathcal{E}_1}w_l\|\mathbf{v}_l\|_q+\sum_{k\in\mathcal{E}_2}\widetilde{w}_k\|\mathbf{z}_k\|_q\right\}+\gamma_3\sum_{j=1}^{p}u_j\|\mathbf{g}_j\|_2 \qquad (5)$$

现有研究已提出多种双聚类调优参数选择策略。Chi 等人（2017）针对凸双聚类提出了一种留出验证法：从数据矩阵中随机选择元素，并利用基于其余元素估计的模型评估对留出集进行预测的质量。然而，Fang 和 Wang（2012）指出，数据拆分会减少训练数据集的规模，从而使交叉验证方法效率低下。他们提出在聚类分析中采用稳定性选择，该方法也被后续聚类研究所采用。Wang 等人（2018）和 Wang 等人（2023）分别在其稀疏凸聚类和凸双聚类算法中采用了稳定性选择。对于稀疏凸双聚类，我们以类似方式应用稳定性选择来调优 γ_1、γ_2 和 γ_3。

具体而言，对于两个自助抽样样本和一组调优参数，聚类算法可以生成两个双聚类结果，每个结果都包含中心和聚类数。利用这两个双聚类结果，可以计算稳定性度量，以评估两个聚类结果之间的一致性；该度量使用了 Fang 和 Wang（2012）定义的聚类距离：

定义 1：（聚类距离）任意两个聚类 $\psi_1(x)$ 和 $\psi_2(x)$ 之间的距离定义为

$$d_F\left(\psi_1,\psi_2\right)=E_{x^0\sim F,y^0\sim F}\left\{\left|I\left\{\psi_1\left(x^0\right)=\psi_1\left(y^0\right)\right\}-I\left\{\psi_2\left(x^0\right)=\psi_2\left(y^0\right)\right\}\right|\right\},$$

其中 $I\{\cdot\}$ 为指示函数，期望是对 x^0 和 y^0取的，它们是从 F 中抽取的两个独立观测。

3.4 热启动

为减轻 SpaCoBi算法在多次重复模拟中带来的巨大计算负担，我们探索了一种“热启动”策略。在机器学习和推荐系统中，冷启动和热启动指系统如何应对数据可用性不同阶段所带来的挑战。冷启动通常意味着缺乏历史数据，而热启动则是一种有价值的优化技术。具体而言，热启动将相关问题或简化问题的最优解用作当前更复杂问题的初始值。这种高质量的初始化使优化器能够从更接近全局最优解的位置开始，从而加快收敛并提高计算效率，尤其是在存在多个局部极小值的非线性优化问题中。

在一些双聚类文献中，研究人员采用激进的计算方案，利用良好的初始点执行一次迭代，以获得近似解（Ramachandra等人，2023年）。尽管这种方法显著提高了计算速度，我们采用的策略则更加系统。在本文中，我们运用热启动策略，将网格搜索中前一个优化步骤的收敛结果用作后续优化的初始值。在广泛的数值实验中，这种方法取得了最佳的计算结果。

我们进行了实验，比较了在不同样本量 n、总特征数 p 和真实信息特征数 p_{true} 条件下，十个不同网格搜索点的计算时间和迭代次数。对于每个样本，我们重复进行了 30 次实验，并计算平均值。我们将 γ_1 和 γ_2 固定为 50（一个可能的最优参数），同时在 30 到 150 之间的 10 个值中搜索最优的 γ_3。两个程序均采用相同的迭代收敛容差 e^{-5} 运行。尽管将容差收紧至 e^{-6} 到 e^{-7} 可以获得

略微更好的结果，但在大多数数据集中，容差为 e^{-5} 通常足以获得相当准确的簇。表中所列时间以秒为单位，表示程序达到迭代收敛所需的平均时长；括号中的数值则表示每次收敛所需的平均迭代次数。实验在一台配备 AMD Ryzen 5 4600H 中央处理器和 16GB 随机存取存储器的计算机上进行。

表1：SpaCoBi 热启动的计算效率分析（时间以秒为单位，平均迭代次数）

参数	$n = 60$			$n = 120$			$n = 200$	
p	120	200	400	120	200	400	120	200
p_{true}	40	40	40	40	40	40	40	40
SpaCoBi	6.56 (112.8)	10.56 (62.5)	18.41 (32.3)	13.59 (125.4)	12.80 (40.5)	27.04 (24.1)	22.42 (121.4)	13.96 (22.4)
SpaCoBi（热启动）	4.21（57.3）	8.71（49.2）	13.95（23.9）	8.05（42.9）	10.29（33.3）	17.81 (15.7)	11.40 (33.6)	10.16 (16.3)
效率 ↑	55.72%	21.27%	32.01%	68.80%	24.45%	51.86%	96.66%	37.51%

通过比较表1中的时间和迭代次数，我们发现，与不具备热启动功能的SpaCoBi算法相比，热启动算法将计算效率至少提高了21.72%（作为参考，将迭代时间从100秒缩短至50秒，代表效率提高了100%）。这些结果清楚地凸显了采用热启动策略的显著优势。在处理需要进行双聚类的现实数据时，实践者通常会固定两个参数，并搜索重复的惩罚参数，这与我们采用十个搜索点的方法相似。在这种常见情形下，热启动方法被证明能够非常有效地加速SpaCoBi程序。因此，本文随后呈现的所有数值计算均采用了这一增强方法。

4 数值结果

本节旨在展示我们提出的稀疏凸双聚类（SpaCoBi）方法的优越性能。我们在两个不同领域进行评估：一组全面的模拟示例（第 4.1 小节），以及一项基于小鼠嗅球（MOB）基因表达数据的真实应用（第 4.2 小节）。

在模拟研究中，每项模拟均重复 50 次，以确保统计推断稳健。用于量化双聚类结果准确率的主要指标是调整兰德指数（ARI），它衡量估计的双聚类分配与真实聚类标签之间的一致程度；在模拟情境中，这些标签是预先定义且已知的。ARI 的取值范围为 -1 至 1，数值越高表示聚类性能越优。已知真实聚类标签后，可以通过调整候选方法以最大化 ARI，评估其潜在的最高性能。此外，我们还使用假阴性率（FNR）和假阳性率（FPR），严格评估算法选择特征的能力。针对双聚类问题，我们根据上述指标，将所提出的 SpaCoBi 算法的数值表现与 Bi-ADMM（（Wang 等人，2023年））和 COBRA（Chi 等人，2017年）进行比较。

为了与 Bi-ADMM（L_2）方法进行公平比较，我们采用公式（5）中的表述实现了这两种算法，该表述要求对调优参数进行二维网格搜索。对于每次重复实验，通过最大化验证数据矩阵上的 ARI 来确定最优调优参数组合。该验证数据矩阵与训练数据具有相同的底层分类结构，但彼此独立（Witten 和 Tibshirani，2010）。

4.1 模拟研究

我们模拟一个由非信息性特征和棋盘式双聚类结构组成的 $n \times p$ 数据矩阵。该结构包含具有非零均值的 p_{true} 信息特征和$p - p_{\text{true}}$ 非信息性特征。对于信息特征， X_{ij} 按如下方式生成：通过从集合 $\{1, \ldots, 4\}$中随机抽样，为观测（行）分配簇索引；特征（列）的簇索引则采用类似的方法分配。因此，不同运行中生成的数据矩阵和分类结果会有所不同。例如，对于分为 4 个类别的 60 个观测，一次运行可能产生 4 个各含 15 个元素的组，而下一次运行中各类别的样本数可能分别为 10，5，5和 *20*。双聚类组的总数为，这表示每个都属于这些双聚类组之一。随后，从正态分布中为每个双聚类组生成随机样本：独立同分布（*i.i.d.*）*(*， σ^2)，其中来自行簇的样本

$k \in \{1, \ldots, 4\}$ 和列簇 $r \in \{1, \ldots, 4\}$ 服从均值为 μ_{kr}、方差为 σ^2的正态分布。均值 μ_{kr} 从序列 $\{-10, -9, \ldots, 9, 10\}$中均匀选取。最后，其余 $p - p_{\text{true}}$ 噪声特征由 $\mathcal{N}$(0，9)生成。

表2所示结果清楚表明，在所有测试的样本量设置下，SpaCoBi方法的表现始终优于Bi-ADMM(L_2)算法。值得注意的是，随着特征维度（p）增加，尤其是在高维情形下，缺乏稀疏性惩罚的Bi-ADMM(L_2)算法性能迅速下降。相比之下，SpaCoBi方法的性能下降则不那么明显。在大样本条件下，SpaCoBi的表现显著更优，凸显了信息特征选择机制在高维双聚类过程中的关键作用。此外，只要数据中存在非信息性特征，SpaCoBi的调整兰德指数（ARI）就始终高于Bi-ADMM(L_2)算法。

从表3所示的曲线下面积（AUC）数值可以看出，在这些模拟条件下，SpaCoBi方法几乎能够完美识别信息特征，其AUC接近0.8。假阴性率（FNR）和假阳性率（FPR）均较低，也印证了其较高的特征选择准确率。Bi-ADMM(L_2)情形下聚类准确率较低，可直接归因于非信息性特征过多，这凸显了选择信息特征的必要性，也表明SpaCoBi算法的特征选择机制具有更强的能力。这一观察结果与Tan 和Witten（2014年）以及Wang 等人（2018年）在稀疏凸聚类研究中报告的类似发现一致；这些研究为SpaCoBi的开发提供了重要的理论基础和动机。

4.2 MOB 数据的应用

本研究使用了一个真实世界的小鼠嗅球（MOB）基因表达数据集，其样本标签由高级自然研究所生物技术研究中心确定

表2：SpaCoBi、Bi-ADMM 和 COBRA 在 ARI 指标上的模拟结果，分别列出训练集（记作 Train）和验证集（记作 Val）的结果。

n	p	p_{true}	SpaCoBi				Bi-ADMM				COBRA			
			训练		Val		训练		Val		训练		Val	
			均值	标准差	均值	标准差	均值	标准差	均值	标准差	均值	标准差	均值	标准差
60	200	40	0.82	0.13	0.84	0.12	0.77	0.12	0.80	0.10	0.15	0.18	0.15	0.18
	400	40	0.91	0.05	0.79	0.08	0.77	0.05	0.77	0.05	0.06	0.08	0.06	0.08
	600	40	0.83	0.03	0.79	0.08	0.25	0.27	0.26	0.25	0.02	0.02	0.02	0.02
	200	60	0.83	0.14	0.86	0.13	0.79	0.15	0.82	0.14	0.50	0.25	0.52	0.26
	400	60	0.72	0.21	0.77	0.19	0.66	0.21	0.68	0.21	0.19	0.16	0.19	0.16
	600	60	0.78	0.17	0.74	0.14	0.15	0.18	0.15	0.19	0.03	0.02	0.03	0.02
120	200	40	0.82	0.15	0.83	0.14	0.77	0.15	0.78	0.15	0.17	0.18	0.17	0.18
	400	40	0.96	0.03	0.95	0.02	0.73	0.04	0.76	0.05	0.05	0.08	0.05	0.08
	600	40	0.75	0.03	0.75	0.03	0.20	0.21	0.20	0.21	0.03	0.02	0.03	0.02
	200	60	0.84	0.12	0.89	0.10	0.40	0.20	0.41	0.20	0.49	0.21	0.49	0.21
	400	60	0.71	0.22	0.76	0.21	0.23	0.22	0.24	0.22	0.19	0.19	0.19	0.19
	600	60	0.94	0.01	0.96	0.02	0.12	0.15	0.12	0.15	0.09	0.12	0.09	0.12
240	200	40	0.79	0.17	0.85	0.15	0.26	0.18	0.27	0.18	0.24	0.19	0.24	0.19
	400	40	0.72	0.03	0.72	0.03	0.25	0.24	0.25	0.24	0.06	0.07	0.06	0.07
	600	40	0.79	0.13	0.78	0.11	0.13	0.18	0.13	0.18	0.03	0.02	0.03	0.02
	200	60	0.91	0.07	0.95	0.06	0.46	0.28	0.47	0.28	0.42	0.20	0.42	0.20
	400	60	0.84	0.15	0.85	0.15	0.30	0.22	0.30	0.22	0.16	0.19	0.19	0.17
	600	60	0.83	0.15	0.84	0.15	0.28	0.25	0.28	0.25	0.10	0.11	0.10	0.11

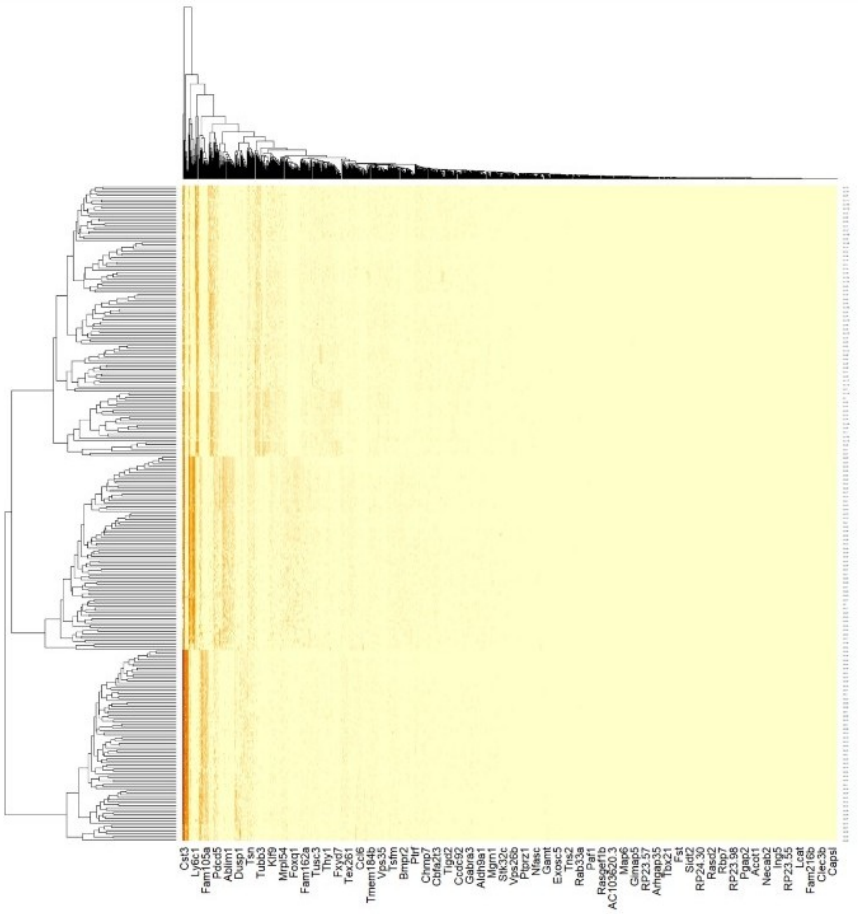
表3：特征选择性能比较：假阴性率（FNR）、假阳性率（FPR）和曲线下面积（AUC）

n	p	p_{true}	SpaCoBi						Bi-ADMM			
			FNR		FPR		AUC		FNR		FPR	
			均值	标准差	均值	标准差	均值	标准差	均值	标准差	均值	标准差
60	200	40	0.06	0.04	0.23	0.16	0.90	0.15	0.00	0.00	1.00	0.00
	400	40	0.00	0.01	0.04	0.05	0.79	0.15	0.00	0.00	1.00	0.00
	600	60	0.02	0.03	0.13	0.16	0.89	0.16	0.00	0.00	1.00	0.00
120	200	40	0.07	0.03	0.27	0.12	0.81	0.12	0.00	0.00	1.00	0.00
	400	60	0.03	0.04	0.16	0.22	0.78	0.13	0.00	0.00	1.00	0.00
240	200	40	0.03	0.04	0.12	0.18	0.76	0.24	0.00	0.00	1.00	0.00
	400	40	0.01	0.02	0.12	0.18	0.88	0.18	0.00	0.00	1.00	0.00

图2：原始数据的热图。

北京师范大学生命科学学院，作为一个生物基因组学应用案例，用于评估稀疏凸双聚类算法（SpaCoBi）和凸双聚类算法（Bi-ADMM）在 L_2-范数下的性能。原始数据集包含305个观测样本和1,250个基因特征。如图2所示，原始数据的热图呈现出两个显著特征：第一，明显的垂直条纹模式，表明某些样本子集可能会显著影响聚类结果；第二，大范围接近零的表达值区域，表明数据具有高度稀疏性，并包含大量无信息特征。这些高维且高度稀疏的特征促使研究采用SpaCoBi算法，该算法利用稀疏性揭示潜在的聚类结构并识别关键遗传标记。

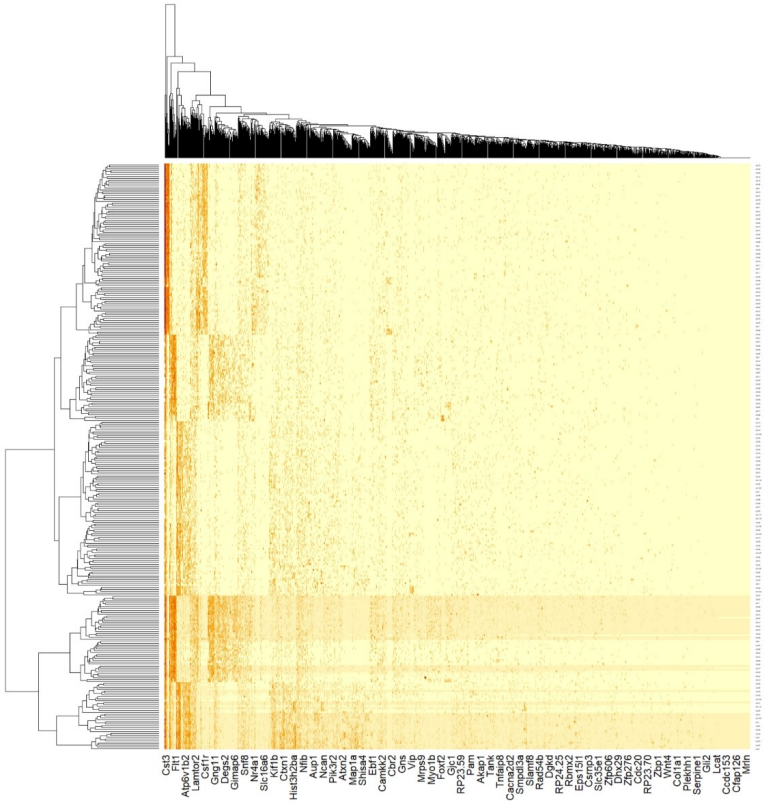
为了严格评估 SpaCoBi 和 Bi-ADMM（L_2-范数）算法在基因组数据分析中的实际效用，我们利用了这 305 个样本已知的生物学分类。通过将两种算法的聚类结果与真实生物学类别进行比较，我们定量评估了它们的准确率。两种算法均采用相同的



ˆ 图3：SpaCoBi算法估计的 A 热图

惩罚参数设置，以确保比较公平。图 3 和图 4 分别展示了 SpaCoBi 和 Bi-ADMM 得到的聚类结果热图。比较分析表明，SpaCoBi 生成的热图对簇的划分更清晰，并能有效抑制无信息特征，从而凸显其稀疏性诱导惩罚项的优势。此外，SpaCoBi 得出的聚类结构与样本已知的三类组织结构高度吻合，而 Bi-ADMM 则难以区分这些类别。这一观察结果得到了调整兰德指数（ARI）的定量支持：SpaCoBi 的 ARI 达到 1.0，而 Bi-ADMM 仅为 0.12，两者形成鲜明对比。

SpaCoBi算法的 ARI 达到 1.0，有力地证明了它能够准确捕捉高维生物基因表达数据中的内在聚类结构，分类结果与真实的生物学注释完全一致。这一显著



ˆ 图4：由 Bi-ADMM算法估计的 A 热图

这一改进归因于 SpaCoBi 能够通过其内在的稀疏机制识别并减轻无关或含噪特征的影响，从而提高分类准确率。根据 SpaCoBi 的特征选择结果，对聚类有显著贡献的关键基因特征包括：“Pbxip1”、“Pdlim2”、“Cdc34”、“Kdm7a”、“Ptprz1”、“Kctd13”、“Higd1b”、“Bcas1”、“Gpcpd1”、“Man2b2”、“Inpp4a”、“Mef2c”、“Ftsj3”、“Flii”、“Osr1”、“Slc39a1”、“Armc6”、“标签”、“Nell2”、“RP23.96”、“Car4”、“Epb41l5”和“Isg15”。这组筛选出的信息性基因为未来在分子生物学领域开展实验验证和机制研究提供了宝贵见解与有针对性的研究方向。

5 结论

本文提出了稀疏凸双聚类算法，作为一种分析高维数据的稳健方法。通过全面的模拟和一项使用小鼠嗅球基因表达数据集的真实应用，我们展示了 SpaCoBi 相较于现有凸双聚类算法的显著优势。

基于MOB数据的结果表明，在聚类准确率方面，SpaCoBi显著优于Bi-ADMM；调整兰德指数（ARI）达到1.0，证明其能够准确捕捉数据的内在聚类结构。这种效率源于SpaCoBi固有的稀疏机制，该机制能够有效识别并消除无关或含噪特征的影响，从而提高分类精度。此外，SpaCoBi的特征选择能力凸显了Pbxip1、Pdlim2、Cdc34等关键信息性基因，为理解数据背后的生物过程提供了宝贵的见解。这一筛选出的基因子集为分子生物学领域未来的实验验证和机制研究指明了具体方向。总之，本研究强调了在高维场景中选择信息特征的重要性，并展示了SpaCoBi算法如何成为基因组数据分析的有力工具。

参考文献

Bartels, R. H. 和 Stewart, G. W.（1972）。矩阵方程 ax+xb=c [f4] 的解。*ACM* 通讯，15(9):820–826。

Bergmann, S.、Ihmels, J. 和 Barkai, N.（2003）。用于分析大规模基因表达数据的迭代特征算法。《物理评论*E*》，67(3)：031902。

Busygin, S.、Prokopyev, O. 和 Pardalos, P. M.（2008）。数据挖掘中的双聚类。《计算机与运筹研究》，35(9)：2964–2987。

Chen, G.、Sullivan, P. F. 和 Kosorok, M. R.（2013）。异方差双聚类。《美国国家科学院院刊》，110(30)：12253–12258。

Chi, E. C.、Allen, G. I. 和 Baraniuk, R. G.（2017）。凸双聚类。《生物统计学》，73(1)：10–19。

Chi, E. C. 和 Lange, K.（2015）。凸聚类的分裂方法。《计算与图形统计学杂志》，24(4)：994–1013。

Fang, Y. 和 Wang, J.（2012）。通过自助法选择聚类数。《计算统计与数据分析》，56：468–477。

Helgeson, E. S.、Liu, Q.、Chen, G.、Kosorok, M. R. 和 Bair, E.（2020）。通过稀疏聚类实现双聚类。《生物统计学》，76(1)：348–358。

Jameson, A.（1968）。通过求逆一个 m*m 或 n*n 矩阵求解方程 ax+xb=c。《*SIAM*应用数学杂志》，16(5)：1020–1023。

Lazzeroni, L. 和 Owen, A.（2002）。用于基因表达数据的格子模型。《中华统计学志》，第 61–86 页。

Lee, M.、Shen, H.、Huang, J. Z. 和 Marron, J. S.（2010）。通过稀疏奇异值分解进行双聚类。《生物统计学》，66(4)：1087–1095。

Madeira, S. C. 和 Oliveira, A. L.（2004）。生物学数据的双聚类算法分析：综述。*IEEE/ACM* 计算生物学与生物信息学汇刊（*TCBB*），1(1)：24–45。

Ramachandra, H., Ali, A., Ambili, P., Thota, S., 和 Asha, P.（2023年）。用于基因表达数据的双聚类算法的优化。载于*2023*年第四届*IEEE*技术进步全球会议（*GCAT*），第1–6页。IEEE。

Sill, M., Kaiser, S., Benner, A., 和 Kopp-Schneider, A.（2011）。通过结合稳定性选择的稀疏奇异值分解实现稳健双聚类。生物信息学，27（15）：2089–2097。

Sorensen, D. C., Zhou, Y., 等（2003）。矩阵 Sylvester 方程和 Lyapunov 方程的直接求解方法。应用数学杂志，6（2003）：277–303。

Tan, K. M. 和 Witten, D. M.（2014年）。可转置数据的稀疏双聚类。计算与图形统计学杂志，23（4）：985–1008。PMID：25364221。

Turner, H., Bailey, T., 和 Krzanowski, W.（2005）。通过系统性能测试展示改进的微阵列数据双聚类。计算统计与数据分析，48(2):235–254。

Wang, B., Yao, L., Hu, J., 和 Li, H.（2023年）。一种新的凸双聚类算法及其在组成数据中的扩展。生物科学统计学，15(1):193–216。

Wang, B., Zhang, Y., Sun, W. W., 和 Fang, Y.（2018年）。稀疏凸聚类。计算与图形统计学杂志，27:393–403。

Witten, D. 和 Tibshirani, R.（2010）。聚类中特征选择的框架。美国统计学会杂志，105:713–726。

Xie, J., Ma, A., Fennell, A., Ma, Q., 和 Zhao, J.（2019）。是时候应用双聚类了：一篇

关于双聚类在生物和生物医学数据中应用的全面综述。*Briefingsin bioinformatics*，20(4):1450–1465。

6 附录

第2.2节第1步中 A 的更新推导详情：

注意，$A_{l_1\cdot} - A_{l_2\cdot} = \mathbf{A}^{\mathrm{T}}(\mathbf{e}_{l_1} - \mathbf{e}_{l_2})$、$\mathbf{a}_{k_1} - \mathbf{a}_{k_2} = \mathbf{A}(\mathbf{e}^*_{k_1} - \mathbf{e}^*_{k_2})$ 以及 $\mathbf{a}_j = \mathbf{A}\mathbf{e}^*_j$，其中 $\mathbf{e}_{l_1}$ 是一个 n 维向量，其第 l_1 个元素为 1，其余元素均为 0；$\mathbf{e}^*_{k_1}$ 是一个 p 维向量，其第 k_1 个元素为 1，其余元素均为 0。对矩阵 $\boldsymbol{a} = \mathrm{vec}(\mathbf{A})$ 进行矩阵向量化，并应用恒等式

$$\mathrm{vec}(\mathbf{RST}) = [\mathbf{T}^{\mathrm{T}} \otimes \mathbf{R}]\mathrm{vec}(\mathbf{S}),$$

可得

$$f(\boldsymbol{a}) = \frac{1}{2}\|\mathbf{x} - \boldsymbol{a}\|_2^2 + \frac{\nu_1}{2}\sum_{l\in\mathcal{E}_l}\|\mathbf{B}_l\mathbf{P}\boldsymbol{a} - \widetilde{\mathbf{v}}_l\|_2^2 + \frac{\nu_2}{2}\sum_{k\in\mathcal{E}_2}\|\mathbf{C}_k\boldsymbol{a} - \widetilde{\mathbf{z}}_k\|_2^2 + \frac{\nu_3}{2}\sum_{j=1}^{p}(\mathbf{D}_j\boldsymbol{a} - \widetilde{\mathbf{g}}_j)^2,$$

其中

$$\begin{aligned}\mathbf{B}_l &= (\mathbf{e}_{l_1} - \mathbf{e}_{l2})^{\mathrm{T}} \otimes \mathbf{I}_p,\ \mathbf{C}_k = (\mathbf{e}^*_{k_1} - \mathbf{e}^*_{k2})^{\mathrm{T}} \otimes \mathbf{I}_n \\ \mathbf{D}_i &= (\mathbf{e}^*_j)^{\mathrm{T}} \otimes \mathbf{I}_n,\ \mathrm{vec}(\mathbf{A}^{\mathrm{T}}) = \mathbf{P}\mathrm{vec}(\mathbf{A}).\end{aligned}$$

稍微滥用一下记号，注意到 $\mathbf{P} = (P_{kl})$ $1 \le k,\ l \le np$ 存在唯一的置换矩阵，使得 $P_{kl} = 1$ 当 $k = (i-1)n + j$ 且 $l = (j-1)p + i,\ 1 \le i \le p,\ 1 \le j \le n$，否则为 0。容易看出 $\mathbf{P}^{\mathrm{T}} = \mathbf{P}^{-1}$。令 $\varepsilon_1 = |\mathcal{E}_1|,\ \varepsilon_2 = |\mathcal{E}_2|$，并且

$$\begin{aligned}\mathbf{B}^{\mathrm{T}} &= \left(\mathbf{B}_1^{\mathrm{T}}, \ldots, \mathbf{B}_{\varepsilon_1}^{\mathrm{T}}\right), \quad \widetilde{\mathbf{v}}^{\mathrm{T}} = \left(\widetilde{\mathbf{v}}_1^{\mathrm{T}}, \ldots, \widetilde{\mathbf{v}}_{\varepsilon_1}^{\mathrm{T}}\right) \\ \mathbf{C}^{\mathrm{T}} &= \left(\mathbf{C}_1^{\mathrm{T}}, \ldots, \mathbf{C}_{\varepsilon_2}^{\mathrm{T}}\right), \quad \widetilde{\mathbf{z}}^{\mathrm{T}} = \left(\widetilde{\mathbf{z}}_1^{\mathrm{T}}, \ldots, \widetilde{\mathbf{z}}_{\varepsilon_2}^{\mathrm{T}}\right) \\ \mathbf{D}^{\mathrm{T}} &= \left(\mathbf{D}_1^{\mathrm{T}}, \ldots, \mathbf{D}_n^{\mathrm{T}}\right), \quad \widetilde{\mathbf{g}}^{\mathrm{T}} = \left(\widetilde{\mathbf{g}}_1^{\mathrm{T}}, \ldots, \widetilde{\mathbf{g}}_p^{\mathrm{T}}\right).\end{aligned}$$

于是我们有

$$f(\boldsymbol{a}) = \frac{1}{2}\|\mathbf{x} - \boldsymbol{a}\|_2^2 + \frac{\nu_1}{2}\|\mathbf{B}\mathbf{P}\boldsymbol{a} - \widetilde{\mathbf{v}}\|_2^2 + \frac{\nu_2}{2}\|\mathbf{C}\boldsymbol{a} - \widetilde{\mathbf{z}}\|_2^2 + \frac{\nu_3}{2}\|\mathbf{D}\boldsymbol{a} - \widetilde{\mathbf{g}}\|_2^2.$$

稳态方程可通过以下方式得到

$$(\mathbf{I}_{np} + \nu_1\mathbf{P}^{\mathrm{T}}\mathbf{B}^{\mathrm{T}}\mathbf{B}\mathbf{P} + \nu_2\mathbf{C}^{\mathrm{T}}\mathbf{C} + \nu_3\mathbf{D}^{\mathrm{T}}\mathbf{D})\boldsymbol{a} = \mathbf{x} + \nu_1\mathbf{P}^{\mathrm{T}}\mathbf{B}^{\mathrm{T}}\widetilde{\mathbf{v}} + \nu_2\mathbf{C}^{\mathrm{T}}\widetilde{\mathbf{z}} + \nu_3\mathbf{D}^{\mathrm{T}}\widetilde{\mathbf{g}}.$$

这是一个由 np线性方程组成的方程组。我们可以利用克罗内克积的性质来简化其形式，例如 $(\mathbf{S}\otimes\mathbf{T})^{\mathrm{T}} = \mathbf{S}^{\mathrm{T}}\otimes\mathbf{T}^{\mathrm{T}}$ 以及 $(\mathbf{Q}\otimes\mathbf{R})(\mathbf{S}\otimes\mathbf{T}) = (\mathbf{Q}\mathbf{S})\otimes(\mathbf{R}\mathbf{T})$。由此可得

$$\begin{aligned}
\nu_2\mathbf{C}^{\mathrm{T}}\mathbf{C} &= \nu_2\sum_{k\in\mathcal{E}_2}\left[\left((\mathbf{e}_{k_1}^* - \mathbf{e}_{k_2}^*)(\mathbf{e}_{k_1}^* - \mathbf{e}_{k_2}^*)^{\mathrm{T}}\right)\right]\otimes\mathbf{I}_n \\
&= \left[\sum_{k\in\mathcal{E}_2}\nu_2\left((\mathbf{e}_{k_1}^* - \mathbf{e}_{k_2}^*)(\mathbf{e}_{k_1}^* - \mathbf{e}_{k_2}^*)^{\mathrm{T}}\right)\otimes\right]\mathbf{I}_n \\
\nu_2\mathbf{C}^{\mathrm{T}}\widetilde{\mathbf{z}} &= \nu_2\sum_{k\in\mathcal{E}_2}[(\mathbf{e}_{k_1}^* - \mathbf{e}_{k_2}^*)\otimes\mathbf{I}_n]\widetilde{\mathbf{z}}_k = \nu_2\sum_{k\in\mathcal{E}_2}\left((\mathbf{e}_{k_1}^* - \mathbf{e}_{k_2}^*)\otimes\mathbf{I}_n\right)\widetilde{\mathbf{z}}_k \\
\nu_3\mathbf{D}^{\mathrm{T}}\mathbf{D} &= \nu_3\sum_{j=1}^{p}\mathbf{D}_i^{\mathrm{T}}\mathbf{D}_i = \left[\nu_3\sum_{j=1}^{p}\mathbf{e}_j^*(\mathbf{e}_j^*)^{\mathrm{T}}\right]\otimes\mathbf{I}_n \\
\nu_3\mathbf{D}^{\mathrm{T}}\widetilde{\mathbf{g}} &= \nu_3\sum_{j=1}^{p}\mathbf{D}_j^{\mathrm{T}}\widetilde{\mathbf{g}}_j = \nu_3\sum_{j=1}^{p}\left(\mathbf{e}_j^*\otimes\mathbf{I}_n\right)\widetilde{\mathbf{g}}_j.
\end{aligned}$$

这里，我们应用命题1中所示的 $\mathbf{P}$ 的性质，可以得到

$$\begin{aligned}
\mathbf{I}_{np} + \nu_1\mathbf{P}^{\mathrm{T}}\mathbf{B}^{\mathrm{T}}\mathbf{B}\mathbf{P} &= \mathbf{I}_p\otimes\left[\mathbf{I}_n + \nu_1\sum_{l\in\mathcal{E}_1}(\mathbf{e}_{l_1} - \mathbf{e}_{l_2})(\mathbf{e}_{l_1} - \mathbf{e}_{l_2})^{\mathrm{T}}\right] \\
\nu_1\mathbf{P}^{\mathrm{T}}\mathbf{B}^{\mathrm{T}}\widetilde{\mathbf{v}} &= \sum_{l\in\mathcal{E}_1}\left[(\mathbf{I}_p\otimes\nu_1(\mathbf{e}_{l_1} - \mathbf{e}_{l_2}))\,\widetilde{\mathbf{v}}_l\right].
\end{aligned}$$

因此，该方程组等价于

$$\begin{aligned}
&(\mathbf{I}_p\otimes\mathbf{M})\mathrm{vec}(\mathbf{A}) + (\mathbf{N}\otimes\mathbf{I}_n)\mathrm{vec}(\mathbf{A}) = \mathrm{vec}(\mathbf{H}) \\
\Leftrightarrow\quad &\mathbf{M}\mathbf{A} + \mathbf{A}\mathbf{N} = \mathbf{H}. \quad \blacksquare
\end{aligned}$$

命题1 对于上文定义的置换矩阵$\mathbf{P}$，我们可以证明，对于任意 $k\in\mathcal{E}_2$ 和

p维向量 $\mathbf{d}$,

(1). $\left[\mathbf{d}^{\mathrm{T}} \otimes \mathbf{I}_p\right]\mathbf{P} = \mathbf{I}_p \otimes \mathbf{d}^{\mathrm{T}}$;

(2). $\mathbf{P}^{\mathrm{T}}\left[\left(\mathbf{d}\mathbf{d}^{\mathrm{T}}\right) \otimes \mathbf{I}_p\right]\mathbf{P} = \mathbf{I}_p \otimes \left(\mathbf{d}\mathbf{d}^{\mathrm{T}}\right)$。

命题1的证明如下:

(1). 注意，$\mathbf{P} = (P_{kl})\ 1 \le k, l \le np$ 这里存在唯一的置换矩阵，使得 $P_{kl} = 1$当 $k = (i-1)p + j$ 且 $l = (j-1)n + i, 1 \le i \le n, 1 \le j \le p$，否则为 0。根据**P**的定义，显然，右乘矩阵**P**会将其第 k 列移至第 l列，其中 $P_{kl} = 1$。

考虑 $\mathbf{d}$ 的第 i 个元素 d_i，那么在 $\mathbf{d}^{\mathrm{T}} \otimes \mathbf{I}_p$ 中，其位于 $(j, (i-1)p + j)$ 的元素为 d_i, $j = 1, \ldots, p$。因此，在 $(\mathbf{d}^{\mathrm{T}} \otimes \mathbf{I}_n)\mathbf{P}$ 中，位于 $(j, (j-1)n + i)$ 的元素等于 d_i。在 $\mathbf{I}_p \otimes \mathbf{d}^{\mathrm{T}}$ 中，很容易看出位于 $(j, (j-1)n + i)$ 的元素等于 $d_i, i = 1, \ldots, n, j = 1, \ldots, p$。

(2).

$$\begin{aligned}
\mathbf{P}^{\mathrm{T}}\left[\left(\mathbf{d}\mathbf{d}^{\mathrm{T}}\right) \otimes \mathbf{I}_p\right]\mathbf{P} &= \mathbf{P}^{\mathrm{T}}\left[(\mathbf{d} \otimes \mathbf{I}_p)(\mathbf{d}^{\mathrm{T}} \otimes \mathbf{I}_p)\right]\mathbf{P} \\
&= \left[(\mathbf{d}^{\mathrm{T}} \otimes \mathbf{I}_p)\mathbf{P}\right]^{\mathrm{T}}\left[(\mathbf{d}^{\mathrm{T}} \otimes \mathbf{I}_p)\mathbf{P}\right] \\
&= (\mathbf{I}_p \otimes \mathbf{d}^{\mathrm{T}})^{\mathrm{T}}(\mathbf{I}_p \otimes \mathbf{d}^{\mathrm{T}}) \\
&= \mathbf{I}_p \otimes \left(\mathbf{d}\mathbf{d}^{\mathrm{T}}\right). \quad \blacksquare
\end{aligned}$$