Frontiers in Applied Mathematics and Statistics

TYPE Original Research
PUBLISHED 21 August 2025
DOI 10.3389/fams.2025.1632218

OPEN ACCESS

EDITED BY
Lixin Shen,
Syracuse University, United States

REVIEWED BY
Jianqing Jia,
Syracuse University, United States
Rongrong Lin,
Guangdong University of Technology, China

*CORRESPONDENCE
Kathryn Linehan
kjl5t@virginia.edu

RECEIVED 20 May 2025
ACCEPTED 15 July 2025
PUBLISHED 21 August 2025

CITATION
Linehan K and Balan R (2025) CUR matrix approximation through convex optimization for feature selection.
Front. Appl. Math. Stat. 11:1632218.
doi: 10.3389/fams.2025.1632218

CUR matrix approximation through convex optimization for feature selection

Kathryn Linehan[1,2]* and Radu Balan[1]

[1]Department of Mathematics, University of Maryland, College Park, MD, United States, [2]Research Computing, University of Virginia, Charlottesville, VA, United States

The singular value decomposition (SVD) is commonly used in applications that require a low-rank matrix approximation. However, the singular vectors cannot be interpreted in terms of the original data. For applications requiring this type of interpretation, e.g., selection of important data matrix columns or rows, the approximate CUR matrix factorization can be used. Work on the CUR matrix approximation has generally focused on algorithm development, theoretical guarantees, and applications. In this study, we present a novel deterministic CUR formulation and algorithm with theoretical convergence guarantees. The algorithm utilizes convex optimization, finds important columns and rows separately, and allows the user to control the number of important columns and rows selected from the original data matrix. We present numerical results and demonstrate the effectiveness of our CUR algorithm as a feature selection method on gene expression data. These results are compared to those using the SVD and other CUR algorithms as the feature selection method. Finally, we present a novel application of CUR as a feature selection method to determine discriminant proteins when clustering protein expression data in a self-organizing map (SOM), and compare the performance of multiple CUR algorithms in this application.

KEYWORDS

CUR matrix approximation, convex optimization, low-rank matrix approximation, feature selection, interpretation

1 Introduction

Low-rank matrix approximations are common tools in many applications, including principal component analysis (PCA), signal denoising, and least squares. While the truncated singular value decomposition (SVD) is the optimal approximation in terms of matrix reconstruction (Eckart-Young theorem), the singular vectors cannot be interpreted in terms of the original data. Mahoney and Drineas [1] provided an example of this: $[\frac{1}{2}\text{age} - \frac{1}{\sqrt{2}}\text{height} + \frac{1}{2}\text{income}]$ is an eigenvector for a dataset of features about people that "is not particularly informative or meaningful." However, the approximate CUR matrix factorization can be interpreted in terms of the original data, making it an attractive low-rank approximation option, especially for applications that seek important matrix columns or rows. Several of these applications exist [2], e.g., selecting important genes from gene expression data to cluster patients by cancer type [1], and more broadly can be considered feature selection applications.

The approximate CUR factorization of $\mathbf{X} \in \mathbb{R}^{m\times n}$ is generally computed in three steps, but steps (1) and (2) can also be computed simultaneously: (1) select $c \in \mathbb{N}$ columns of $\mathbf{X}$ and let $\mathbf{C} \in \mathbb{R}^{m\times c}$ contain these columns, (2) select $r \in \mathbb{N}$ rows of $\mathbf{X}$ and let $\mathbf{R} \in \mathbb{R}^{r\times n}$

contain these rows, and (3) compute $\mathbf{U} \in \mathbb{R}^{c\times r}$, so that **CUR** is a good approximation to **X**. The result is a matrix approximation

$$\underset{m\times n}{\mathbf{X}} \approx \underset{m\times c}{\mathbf{C}}\ \underset{c\times r}{\mathbf{U}}\ \underset{r\times n}{\mathbf{R}},$$

where generally $c \ll n$ and $r \ll m$. Hence, CUR maintains the structure of the data, for example, sparsity or non-negativity, and **C** and **R** can be viewed as containing the most important columns and rows of the original data, respectively. CUR matrix approximation has been successfully used for feature selection in applications such as document clustering [1], gene expression data clustering [1, 2], image classification [3], and sensor selection and channel assignment [4]. CUR has also been used for simultaneous feature selection and sample selection for active learning [5].

Hamm and Huang [6] provided a history of CUR and mentioned that recent work on the CUR approximation most likely began with developments in the mid-to-late 1990s by Goreinov et al., e.g., [7]. Since then, several CUR algorithms have been developed; some are randomized, e.g., [1, 8, 9], and others are deterministic, e.g., [2, 10]. Work on CUR includes proving accuracy and/or other theoretical guarantees for algorithms, e.g., [1, 8], and also performance of CUR algorithms in practice without theoretical guarantees, e.g., [11]. In this study, we are particularly interested in deterministic CUR algorithms that can independently select columns of **X** without simultaneous selection of its rows due to the fact that (1) several applications exist that seek important matrix columns or rows and not a full matrix factorization [2], and (2) for a practical application, a randomized CUR will likely produce a different set of important columns and/or rows in each run of the algorithm, which may not be desirable to the scientist [11, 12].

One deterministic approach to computing a CUR approximation is to select columns and rows of **X** for inclusion in **C** and **R** using convex optimization with regularization, e.g., [11, 12]. In this study, we propose a novel CUR algorithm utilizing convex optimization with contributions in the formulation, implementation, and application of CUR. The main contributions of the study include (1) a novel convex optimization formulation for CUR, (2) an algorithm utilizing convex optimization that solves for **C** and **R** separately and allows the user to select c and r, and (3) an implementation utilizing the "surrogate functional" technique of Daubechies et al. [13], which we adapt for use with a new penalty function. We also note that our CUR algorithm and implementation can accommodate a variety of penalty functions, allowing the user a flexible framework. We provide numerical results that compare our CUR algorithm with the SVD and other deterministic CUR algorithms that select **C** and **R** separately, allowing the user to select c and r. Specifically, we show that our CUR algorithm performs very well as a feature selection method in an extension of an experiment by Sorenson and Embree [2] on gene expression data, in which important genes are selected to cluster patients into two classes - those with and without a lung tumor.

Another main contribution of the study is a novel application of CUR for feature selection. We adapt the clustering analysis of Higuera et al. [14] in which Self-Organizing Maps (SOMs)[1] and the Wilcoxon rank-sum test were used to determine proteins that critically affect learning in wild-type and trisomic (Down syndrome) mice. Specifically, we use CUR as the feature selection method instead of the Wilcoxon rank-sum test. We show that CUR can be used effectively in this application and compare the performance of our CUR algorithm to that of other deterministic CUR algorithms that select **C** and **R** separately and allow the user to select c and r. This is not only a novel application of CUR, but to the best of our knowledge, also the first use of CUR on protein expression data.

The remainder of this article is organized as follows: we present related work in Section 2, our novel CUR algorithm utilizing convex optimization in Section 3, the theoretical foundations of the algorithm in Section 4, numerical experiments in Section 5, a novel application of CUR as a feature selection method in protein expression discriminant analysis in Section 6, and a conclusion in Section 7. Throughout this study, we use MATLAB notation to denote rows and columns of matrices, e.g., row i of **X** is denoted $\mathbf{X}(i,:)$ and column j of **X** is denoted $\mathbf{X}(:,j)$. In addition, the set $\{1,2,...,n\}$ is denoted $[n]$.

2 Related work

As mentioned in Section 1, work on the CUR matrix approximation has generally focused on algorithm development, theoretical guarantees, and applications. In this section, we focus on related work in three areas: (1) deterministic CUR algorithms that solve for **C** and **R** separately and allow the user to select c and r, (2) CUR algorithms that use convex optimization with regularization to select columns and rows of the data matrix for inclusion in **C** and **R**, and (3) CUR feature selection applications. Since we mentioned a number of feature selection applications in the introduction, we will now provide more details. For the interested reader, Dong and Martinsson [15] provide a survey of CUR algorithms (including those that do not fit the criterion for inclusion in this section).

Deterministic CUR algorithms that solve for **C** and **R** separately and allow the user to select c and r include a leverage score approach [1], a discrete empirical interpolation method (DEIM) approach [2], and a pivoted QR approach [10]. The leverage score approach by Mahoney and Drineas [1] is often compared to in the CUR literature. This approximation is randomized and columns and rows are sampled based on their "normalized statistical leverage scores," which capture information on how much a column or row contributes to the optimal low-rank approximation to the data matrix, the rank-k SVD, where k is a rank parameter chosen by the user. However, a deterministic variant of this algorithm is to select the columns and rows with the largest leverage scores for inclusion in **C** and **R**, respectively. In the DEIM approach by Sorenson and Embree [2], columns are chosen for inclusion in **C** and rows are chosen for inclusion in **R** using the discrete empirical interpolation method (DEIM) on the top-k right and left singular vectors of the data matrix, respectively, where $k = c = r$. In the pivoted QR approach by Stewart [10], columns are selected for inclusion in **C** and rows are selected for inclusion in **R** using a pivoted QR factorization of **X** and $\mathbf{X}^{\mathbf{T}}$, respectively. Sorenson and Embree [2] present a slight adaptation of this approach in which rows are selected for inclusion in **R** using a pivoted QR factorization of $\mathbf{C}^{\mathbf{T}}$. In each of these four CUR algorithms, **U** is computed as

1 Also known as Kohonen Maps.

$\mathbf{U} = \mathbf{C}^{+}\mathbf{X}\mathbf{R}^{+}$, i.e., the minimizer of $\|\mathbf{X} - \mathbf{CUR}\|_F$ [10], where $\mathbf{X}^{+}$ denotes the Moore-Penrose generalized inverse or pseudoinverse of $\mathbf{X}$.

Since the CUR algorithm that we present in this study uses convex optimization with regularization to select columns and rows of the data matrix for inclusion in $\mathbf{C}$ and $\mathbf{R}$, we also note related work in this area. In 2010, Bien et al. [12] related CUR to sparse PCA and used the following convex minimization problem to find $\mathbf{C}$:

$$\mathbf{B}^* = \underset{\mathbf{B}\in\mathbb{R}^{n\times n}}{\operatorname{argmin}} \|\mathbf{X} - \mathbf{XB}\|_F + \lambda \sum_{i=1}^{n} \|\mathbf{B}(i,:)\|_2, \tag{1}$$

where $\lambda > 0$ is a regularization parameter. The indices of non-zero rows of $\mathbf{B}^*$ are the indices of columns to choose from $\mathbf{X}$ for inclusion in $\mathbf{C}$. $\mathbf{R}$ can be found using a similar optimization problem; however, the computation of $\mathbf{U}$ is not discussed. Mairal et al. [11] formulated CUR as a convex optimization problem that selects columns and rows of $\mathbf{X}$ at the same time:

$$\mathbf{W}^* = \underset{\mathbf{W}\in\mathbb{R}^{n\times m}}{\operatorname{argmin}} \|\mathbf{X} - \mathbf{XWX}\|_F^2 + \lambda_{\text{row}} \sum_{i=1}^{m} \|\mathbf{W}(:,i)\|_\infty + \lambda_{\text{col}} \sum_{j=1}^{n} \|\mathbf{W}(j,:)\|_\infty, \tag{2}$$

where λ_{row} and $\lambda_{\text{col}} > 0$ are regularization parameters. Similar to Bien et al. [12], the non-zero row indices of $\mathbf{W}^*$ are the indices of columns to select from $\mathbf{X}$, and the non-zero column indices of $\mathbf{W}^*$ are the indices of rows to select from $\mathbf{X}$. $\mathbf{U}$ is computed as $\mathbf{U} = \mathbf{C}^{+}\mathbf{X}\mathbf{R}^{+}$. In both Bien et al. [12] and Mairal et al. [11], the convex optimization CUR algorithm achieves similar matrix reconstruction accuracy to that of the leverage score-based CUR [1] in numerical experiments. In addition, Ida et al. [16] presented a method to speed up the coordinate descent algorithm for solving Equation 1 as presented in Bien et al. [12] and claimed it can be extended to solve Equation 2 as well.

In 2018, Peng et al. [17] used an optimization problem with regularization terms that simultaneously performed a CUR approximation of a network node attribute matrix (to choose representative nodes and attributes at the same time) and residual analysis to detect anomalies on attributed networks. The part of the optimization formulation related to CUR is similar to Equation 2, but uses the ℓ_2 norm rather than the ℓ_∞ norm in the regularization terms. This optimization problem was solved using alternating convex optimizations, and parameters were chosen using a grid search in experimental results. In each of the convex optimization CUR approaches mentioned above [11, 12, 16, 17], there is no built-in algorithmic control for selecting c columns and r rows of the data matrix.

Li et al. [5] used a convex optimization CUR to simultaneously select features and representative data samples to perform feature selection and active learning at the same time. The optimization problem used is similar to Equation 2 except that the regularization terms use the ℓ_2 norm rather than the ℓ_∞ norm, and there is an additional regularization term that provides "local linear reconstruction." Parameters were grid searched in experimental results and after the optimization problem is solved, the indices of the c rows of $\mathbf{W}^*$ with the largest ℓ_2 norms are the indices of columns selected from the data matrix for inclusion in $\mathbf{C}$. The indices of r rows to include in $\mathbf{R}$ are found similarly.

We provided examples of applications that use CUR for feature selection in Section 1, and here provide more details for those examples. Mahoney and Drineas [1] applied CUR to a term-document matrix and used the results to cluster the documents into two topics, with interpretation provided by selection of the five most important terms by CUR. The clustering provided by CUR outperformed that provided by the truncated SVD. They also similarly applied CUR to gene expression data to cluster patients by cancer type. Clustering peformance was equivalent to that of using the truncated SVD, but CUR provided insight into which genes are most important to the clustering, and of the 12 selected, some are known to be medically associated with cancer. Sorenson and Embree [2] also used CUR on gene expression data to discover genes that cluster patients into those with and without a tumor. They compared results using their DEIM CUR and the deterministic leverage score CUR of Mahoney and Drineas [1]. While the DEIM CUR reconstructed the original data matrix better than the deterministic leverage score CUR, the genes selected by the leverage score CUR performed much better in separating patients with and without a tumor.

Liu and Shao [3] leveraged CUR for feature selection to improve image classification accuracy. CUR performed the best of the dimensionality reduction methods used, which included PCA. Esmaeili et al. [4] used CUR for cognitive radio sensor selection and channel assignment. Specifically, sensors were chosen using the selected columns from $\mathbf{C}$, and channels were selected using the highest magnitude elements of $\mathbf{U}$ for each chosen sensor, and the resulting samples were interpolated to create the spectrum map. They tested various CUR algorithms, including the leverage score CUR [1], and showed that CUR is more effective than random uniform sampling (the prior method) in recreating the spectrum map. As mentioned earlier, Li et al. [5] used CUR for simultaneous feature selection and sample selection for active learning to classify synthetic data, gene expression data, molecular data, image and video data, and human activity recognition data. They demonstrated that their convex optimization CUR almost always outperformed other feature selection and active learning methods, including the randomized leverage score CUR [1], in terms of classifier accuracy.

3 CUR algorithm

Let $\mathbf{X} \in \mathbb{R}^{m\times n}$ be the matrix we wish to approximate as $\mathbf{X} \approx \mathbf{CUR}$. Our formulation of CUR using convex optimization builds upon ideas from Bien et al. [12], Mahoney and Drineas [1], and Mairal et al. [11]. To select a subset of columns from $\mathbf{X}$ to form the matrix $\mathbf{C}$, we solve

$$\mathbf{W}^* = \underset{\mathbf{W}\in\mathbb{R}^{n\times m}}{\operatorname{argmin}} \|\mathbf{X} - \mathbf{XWX}\|_F^2 + \lambda_C \sum_{i=1}^{n} \|\mathbf{W}(i,:)\|_\infty, \tag{3}$$

for a given $\lambda_C \in \mathbb{R} \geq 0$. Then $\mathbf{C} = \mathbf{X}(:, I_C)$, where I_C is the set of indices of non-zero rows in $\mathbf{W}^*$. Hence, λ_C controls how many columns are selected from $\mathbf{X}$, i.e., the larger the value of λ_C, the more rows of $\mathbf{W}$ are forced to $\mathbf{0}$, and the fewer columns of $\mathbf{X}$ are selected.

After $\mathbf{C}$ has been calculated, we select a set of rows from $\mathbf{X}$ to form the matrix $\mathbf{R}$ by solving

$$\mathbf{W}^* = \underset{\mathbf{W}\in\mathbb{R}^{c\times m}}{\operatorname{argmin}} \|\mathbf{X} - \mathbf{CWX}\|_F^2 + \lambda_R \sum_{j=1}^{m} \|\mathbf{W}(:,j)\|_\infty, \qquad (4)$$

for a given $\lambda_R \in \mathbb{R} \geq 0$. Then, $\mathbf{R} = \mathbf{X}(I_R, :)$, where I_R is the set of indices of non-zero columns in $\mathbf{W}^*$, and λ_R controls the number of rows of selected.

Input to our algorithm includes c and r, the number of columns and rows to be selected from $\mathbf{X}$ for $\mathbf{C}$ and $\mathbf{R}$, respectively. For a given λ_C, it is unknown in advance how many columns will be selected from $\mathbf{X}$ by the solution of Equation 3. Hence, we utilize bisection on λ_C with multiple iterates of the column selection procedure to find a selection of exactly c columns. We use a similar process with λ_R and the row selection procedure to find a selection of exactly r rows. To complete the algorithm, $\mathbf{U}$ is computed as $\mathbf{U} = \mathbf{C}^+\mathbf{X}\mathbf{R}^+$. The pseudocode for our CUR approximation is given in Algorithm 1.

The initial minimum value for λ_C in the bisection method is 0, which corresponds to potentially all columns of $\mathbf{X}$ being selected for the matrix $\mathbf{C}$. The initial maximum value for λ_C in the bisection method is the smallest value of λ_C that forces zero columns of $\mathbf{X}$ to be selected, i.e., the solution to Equation 3 to be $\mathbf{W}^* = \mathbf{0}$. We call this the critical value of λ_C and denote it λ_C^*. The range of values for λ_R in the bisection method with Equation 4 is set similarly, with the critical value of λ_R denoted as λ_R^*. To prove the exact values of λ_C^* and λ_R^*, we first provide a helpful lemma. We will also use the matrix identity $\text{vec}(\mathbf{AXB}) = (\mathbf{B}^\mathbf{T} \otimes \mathbf{A})\text{vec}(\mathbf{X})$ in which $\otimes$ denotes the Kronecker product, and $\text{vec}(\mathbf{X})$ is a column-stacked version of $\mathbf{X}$, so that $\text{vec}(\mathbf{X}) \in \mathbb{R}^{mn}$ for $\mathbf{X} \in \mathbb{R}^{m\times n}$. Using this identity, Equations 3, 4 can be reshaped as

$$\mathbf{w}^* = \min_{\mathbf{w}\in\mathbb{R}^{mn}} \|(\mathbf{X}^\mathbf{T} \otimes \mathbf{X})\mathbf{w} - \mathbf{b}\|_2^2 + \lambda_C \sum_{i=1}^{n} \max_{1\leq j\leq m} |\mathbf{w}_{i+(j-1)n}|, \qquad (5)$$

and

$$\mathbf{w}^* = \min_{\mathbf{w}\in\mathbb{R}^{mc}} \|(\mathbf{X}^\mathbf{T} \otimes \mathbf{C})\mathbf{w} - \mathbf{b}\|_2^2 + \lambda_R \sum_{j=1}^{m} \max_{1\leq i\leq c} |\mathbf{w}_{i+(j-1)n}|, \qquad (6)$$

where $\mathbf{b} = \text{vec}(\mathbf{X}) \in \mathbb{R}^{mn}$, $\mathbf{w} = \text{vec}(\mathbf{W}) \in \mathbb{R}^{mn}$ (Equation 5) or $\mathbf{w} = \text{vec}(\mathbf{W}) \in \mathbb{R}^{mc}$ (Equation 6), and $\mathbf{w}^*$ is defined similarly to $\mathbf{w}$.

Lemma 3.1. Let $\mathbf{v} \in \mathbb{R}^m$ be fixed, and $\lambda \in \mathbb{R}$. If $\langle \mathbf{v}, \mathbf{x}\rangle \leq \frac{\lambda}{2}\|\mathbf{x}\|_\infty$ $\forall \mathbf{x} \in \mathbb{R}^m$, then $\lambda \geq 2\|\mathbf{v}\|_1$.

Proof.

$$\langle \mathbf{v}, \mathbf{x}\rangle = \sum_{k=1}^{m} \mathbf{v}_k \mathbf{x}_k \leq \left(\sum_{k=1}^{m} |\mathbf{v}_k|\right) \max_{1\leq k\leq m} |\mathbf{x}_k| = \|\mathbf{v}\|_1 \|\mathbf{x}\|_\infty.$$

Hence for $\mathbf{x}_k = \text{sign}(\mathbf{v}_k)$, $\langle \mathbf{v}, \mathbf{x}\rangle = \|\mathbf{v}\|_1\|\mathbf{x}\|_\infty$ and $\lambda = 2\|\mathbf{v}\|_1$. Since the smallest value of λ, which holds $\forall \mathbf{x} \in \mathbb{R}^m$, will occur when $\langle \mathbf{v}, \mathbf{x}\rangle = \frac{\lambda}{2}\|\mathbf{x}\|_\infty$, it is the case that $\forall \mathbf{x} \in \mathbb{R}^m$, $\lambda \geq 2\|\mathbf{v}\|_1$.

Theorem 3.2. Let $\mathbf{M} = \textit{reshape}((\mathbf{X}^\mathbf{T} \otimes \mathbf{X})^\mathbf{T}\mathbf{b}, n, m)$, i.e., $(\mathbf{X}^\mathbf{T} \otimes \mathbf{X})^\mathbf{T}\mathbf{b}$ reshaped from $\mathbb{R}^{mn}$ to $\mathbb{R}^{n\times m}$ so that $(\mathbf{X}^\mathbf{T} \otimes \mathbf{X})^\mathbf{T}\mathbf{b} = \text{vec}(\mathbf{M})$. Then,

$$\lambda_C^* = 2 \max_{1\leq i\leq n} \|\mathbf{M}(i,:)\|_1 = 2\|\mathbf{M}\|_\infty.$$

Similarly, let $\mathbf{N} = \textit{reshape}((\mathbf{X}^\mathbf{T} \otimes \mathbf{C})^\mathbf{T}\mathbf{b}, c, m)$. Then,

$$\lambda_R^* = 2 \max_{1\leq j\leq m} \|\mathbf{N}(:,j)\|_1 = 2\|\mathbf{N}\|_1.$$

Proof. Let $\mathbf{A} = (\mathbf{X}^\mathbf{T} \otimes \mathbf{X})$ and the objective function of Equation 5 be $J(\mathbf{w})$:

$$J(\mathbf{w}) = \|\mathbf{Aw} - \mathbf{b}\|_2^2 + \lambda_C \sum_{i=1}^{n} \max_{1\leq j\leq m} |\mathbf{w}_{i+(j-1)n}|.$$

We want to find the smallest $\lambda_C^* > 0$ such that $\forall \lambda \geq \lambda_C^*$, $\text{argmin}_{\mathbf{w}\in\mathbb{R}^{mn}} J(\mathbf{w}) = \mathbf{0}$. We have

$$\begin{aligned}
J(\mathbf{w}) &= \|\mathbf{Aw}\|_2^2 - 2\mathbf{w}^\mathbf{T}\mathbf{A}^\mathbf{T}\mathbf{b} + \|\mathbf{b}\|_2^2 + \lambda_C \sum_{i=1}^{n} \max_{1\leq j\leq m} |\mathbf{w}_{i+(j-1)n}| \\
&= \|\mathbf{Aw}\|_2^2 + \|\mathbf{b}\|_2^2 + \lambda_C \sum_{i=1}^{n} \max_{1\leq j\leq m} |\mathbf{w}_{i+(j-1)n}| - 2\sum_{k=1}^{mn} (\mathbf{A}^\mathbf{T}\mathbf{b})_k \mathbf{w}_k \\
&= \|\mathbf{Aw}\|_2^2 + \|\mathbf{b}\|_2^2 + \lambda_C \sum_{i=1}^{n} \max_{1\leq j\leq m} |\mathbf{W}_{ij}| - 2\sum_{i=1}^{n}\sum_{j=1}^{m} \mathbf{M}_{ij}\mathbf{W}_{ij},
\end{aligned}$$

where $\mathbf{W} = \text{reshape}(\mathbf{w}, n, m)$. If $\forall i$,

$$\frac{\lambda_C}{2} \max_{1\leq j\leq m} |\mathbf{W}_{ij}| - \sum_{j=1}^{m} \mathbf{M}_{ij}\mathbf{W}_{ij} \geq 0, \qquad (7)$$

then

$$\sum_{i=1}^{n} \left[\frac{\lambda_C}{2} \max_{1\leq j\leq m} |\mathbf{W}_{ij}| - \sum_{j=1}^{m} \mathbf{M}_{ij}\mathbf{W}_{ij}\right] \geq 0,$$

and

$$\lambda_C \sum_{i=1}^{n} \max_{1\leq j\leq m} |\mathbf{W}_{ij}| - 2\sum_{i=1}^{n}\sum_{j=1}^{m} \mathbf{M}_{ij}\mathbf{W}_{ij} \geq 0.$$

Hence, $J(\mathbf{w}) \geq J(\mathbf{0}) = \|\mathbf{b}\|_2^2$, for all $\mathbf{w}$, thus $\text{argmin}_{\mathbf{w}\in\mathbb{R}^{mn}} J(\mathbf{w}) = \mathbf{0}$. By Lemma 3.1, assuming Equation 7 is true, $\forall i$, $\lambda_C \geq 2\|\mathbf{M}(i,:)\|_1$. Thus, $\lambda_C^* = 2\max_{1\leq i\leq n} \|\mathbf{M}(i,:)\|_1 = 2\|\mathbf{M}\|_\infty$.

A similar proof can be used to show the result for λ_R^*, letting $\mathbf{A} = (\mathbf{X}^\mathbf{T} \otimes \mathbf{C})$ and $J(\mathbf{w})$ be the objective function of Equation 6:

$$J(\mathbf{w}) = \|\mathbf{Aw} - \mathbf{b}\|_2^2 + \lambda_R \sum_{j=1}^{m} \max_{1\leq i\leq c} |\mathbf{w}_{i+(j-1)n}|.$$

By Theorem 3.2 and the fact that $\mathbf{b} = \text{vec}(\mathbf{X})$, we can calculate

$$\begin{aligned}
\lambda_C^* &= 2\|\text{reshape}((\mathbf{X}^\mathbf{T} \otimes \mathbf{X})^\mathbf{T}\text{vec}(\mathbf{X}), n, m)\|_\infty \\
&= 2\|\text{reshape}((\mathbf{X} \otimes \mathbf{X}^\mathbf{T})\text{vec}(\mathbf{X}), n, m)\|_\infty \\
&= 2\|\text{reshape}(\text{vec}(\mathbf{X}^\mathbf{T}\mathbf{X}\mathbf{X}^\mathbf{T}), n, m)\|_\infty \\
&= 2\|\mathbf{X}^\mathbf{T}\mathbf{X}\mathbf{X}^\mathbf{T}\|_\infty,
\end{aligned}$$

where the third line follows from the same matrix identity used above. A similar calculation shows that $\lambda_R^* = 2\|\mathbf{C}^\mathbf{T}\mathbf{X}\mathbf{X}^\mathbf{T}\|_1$.

```
Input: $\mathbf{X} \in \mathbb{R}^{m\times n}$, $c \in \mathbb{N}$, $r \in \mathbb{N}$
Output: $\mathbf{C} \in \mathbb{R}^{m\times c}$, $\mathbf{U} \in \mathbb{R}^{c\times r}$, $\mathbf{R} \in \mathbb{R}^{r\times n}$ such that
    $\mathbf{X} \approx \mathbf{CUR}$
 1: $\lambda_C^* = 2\|\mathbf{X^TXX^T}\|_\infty$
 2: $n_c = 0$, $\lambda_{\min} = 0$, $\lambda_{\max} = \lambda_C^*$
 3: while $n_c \neq c$ do
 4:     $\lambda_C = (\lambda_{\max} + \lambda_{\min})/2$
 5:     solve $\mathbf{W}^* = \text{argmin}_{\mathbf{W}\in\mathbb{R}^{n\times m}} \|\mathbf{X} - \mathbf{XWX}\|_F^2 + \lambda_C \sum_{i=1}^n \|\mathbf{W}(i,:)\|_\infty$
 6:     let $I_C$ be the set of indices of nonzero rows of $\mathbf{W}^*$
 7:     $n_c = |I_C|$
 8:     if $c > n_c$ then
 9:         $\lambda_{\min} = \lambda_C$
10:     else if $c < n_c$ then
11:         $\lambda_{\max} = \lambda_C$
12:     end if
13: end while
14: $\mathbf{C} = \mathbf{X}(:, I_C)$
15: $\lambda_R^* = 2\|\mathbf{C^TXX^T}\|_1$
16: $n_r = 0$, $\lambda_{\min} = 0$, $\lambda_{\max} = \lambda_R^*$
17: while $n_r \neq r$ do
18:     $\lambda_R = (\lambda_{\max} + \lambda_{\min})/2$
19:     solve $\mathbf{W}^* = \text{argmin}_{\mathbf{W}\in\mathbb{R}^{c\times m}} \|\mathbf{X} - \mathbf{CWX}\|_F^2 + \lambda_R \sum_{j=1}^m \|\mathbf{W}(:,j)\|_\infty$
20:     let $I_R$ be the set of indices of nonzero columns of $\mathbf{W}^*$
21:     $n_r = |I_R|$
22:     if $r > n_r$ then
23:         $\lambda_{\min} = \lambda_R$
24:     else if $r < n_r$ then
25:         $\lambda_{\max} = \lambda_R$
26:     end if
27: end while
28: $\mathbf{R} = \mathbf{X}(I_R, :)$
29: $\mathbf{U} = \mathbf{C}^+\mathbf{X}\mathbf{R}^+$
30: return $\mathbf{C}, \mathbf{U}, \mathbf{R}$
```

Algorithm 1. CUR through convex optimization.

3.1 Implementation for minimization problems

For $\mathbf{X}$ of small[2] dimensions, we can solve the minimization problems on lines 5 and 19 of Algorithm 1 using the reshaped versions (Equations 5, 6) and a convex programming solver, such as the CVX package in MATLAB [18, 19]. However, using a solver for $\mathbf{X}$ of larger dimensions becomes infeasible due to the use of the Kronecker product in the reshaped problems. For example, to store the genetics dataset referenced in Section 5, a dense double array $\mathbf{X} \in \mathbb{R}^{107\times 22{,}283}$, 19.07 MB is used; to store $\mathbf{X^T} \otimes \mathbf{X} \in \mathbb{R}^{2{,}384{,}281\times 2{,}384{,}281}$, 45.48 TB is used.

To accommodate large-scale problems, we solve these minimizations in Algorithm 1 using an extension of an iterative method by Daubechies et al. [13] that utilizes a "surrogate functional" to solve regularized least squares minimization problems in which weighted ℓ_p-norm penalty functions are used, for $1 \leq p \leq 2$. This technique decouples a large minimization problem into smaller, easy-to-solve problems. We extend the results of Daubechies et al. [13] to apply to our penalty functions, e.g., $\sum_{i=1}^n \|\mathbf{W}(i,:)\|_\infty$. We demonstrate the method for the line 5 minimization in Algorithm 1. The line 19 minimization is handled similarly.

Let the objective function of Equation 3 be denoted by

$$J(\mathbf{W}) = \|\mathbf{X} - \mathbf{XWX}\|_F^2 + \lambda_C \sum_{i=1}^n \|\mathbf{W}(i,:)\|_\infty,$$

and the corresponding surrogate functional by

$$\widehat{J}(\mathbf{W},\mathbf{Z}) = \|\mathbf{X} - \mathbf{XWX}\|_F^2 + \lambda_C \sum_{i=1}^n \|\mathbf{W}(i,:)\|_\infty + \mu\|\mathbf{W} - \mathbf{Z}\|_F^2 - \|\mathbf{XWX} - \mathbf{XZX}\|_F^2,$$

where $\mathbf{Z} \in \mathbb{R}^{n\times m}$ and $\mu > 0$. For any $\mathbf{Z} \in \mathbb{R}^{n\times m}$ and $\mu > 0$, we have

$$\begin{aligned}\widehat{J}(\mathbf{W},\mathbf{Z}) &= \mu\|\mathbf{W}\|_F^2 - 2\,\mathrm{tr}\{\mathbf{W}(\mu\mathbf{Z^T} + \mathbf{XX^TX} - \mathbf{XX^TZ^TX^TX})\} \\ &\quad + \lambda_C \sum_{i=1}^n \|\mathbf{W}(i,:)\|_\infty + \|\mathbf{X}\|_F^2 + \mu\|\mathbf{Z}\|_F^2 - \|\mathbf{XZX}\|_F^2 \\ &= \sum_{i=1}^n \left[\mu\|\mathbf{W}(i,:)\|_2^2 - 2\langle \mathbf{W}(i,:), \mathbf{L^T}(i,:)\rangle + \lambda_C\|\mathbf{W}(i,:)\|_\infty\right] \\ &\quad + \|\mathbf{X}\|_F^2 + \mu\|\mathbf{Z}\|_F^2 - \|\mathbf{XZX}\|_F^2,\end{aligned}$$

where $\mathbf{L} = \mu\mathbf{Z^T} + \mathbf{XX^TX} - \mathbf{XX^TZ^TX^TX}$. Hence,

$$\begin{aligned}\underset{\mathbf{W}\in\mathbb{R}^{n\times m}}{\text{argmin}}\,\widehat{J}(\mathbf{W},\mathbf{Z}) &= \underset{\mathbf{W}\in\mathbb{R}^{n\times m}}{\text{argmin}}\,\mu \sum_{i=1}^n \Big[\|\mathbf{W}(i,:) - \frac{1}{\mu}\mathbf{L^T}(i,:)\|_2^2 \\ &\quad - \|\frac{1}{\mu}\mathbf{L^T}(i,:)\|_2^2 + \frac{\lambda_C}{\mu}\|\mathbf{W}(i,:)\|_\infty\Big] \\ &= \underset{\mathbf{W}\in\mathbb{R}^{n\times m}}{\text{argmin}}\,2\mu \sum_{i=1}^n \Big[\frac{1}{2}\|\mathbf{W}(i,:) - \frac{1}{\mu}\mathbf{L^T}(i,:)\|_2^2 \\ &\quad + \frac{\lambda_C}{2\mu}\|\mathbf{W}(i,:)\|_\infty\Big].\end{aligned}$$

Thus, we can easily minimize $\widehat{J}$ over $\mathbf{W}$ by computing the proximal operator of the ℓ_∞ norm,

$$\mathbf{prox}_{\alpha\|\cdot\|_\infty}(\mathbf{x}) = \underset{\mathbf{y}\in\mathbb{R}^m}{\text{argmin}} \left(\frac{1}{2}\|\mathbf{y} - \mathbf{x}\|_2^2 + \alpha\|\mathbf{y}\|_\infty\right) \tag{8}$$

for $\mathbf{x} \in \mathbb{R}^m$ and $\alpha \geq 0$, for each row of $\mathbf{W}$. To find a minimizer of J, we utilize the minimization of $\widehat{J}$ in the iterative process in Algorithm 2. In Section 4, we will show that $\mathbf{W}^* = \text{argmin}_{\mathbf{W}\in\mathbb{R}^{n\times m}} J(\mathbf{W})$, where $\mathbf{W}^*$ is the output of Algorithm 2.

3.2 Complexity

For this analysis, we will assume that $r \ll m$ and $c \ll n$, and rename μ as μ_C to avoid confusion with μ_R, a similar

2 Small is relative to the user's computer memory size.

```
Input: $\mathbf{X} \in \mathbb{R}^{m\times n}$, $\lambda_C \geq 0$, $\mu > \|\mathbf{X}\|_2^4$
Output: $\mathbf{W}^* \in \mathbb{R}^{n\times m}$ s.t. $\mathbf{W}^* = \text{argmin}_{\mathbf{W}\in\mathbb{R}^{n\times m}} \|\mathbf{X} - \mathbf{XWX}\|_F^2 + \lambda_C \sum_{i=1}^{n} \|\mathbf{W}(i,:)\|_\infty$
 1: $\mathbf{W}^0 = \mathbf{0}$
 2: $k = 0$
 3: repeat                ▷ each iteration solves $\mathbf{W}^\mathbf{k} = \text{argmin}_{\mathbf{Y}\in\mathbb{R}^{n\times m}} \widehat{J}(\mathbf{Y}, \mathbf{W}^{\mathbf{k-1}})$.
 4:     $k = k + 1$
 5:     $\mathbf{L} = \mu(\mathbf{W}^{\mathbf{k-1}})^\mathbf{T} + \mathbf{XX}^\mathbf{T}\mathbf{X} - \mathbf{XX}^\mathbf{T}(\mathbf{W}^{\mathbf{k-1}})^\mathbf{T}\mathbf{X}^\mathbf{T}\mathbf{X}$
 6:     for $i = 1$ to $n$ do
 7:         $\mathbf{W}^\mathbf{k}(i,:) = \text{argmin}_{\mathbf{y}\in\mathbb{R}^m} \left[\frac{1}{2}\|\mathbf{y} - \frac{1}{\mu}\mathbf{L}^\mathbf{T}(i,:)\|_2^2 + \frac{\lambda_C}{2\mu}\|\mathbf{y}\|_\infty\right]$
 8:     end for
 9: until convergence of the sequence $\{\mathbf{W}^\mathbf{k}\}$
10: $\mathbf{W}^* = \mathbf{W}^\mathbf{k}$        ▷ $\mathbf{W}^* = \lim_{k\to\infty}\{\mathbf{W}^\mathbf{k}\}$
11: return $\mathbf{W}^*$
```

Algorithm 2. Convex optimization using the surrogate functional.

TABLE 1 Computational complexities of quantities used in Algorithm 1.

Computation	Complexity
λ_C^*	$O(mn^2)$ if $m \geq n$ or $O(m^2n)$ if $m < n$
$\mathbf{XX}^\mathbf{T}\mathbf{X}$	$O(mn)$ utilizing λ_C^* computation
$\mathbf{XX}^\mathbf{T}$	$O(m^2n)$
$\mathbf{X}^\mathbf{T}\mathbf{X}$	$O(mn^2)$
μ_C	$O(mn)$
λ_R^*	$O(cmn)$ if $m \geq c$ or $O(cm^2)$ (utilizing $\mathbf{XX}^\mathbf{T}$ computation) if $m < c$
$\mathbf{XX}^\mathbf{T}\mathbf{C}$	$O(cm)$ utilizing λ_R^* computation
$\mathbf{C}^\mathbf{T}\mathbf{C}$	$O(c^2m)$
μ_R	$O(cm)$

parameter needed to solve the line 19 minimization. We provide computational complexities in Table 1 that help determine the overall complexity of Algorithm 1. Quantities that are used in each iteration of one of the bisection loops should be computed once before the loop begins. For the first bisection loop on lines 3-13, this includes $\mathbf{XX}^\mathbf{T}\mathbf{X}$ (which is the transpose of the matrix computed to find λ_C^* in line 1), $\mathbf{XX}^\mathbf{T}$, $\mathbf{X}^\mathbf{T}\mathbf{X}$, and $\mu_C > \|\mathbf{X}\|_2^4$, which is an input to Algorithm 2. For the second bisection loop on lines 17-27, this includes $\mathbf{XX}^\mathbf{T}\mathbf{C}$ (which is the transpose of the matrix computed to find λ_R^* in line 15), $\mathbf{XX}^\mathbf{T}$ (which was already computed before the first bisection loop), $\mathbf{C}^\mathbf{T}\mathbf{C}$, and $\mu_R > \|\mathbf{X}\|_2^2\|\mathbf{C}\|_2^2$ (an input to the algorithm that solves the line 19 minimization), in which $\|\mathbf{X}\|_2$ was already computed as well.

Before we analyze the entirety of Algorithm 1, we will analyze the complexity of Algorithm 2, which solves the minimization problem on line 5 of Algorithm 1. In each iteration, the most expensive steps are the computation of $\mathbf{L}$ and the n proximal operators. $\mathbf{L}$ can be computed in $O(mn(m+n))$ time and each proximal operator can be computed in $O(m\log m)$ time. Determining if the sequence has converged in line 9 can be completed in $O(mn)$ time. Hence, the total time for Algorithm 2, assuming k iterations are completed, is $O(kmn(m+n))$. A similar analysis shows that the complexity of the algorithm for solving the minimization problem on line 19 of Algorithm 1 is $O(\hat{k}cm(m+c))$, assuming $\hat{k}$ iterations are completed. For each minimization problem, we have found that using a small maximum number of iterations in practice, e.g., 20, is sufficient for use in the CUR algorithm (Algorithm 1). For the remainder of this analysis, we will assume that the number of iterations for each minimization problem is a small constant.

In Algorithm 1, the bisection loop in lines 3–13 dominates the computational complexity. This loop runs in $O(\ell mn(m+n))$ time, assuming ℓ iterations. This is due to the computation time of Algorithm 2 and the fact that finding the set I_C can be completed in $O(mn)$ time. Using a similar analysis, the second bisection loop on lines 17–27 is an order of magnitude less expensive, i.e., $O(\hat{\ell}cm(m+c))$ assuming $\hat{\ell}$ iterations. The computation of $\mathbf{U}$ involves two pseudoinverses, $\mathbf{C}^+$ and $\mathbf{R}^+$, which can be computed in $O(cm\min(c,m))$ and $O(nr\min(n,r))$ time, respectively. The product $\mathbf{U} = \mathbf{C}^+\mathbf{XR}^+$ can be computed in $O(cn(m+r))$ time if $m \geq n$, or $O(mr(n+c))$ time if $m < n$. Hence, the total computational complexity for Algorithm 1 is $O(\ell mn(m+n))$.

3.3 Generalizations

Domain expertise can be incorporated into Equations 3, 4 using fixed relative column/row weights, e.g.,

$$\min_{\mathbf{W}\in\mathbb{R}^{n\times m}} \|\mathbf{X} - \mathbf{XWX}\|_F^2 + \lambda_C \sum_{i=1}^{n} \omega_i \|\mathbf{W}(i,:)\|_\infty,$$

where ω_i is the provided expert weight for $\mathbf{X}(:,i)$. The column/row weights reflect the relative importance of each column/row according to the expert. The implementation and theory provided in this manuscript require very minor modifications to apply to this generalization.

To form the matrix $\mathbf{C}$, our algorithm and implementation can also accommodate objective functions of the form

$$\|\mathbf{X} - \mathbf{XWX}\|_F^2 + \lambda_C \sum_{i=1}^{n} \|\mathbf{W}(i,:)\|_p^p,$$

for $1 \leq p \leq 2$. The theory for using the surrogate functional technique with these choices of objective functions is already complete [13], and the only change to the implementation detailed above is that the proximal operator in Algorithm 2 would be of the ℓ_p-norm. Closed-form solutions for the proximal operator of the ℓ_1 and ℓ_2 norms exist, making these easy choices to implement. Similar penalty function adaptations can be made to the objective function used to form the matrix $\mathbf{R}$. Hence, our algorithm and implementation provide a CUR framework. We also note that the objective function could be generalized as

$$\min_{\mathbf{W}\in\mathbb{R}^{n\times m}} \|\mathbf{X} - \mathbf{XWX}\|_p^p + \lambda_C \sum_{i=1}^{n} \|\mathbf{W}(i,:)\|_\infty,$$

where $p \in [1,\infty]$, which remains a potential area for future work.

4 Theoretical foundation

In this section, we follow the theoretical approach of Daubechies et al. [13] to prove the correctness of Algorithm 2 for solving the minimization on line 5 of Algorithm 1, i.e., that

$$\mathbf{W}^* = \underset{\mathbf{W}\in\mathbb{R}^{n\times m}}{\operatorname{argmin}} \|\mathbf{X}-\mathbf{XWX}\|_F^2 + \lambda_C \sum_{i=1}^{n} \|\mathbf{W}(i,:)\|_\infty,$$

where $\mathbf{W}^*$ is the output of Algorithm 2. The correctness of the algorithm for solving the minimization on line 19 of Algorithm 1 can be proved similarly.

For constant $\mathbf{X}, \mu, \lambda_C$, let the non-linear operator $\mathbf{T}_{\mathbf{X},\mu,\lambda_C}: \mathbb{R}^{n\times m} \to \mathbb{R}^{n\times m}$ be defined as $\mathbf{Z} \mapsto \mathbf{T}_{\mathbf{X},\mu,\lambda_C}(\mathbf{Z}) = \mathbf{W}$ and given by:

1. construct $\mathbf{L}(\mathbf{Z}) = \mu\mathbf{Z}^T + \mathbf{XX}^T\mathbf{X} - \mathbf{XX}^T\mathbf{Z}^T\mathbf{X}^T\mathbf{X}$,
2. for each row of $\mathbf{L}^T$ (i.e., $\forall i \in [n]$), solve

$$\mathbf{W}(i,:) = \underset{\mathbf{y}\in\mathbb{R}^m}{\operatorname{argmin}} \left[\frac{1}{2}\|\mathbf{y} - \frac{1}{\mu}\mathbf{L}^T(i,:)\|_2^2 + \frac{\lambda_C}{2\mu}\|\mathbf{y}\|_\infty\right],$$

and
3. reassemble $\mathbf{W}$ from its rows,

so that we can write each $\mathbf{W}^k$ produced in Algorithm 2 as $\mathbf{W}^k = \mathbf{T}^k_{\mathbf{X},\mu,\lambda_C}(\mathbf{W}^0)$. The primary result that will we prove in this section is that the sequence $\{\mathbf{W}^k\}_{k\in\mathbb{N}}$ converges to a minimizer of $\|\mathbf{X}-\mathbf{XWX}\|_F^2 + \lambda_C\sum_{i=1}^n \|\mathbf{W}(i,:)\|_\infty$. This is formally stated below.

Theorem 4.1. Let $\lambda_C \in \mathbb{R} \geq 0$; $\mu \in \mathbb{R} > 0$; $\mathbf{X} \in \mathbb{R}^{m\times n}$; and $\mathbf{W}, \mathbf{Z} \in \mathbb{R}^{n\times m}$. Define

$$J(\mathbf{W}) = \|\mathbf{X}-\mathbf{XWX}\|_F^2 + \lambda_C \sum_{i=1}^{n} \|\mathbf{W}(i,:)\|_\infty,$$

and the nonlinear operator $\mathbf{T}_{\mathbf{X},\mu,\lambda_C}: \mathbb{R}^{n\times m} \to \mathbb{R}^{n\times m}$ as $\mathbf{Z} \mapsto \mathbf{T}_{\mathbf{X},\mu,\lambda_C}(\mathbf{Z}) = \mathbf{W}$ as above.

a. If $\mu > \|\mathbf{X}\|_2^4$, then the sequence $\{\mathbf{W}^k = \mathbf{T}^k_{\mathbf{X},\mu,\lambda_C}(\mathbf{W}^0)\}_{k\in\mathbb{N}}$ produced by Algorithm 2 converges to a fixed point of $\mathbf{T}_{\mathbf{X},\mu,\lambda_C}$.
b. A fixed point of $\mathbf{T}_{\mathbf{X},\mu,\lambda_C}$ is a minimizer of $J(\mathbf{W})$.
c. $J(\mathbf{W})$ has a unique minimizer if $\mathbf{X}$ is square and full rank.

To condense notation for the remainder of this section, we write $\mathbf{T}_{\mathbf{X},\mu,\lambda_C} = \mathbf{T}$.

4.1 Convergence to a fixed point of T

We first provide six lemmas to assist in the proof of Theorem 4.1, part a.

Lemma 4.2. $(\mathbf{X}^T\mathbf{X}) \otimes (\mathbf{XX}^T)$ is symmetric and positive semidefinite.

Proof. $(\mathbf{X}^T\mathbf{X}) \otimes (\mathbf{XX}^T)$ is clearly symmetric. We will show that $\langle((\mathbf{X}^T\mathbf{X}) \otimes (\mathbf{XX}^T))\mathbf{v}, \mathbf{v}\rangle \geq 0$, $\forall \mathbf{v} \in \mathbb{R}^{mn}$, thus proving the lemma. Fix $\mathbf{v} \in \mathbb{R}^{mn}$ and $\mathbf{V} \in \mathbb{R}^{m\times n}$, such that $\operatorname{vec}(\mathbf{V}) = \mathbf{v}$.

$$\begin{aligned}\langle((\mathbf{X}^T\mathbf{X}) \otimes (\mathbf{XX}^T))\mathbf{v}, \mathbf{v}\rangle &= \langle\operatorname{vec}((\mathbf{XX}^T)\mathbf{V}(\mathbf{X}^T\mathbf{X})), \operatorname{vec}(\mathbf{V})\rangle\\ &= \langle(\mathbf{XX}^T)\mathbf{V}(\mathbf{X}^T\mathbf{X}), \mathbf{V}\rangle_F\\ &= \operatorname{tr}\{(\mathbf{XX}^T)\mathbf{V}(\mathbf{X}^T\mathbf{X})\mathbf{V}^T\}\\ &= \operatorname{tr}\{(\mathbf{XX}^T)^{\frac{1}{2}}\mathbf{V}(\mathbf{X}^T\mathbf{X})^{\frac{1}{2}}(\mathbf{X}^T\mathbf{X})^{\frac{1}{2}}\mathbf{V}^T(\mathbf{XX}^T)^{\frac{1}{2}}\}\\ &= \|(\mathbf{XX}^T)^{\frac{1}{2}}\mathbf{V}(\mathbf{X}^T\mathbf{X})^{\frac{1}{2}}\|_F^2 \geq 0,\end{aligned}$$

where $\langle.,.\rangle_F$ is the Frobenius inner product.

Lemma 4.3. $\|(\mathbf{X}^T\mathbf{X}) \otimes (\mathbf{XX}^T)\|_2 \leq (\sigma_{\max}(\mathbf{X}))^4 = \|\mathbf{X}\|_2^4$, where $\sigma_{\max}(\mathbf{X})$ is the largest singular value of the matrix $\mathbf{X}$.

Proof. By Lemma 4.2, the matrix $(\mathbf{X}^T\mathbf{X}) \otimes (\mathbf{XX}^T)$ is symmetric and positive semidefinite. Thus, the eigenvalues and singular values of $(\mathbf{X}^T\mathbf{X}) \otimes (\mathbf{XX}^T)$ are the same. Hence,

$$\begin{aligned}\|(\mathbf{X}^T\mathbf{X}) \otimes (\mathbf{XX}^T)\|_2 &= \sigma_{\max}((\mathbf{X}^T\mathbf{X}) \otimes (\mathbf{XX}^T))\\ &= \max_{\|\mathbf{v}\|_2=1} \langle((\mathbf{X}^T\mathbf{X}) \otimes (\mathbf{XX}^T))\mathbf{v}, \mathbf{v}\rangle\\ &= \max_{\|\mathbf{V}\|_F=1} \|(\mathbf{XX}^T)^{\frac{1}{2}}\mathbf{V}(\mathbf{X}^T\mathbf{X})^{\frac{1}{2}}\|_F^2,\end{aligned}$$

where the last equation follows from the proof of Lemma 4.2.

$$\begin{aligned}\|(\mathbf{XX}^T)^{\frac{1}{2}}\mathbf{V}(\mathbf{X}^T\mathbf{X})^{\frac{1}{2}}\|_F &\leq \|(\mathbf{XX}^T)^{\frac{1}{2}}\|_2\|\mathbf{V}(\mathbf{X}^T\mathbf{X})^{\frac{1}{2}}\|_F\\ &= \|(\mathbf{XX}^T)^{\frac{1}{2}}\|_2\|(\mathbf{X}^T\mathbf{X})^{\frac{1}{2}}\mathbf{V}^T\|_F\\ &\leq \|(\mathbf{XX}^T)^{\frac{1}{2}}\|_2\|(\mathbf{X}^T\mathbf{X})^{\frac{1}{2}}\|_2\|\mathbf{V}^T\|_F\\ &\leq \sigma_{\max}((\mathbf{XX}^T)^{\frac{1}{2}})\sigma_{\max}((\mathbf{X}^T\mathbf{X})^{\frac{1}{2}})\|\mathbf{V}^T\|_F\\ &= (\sigma_{\max}(\mathbf{X}))^2\|\mathbf{V}^T\|_F\end{aligned}$$

Thus,

$$\max_{\|\mathbf{V}\|_F=1} \|(\mathbf{XX}^T)^{\frac{1}{2}}\mathbf{V}(\mathbf{X}^T\mathbf{X})^{\frac{1}{2}}\|_F^2 \leq (\sigma_{\max}(\mathbf{X}))^4,$$

proving the result.

Lemma 4.4. Let $\mu \geq \|\mathbf{X}\|_2^4$ and define the operator $\mathscr{L}: \mathbf{U} \mapsto \mu\mathbf{U} - \mathbf{XX}^T\mathbf{UX}^T\mathbf{X}$. Then, $\|\mathscr{L}\| \leq \mu$.

Proof. The operator norm of $\mathscr{L}$ is

$$\begin{aligned}\|\mathscr{L}\| &= \max_{\|\mathbf{U}\|_F=1} \|\mathscr{L}(\mathbf{U})\|_F\\ &= \max_{\|\mathbf{U}\|_F=1} \|\operatorname{vec}(\mu\mathbf{U} - \mathbf{XX}^T\mathbf{UX}^T\mathbf{X})\|_2\\ &= \max_{\|\mathbf{U}\|_F=1} \|(\mu\mathbf{I} - (\mathbf{X}^T\mathbf{X}) \otimes (\mathbf{XX}^T))\operatorname{vec}(\mathbf{U})\|_2\\ &= \sigma_{\max}(\mu\mathbf{I} - (\mathbf{X}^T\mathbf{X}) \otimes (\mathbf{XX}^T)).\end{aligned}$$

By Lemmas 4.2 and 4.3, $(\mathbf{X}^T\mathbf{X}) \otimes (\mathbf{XX}^T)$ is a symmetric positive semidefinite matrix with 2-norm bounded above by $\|\mathbf{X}\|_2^4$. Hence,

$$\|\mathscr{L}\| = \sigma_{\max}(\mu\mathbf{I} - (\mathbf{X}^T\mathbf{X}) \otimes (\mathbf{XX}^T)) \leq \mu.$$

Lemma 4.5. Let $\mu \geq \|\mathbf{X}\|_2^4$. Then, the mapping $\mathbf{T}$ is non-expansive, i.e., $\forall\, \mathbf{Z}, \mathbf{Z}' \in \mathbb{R}^{n\times m}$,

$$\|\mathbf{T}(\mathbf{Z}) - \mathbf{T}(\mathbf{Z}')\|_F \leq \|\mathbf{Z} - \mathbf{Z}'\|_F.$$

Proof. By the fact that $\mathbf{prox}_{\frac{\lambda}{2\mu}\|\cdot\|_\infty}$ is non-expansive [20] and Lemma 4.4, we have

$$\begin{aligned}\|\mathbf{T}(\mathbf{Z}) - \mathbf{T}(\mathbf{Z}')\|_F^2 &= \sum_{i=1}^{n} \Bigg\| \mathbf{prox}_{\frac{\lambda}{2\mu}\|\cdot\|_\infty}\left(\frac{1}{\mu}[\mathbf{L}(\mathbf{Z})]^{\mathrm{T}}(i,:)\right) \\ &\quad - \mathbf{prox}_{\frac{\lambda}{2\mu}\|\cdot\|_\infty}\left(\frac{1}{\mu}[\mathbf{L}(\mathbf{Z}')]^{\mathrm{T}}(i,:)\right)\Bigg\|_2^2 \\ &\leq \sum_{i=1}^{n} \left\| \frac{1}{\mu}[\mathbf{L}(\mathbf{Z})]^{\mathrm{T}}(i,:) - \frac{1}{\mu}[\mathbf{L}(\mathbf{Z}')]^{\mathrm{T}}(i,:)\right\|_2^2 \\ &= \frac{1}{\mu^2}\|\mathbf{L}(\mathbf{Z}) - \mathbf{L}(\mathbf{Z}')\|_F^2 \\ &= \frac{1}{\mu^2}\|\mathscr{L}(\mathbf{Z} - \mathbf{Z}')\|_F^2 \\ &\leq \frac{1}{\mu^2}\|\mathscr{L}\|^2\|\mathbf{Z} - \mathbf{Z}'\|_F^2 \\ &\leq \|\mathbf{Z} - \mathbf{Z}'\|_F^2.\end{aligned}$$

We omit proofs of the next two lemmas, as they largely mirror those of Daubechies et al. [13].

Lemma 4.6. Let $\mu > \|\mathbf{X}\|_2^4$. Then, the mapping $\mathbf{T}$ is asymptotically regular, i.e., $\forall\, \mathbf{Z} \in \mathbb{R}^{n\times m}$,

$$\lim_{k\to\infty} \|\mathbf{T}^{k+1}(\mathbf{Z}) - \mathbf{T}^k(\mathbf{Z})\|_F = 0.$$

Lemma 4.7. Let the mapping $\mathbf{T}: \mathbb{R}^{n\times m} \to \mathbb{R}^{n\times m}$ be non-expansive and asymptotically regular. Then, the sequence $\{\mathbf{W}^k = \mathbf{T}^k(\mathbf{W}^0)\}_{k\in\mathbb{N}}$ converges to a fixed point of $\mathbf{T}$.

We are now ready to prove Theorem 4.1, part a.

Proof of Theorem 4.1, part a. By Lemmas 4.5 and 4.6, $\mathbf{T}$ is non-expansive and asymptotically regular. By Lemma 4.7, the result is proven.

4.2 Convergence to a minimizer of J

We now focus on proving Theorem 4.1, part b, and begin by establishing three lemmas.

Lemma 4.8. Let $f: I \to \mathbb{R}$ be convex, with $I = (a, b)$. If $f(x) = f_1(x) + \alpha x^2$, where $\alpha > 0$ and $f_1(x)$ is piecewise linear, then f_1 is convex and therefore f is strictly convex.

Proof. Let $x_0, x_1, x_2, ..., x_n, x_{n+1} \in I$, such that $a = x_0 < x_1 < x_2 < ... < x_n < x_{n+1} = b$, and $\forall k \in [n+1]$, f_1 restricted to (x_{k-1}, x_k) is linear and given by $f_1(x) = a_k x + b_k$. Due to the convexity of f, f_1 is continuous. Hence, $a_k x_k + b_k = a_{k+1}x_k + b_{k+1}$, for every $k \in [n]$. We show that $a_1 \leq a_2 \leq ... \leq a_{n+1}$, which implies that f_1 is convex. Consider the inequality $a_k \leq a_{k+1}$ for any $k \in [n]$. Let $h \in \mathbb{R}$, such that $0 \leq h < \min(x_k - x_{k-1}, x_{k+1} - x_k)$. The convexity of f implies that

$$\frac{f(x_k - h) + f(x_k + h)}{2} \geq f(x_k).$$

Thus,

$$\frac{a_k(x_k - h) + b_k + \alpha(x_k - h)^2 + a_{k+1}(x_k + h) + b_{k+1} + \alpha(x_k + h)^2}{2} \geq \frac{a_k x_k + b_k + \alpha x_k^2 + a_{k+1}x_k + b_{k+1} + \alpha x_k^2}{2},$$

where we have used the continuity of f_1 at x_k on the right hand side of the inequality. Hence,

$$(a_{k+1} - a_k)h + 2\alpha h^2 \geq 0. \tag{9}$$

$(a_{k+1} - a_k)h + 2\alpha h^2$ is a convex quadratic in h with roots 0 and $\frac{-(a_{k+1}-a_k)}{2\alpha}$. This leads to three cases:

1. The root $\frac{-(a_{k+1}-a_k)}{2\alpha} = 0$, i.e., 0 is a root of multiplicity two. This implies $a_k = a_{k+1}$ and Equation 9 holds.
2. The root $\frac{-(a_{k+1}-a_k)}{2\alpha} < 0$. This implies $a_k < a_{k+1}$ and Equation 9 holds for $h \geq 0$.
3. The root $\frac{-(a_{k+1}-a_k)}{2\alpha} > 0$. This implies $a_k > a_{k+1}$. For h such that $0 < h < \min(x_k - x_{k-1}, x_{k+1} - x_k, \frac{-(a_{k+1}-a_k)}{2\alpha})$, Equation 9 does not hold.

Thus, Equation 9 holds $\forall h$, such that $0 \leq h < \min(x_k - x_{k-1}, x_{k+1} - x_k)$ if and only if $a_k \leq a_{k+1}$. Hence, $a_1 \leq a_2 \leq ... \leq a_{n+1}$, which implies that f_1 is convex.

Lemma 4.9. Let $f: I \to \mathbb{R}$ with $I = (a, b)$, such that $0 \in I$ be defined as $f(\alpha) = F(\alpha) + \frac{1}{2}\alpha^2 \geq 0$. If F is continuous, piecewise linear, convex, and $F(0) = 0$, then $F(\alpha) \geq 0$.

Proof. Let $\alpha_0, \alpha_1, \alpha_2, ..., \alpha_n, \alpha_{n+1} \in I$, such that $a = \alpha_0 < \alpha_1 < \alpha_2 < ... < \alpha_n < \alpha_{n+1} = b$, and $\forall k \in [n+1]$, $F(\alpha) = a_k\alpha + b_k$, if $\alpha_{k-1} \leq \alpha \leq \alpha_k$. We will use two cases to prove the result.

Case 1: Suppose $0 \in (\alpha_{k-1}, \alpha_k)$ for some $k \in [n+1]$. Since $F(0) = 0$, $b_k = 0$. In addition, $a_k = 0$, which we will show by contradiction. Suppose $a_k > 0$ and $\alpha_{k-1} < \epsilon < 0$, such that $|\epsilon| < 2a_k$. Then, $f(\epsilon) = a_k\epsilon + \frac{1}{2}\epsilon^2 < 0$, which is a contradiction. The case in which $a_k < 0$ similarly leads to a contradiction. Hence, $F(\alpha) = 0$ on $[\alpha_{k-1}, \alpha_k]$. Due to the convexity of F, $a_1 \leq a_2 \leq \cdots \leq a_n \leq a_{n+1}$. Therefore, $F'(\alpha) \leq 0$ for $\alpha \leq \alpha_{k-1}$ and $F'(\alpha) \geq 0$ for $\alpha \geq \alpha_k$. Hence, $F(\alpha) \geq 0$.

Case 2: Suppose $\alpha_k = 0$ for some $k \in [n]$. Since $F(0) = 0$, $F(\alpha) = a_k\alpha$ and $f(\alpha) = a_k\alpha + \frac{1}{2}\alpha^2$ on $\alpha_{k-1} \leq \alpha \leq \alpha_k = 0$. We will show that $a_k \leq 0$. Suppose $a_k > 0$. Then, $f(\alpha) = \alpha(a_k + \frac{1}{2}\alpha) < 0$ on $-2a_k < \alpha < 0$, which is a contradiction. Similarly, $F(\alpha) = \alpha a_{k+1}$ and $f(\alpha) = \alpha a_{k+1} + \frac{1}{2}\alpha^2$ on $0 = \alpha_k \leq \alpha \leq \alpha_{k+1}$. We will show that $a_{k+1} \geq 0$. Suppose $a_{k+1} < 0$. Then, $f(\alpha) = \alpha(a_{k+1} + \frac{1}{2}\alpha) < 0$ on $0 < \alpha < 2|a_{k+1}|$, which is a contradiction. Due to the convexity of F, $a_1 \leq a_2 \leq \cdots \leq a_n \leq a_{n+1}$. Therefore, $F'(\alpha) \leq a_k \leq 0$ for $\alpha \leq \alpha_k$ and $F'(\alpha) \geq a_{k+1} \geq 0$ for $\alpha \geq \alpha_k$. Hence, $F(\alpha) \geq 0$.

Lemma 4.10. Let $\mathbf{X} \in \mathbb{R}^{m\times n}$, $\mathbf{W},\mathbf{Z} \in \mathbb{R}^{n\times m}$, $\mu > 0$, $\lambda_C \geq 0$, and $\forall i \in [n]$

$$\mathbf{W}(i,:) = \underset{\mathbf{y}\in\mathbb{R}^m}{\operatorname{argmin}} \left[\frac{1}{2}\|\mathbf{y} - \frac{1}{\mu}\mathbf{L}^{\mathbf{T}}(i,:)\|_2^2 + \frac{\lambda_C}{2\mu}\|\mathbf{y}\|_\infty\right],$$

where $\mathbf{L}(\mathbf{Z}) = \mu\mathbf{Z}^{\mathbf{T}} + \mathbf{X}\mathbf{X}^{\mathbf{T}}\mathbf{X} - \mathbf{X}\mathbf{X}^{\mathbf{T}}\mathbf{Z}^{\mathbf{T}}\mathbf{X}^{\mathbf{T}}\mathbf{X}$. Then, for every $\mathbf{H} \in \mathbb{R}^{n\times m}$,

$$\widehat{J}(\mathbf{W}+\mathbf{H};\mathbf{Z}) \geq \widehat{J}(\mathbf{W};\mathbf{Z}) + \mu\|\mathbf{H}\|_F^2.$$

Proof. By definition,

$$\begin{aligned}
\widehat{J}(\mathbf{W}+\mathbf{H},\mathbf{Z}) &= \|\mathbf{X} - \mathbf{X}(\mathbf{W}+\mathbf{H})\mathbf{X}\|_F^2 + \lambda_C\sum_{i=1}^n \|(\mathbf{W}+\mathbf{H})(i,:)\|_\infty \\
&\quad + \mu\|\mathbf{W}+\mathbf{H}-\mathbf{Z}\|_F^2 - \|\mathbf{X}(\mathbf{W}+\mathbf{H})\mathbf{X} - \mathbf{XZX}\|_F^2 \\
&= \widehat{J}(\mathbf{W},\mathbf{Z}) + \lambda_C\sum_{i=1}^n \left(\|(\mathbf{W}+\mathbf{H})(i,:)\|_\infty - \|\mathbf{W}(i,:)\|_\infty\right) \\
&\quad + 2\,\mathrm{tr}\{\mathbf{H}(\mu\mathbf{W}^{\mathbf{T}} - \mathbf{L})\} + \mu\|\mathbf{H}\|_F^2 \\
&= \widehat{J}(\mathbf{W},\mathbf{Z}) + \sum_{i=1}^n \Big[\lambda_C(\|(\mathbf{W}+\mathbf{H})(i,:)\|_\infty - \|\mathbf{W}(i,:)\|_\infty) \\
&\quad + 2\mu\left\langle \mathbf{H}(i,:), (\mathbf{W} - \frac{1}{\mu}\mathbf{L}^{\mathbf{T}})(i,:)\right\rangle\Big] + \mu\|\mathbf{H}\|_F^2.
\end{aligned}$$

To prove the result, we need to show that

$$\sum_{i=1}^n \Big[\lambda_C(\|(\mathbf{W}+\mathbf{H})(i,:)\|_\infty - \|\mathbf{W}(i,:)\|_\infty) + 2\mu\left\langle \mathbf{H}(i,:), (\mathbf{W} - \frac{1}{\mu}\mathbf{L}^{\mathbf{T}})(i,:)\right\rangle\Big] \geq 0.$$

Hence, it suffices to show that $\forall i \in [n]$,

$$\frac{\lambda_C}{2\mu}(\|(\mathbf{W}+\mathbf{H})(i,:)\|_\infty - \|\mathbf{W}(i,:)\|_\infty) + \left\langle \mathbf{H}(i,:), (\mathbf{W} - \frac{1}{\mu}\mathbf{L}^{\mathbf{T}})(i,:)\right\rangle \geq 0. \tag{10}$$

To simplify notation, let $\mathbf{w} = \mathbf{W}(i,:)$, $\mathbf{h} = \mathbf{H}(i,:)$, and $\boldsymbol{\ell} = \mathbf{L}^{\mathbf{T}}(i,:)$. Let

$$F(\mathbf{h}) = \frac{\lambda_C}{2\mu}(\|\mathbf{w}+\mathbf{h}\|_\infty - \|\mathbf{w}\|_\infty) + \left\langle \mathbf{h}, \mathbf{w} - \frac{1}{\mu}\boldsymbol{\ell}\right\rangle,$$

where $\mathbf{h} = \alpha\mathbf{u}$ and $\mathbf{u} \in \mathbb{R}^m$ is a random unit vector. Then, we can denote $F(\mathbf{h}) = F(\alpha,\mathbf{u})$, and fix a $\mathbf{u}$, so that F is only a function of α:

$$F(\alpha) = \frac{\lambda_C}{2\mu}(\|\mathbf{w}+\alpha\mathbf{u}\|_\infty - \|\mathbf{w}\|_\infty) + \alpha\left\langle \mathbf{u}, \mathbf{w} - \frac{1}{\mu}\boldsymbol{\ell}\right\rangle.$$

Let $G(\mathbf{w}) = \frac{1}{2}\|\mathbf{w} - \frac{1}{\mu}\boldsymbol{\ell}\|_2^2 + \frac{\lambda_C}{2\mu}\|\mathbf{w}\|_\infty$. Then,

$$G(\mathbf{w}+\alpha\mathbf{u}) = \frac{1}{2}\|\mathbf{w}+\alpha\mathbf{u} - \frac{1}{\mu}\boldsymbol{\ell}\|_2^2 + \frac{\lambda_C}{2\mu}\|\mathbf{w}+\alpha\mathbf{u}\|_\infty,$$

and

$$\begin{aligned}
G(\mathbf{w}+\alpha\mathbf{u}) - G(\mathbf{w}) &= \frac{1}{2}\|\mathbf{w}+\alpha\mathbf{u} - \frac{1}{\mu}\boldsymbol{\ell}\|_2^2 - \frac{1}{2}\|\mathbf{w} - \frac{1}{\mu}\boldsymbol{\ell}\|_2^2 \\
&\quad + \frac{\lambda_C}{2\mu}\|\mathbf{w}+\alpha\mathbf{u}\|_\infty - \frac{\lambda_C}{2\mu}\|\mathbf{w}\|_\infty \\
&= \frac{1}{2}\left(\|\mathbf{w} - \frac{1}{\mu}\boldsymbol{\ell}\|_2^2 + 2\left\langle \alpha\mathbf{u}, \mathbf{w} - \frac{1}{\mu}\boldsymbol{\ell}\right\rangle \right. \\
&\quad \left. + \|\alpha\mathbf{u}\|_2^2 - \|\mathbf{w} - \frac{1}{\mu}\boldsymbol{\ell}\|_2^2\right) \\
&\quad + \frac{\lambda_C}{2\mu}(\|\mathbf{w}+\alpha\mathbf{u}\|_\infty - \|\mathbf{w}\|_\infty) \\
&= \frac{\lambda_C}{2\mu}(\|\mathbf{w}+\alpha\mathbf{u}\|_\infty - \|\mathbf{w}\|_\infty) \\
&\quad + \alpha\left\langle \mathbf{u}, \mathbf{w} - \frac{1}{\mu}\boldsymbol{\ell}\right\rangle + \frac{1}{2}\alpha^2.
\end{aligned}$$

Thus, $F(\alpha) = G(\mathbf{w}+\alpha\mathbf{u}) - G(\mathbf{w}) - \frac{1}{2}\alpha^2$. Let $f(\alpha) = F(\alpha) + \frac{1}{2}\alpha^2$. We note that f is convex since $f(\alpha) = G(\mathbf{w}+\alpha\mathbf{u}) - G(\mathbf{w})$. To use Lemma 4.8, we also need to show that $F(\alpha)$ is piecewise linear in α. There is a constant term of $F(\alpha)$, $\frac{-\lambda_C}{2\mu}\|\mathbf{w}\|_\infty$, and a linear term, $\alpha\langle\mathbf{u}, \mathbf{w} - \frac{1}{\mu}\boldsymbol{\ell}\rangle$. The remaining term, $\frac{\lambda_C}{2\mu}\|\mathbf{w}+\alpha\mathbf{u}\|_\infty$ is piecewise linear in α, since as α increases

$$\|\mathbf{w}+\alpha\mathbf{u}\|_\infty = \max(\mathbf{w}_1 + \alpha\mathbf{u}_1, \dots, \mathbf{w}_m + \alpha\mathbf{u}_m, -\mathbf{w}_1 - \alpha\mathbf{u}_1, \dots, -\mathbf{w}_m - \alpha\mathbf{u}_m),$$

and the maximum of a set of linear functions is piecewise linear. Thus, $F(\alpha)$ is piecewise linear, and by Lemma 4.8, $F(\alpha)$ is convex.

The remaining step is to show that $F(\alpha) \geq 0$, which will establish the claim in Equation 10 and thus the result. Since $\mathbf{w} = \operatorname{argmin}_{\mathbf{y}} G(\mathbf{y})$, we have $f(\alpha) = G(\mathbf{w}+\alpha\mathbf{u}) - G(\mathbf{w}) \geq 0$. We also know that $F(\alpha)$ is continuous and piecewise linear in α, convex, and $F(0) = 0$. Hence, by Lemma 4.9, $F(\alpha) \geq 0$.

Lemma 4.10 is applied with $\mathbf{W}$ equal to a fixed point of $\mathbf{T}$ to prove Theorem 4.1, part b. However, the proof mirrors that of Daubechies et al. [13], so it is omitted. To complete the proof of Theorem 4.1, we provide the proof of part c.

Proof of Theorem 4.1, part c. The minimizer of $J(\mathbf{W})$ is unique if $\|\mathbf{X} - \mathbf{XWX}\|_F^2$ is strictly convex in $\mathbf{W}$. Since

$$\|\mathbf{X} - \mathbf{XWX}\|_F^2 = \|(\mathbf{X}^{\mathbf{T}} \otimes \mathbf{X})\mathbf{w} - \mathbf{b}\|_2^2,$$

where $\mathbf{b} = \mathrm{vec}(\mathbf{X}) \in \mathbb{R}^{mn}$ and $\mathbf{w} = \mathrm{vec}(\mathbf{W}) \in \mathbb{R}^{mn}$, we need to guarantee that $(\mathbf{X}^{\mathbf{T}} \otimes \mathbf{X})$ has full rank. Since $\mathrm{rank}(\mathbf{X}^{\mathbf{T}} \otimes \mathbf{X}) = \mathrm{rank}(\mathbf{X}^{\mathbf{T}})\mathrm{rank}(\mathbf{X}) = (\mathrm{rank}(\mathbf{X}))^2$, $(\mathbf{X}^{\mathbf{T}} \otimes \mathbf{X})$ has full rank if $\mathbf{X}$ is square and full rank, proving the result.

Remark 4.11. For the minimization problem on line 19 of Algorithm 1, we note that $\|\mathbf{X} - \mathbf{CWX}\|_F^2 + \lambda_R\sum_{j=1}^m \|\mathbf{W}(:,j)\|_\infty$ has a unique minimizer if $\mathbf{X} \in \mathbb{R}^{m\times n}$ is full rank and $m \leq n$. This can be proven similarly to Theorem 4.1, part c.

TABLE 2 Summary of methods that we compare to SF CUR in this section.

Method	Complexity	Computation of C and R
LS-R CUR	$O(k'mn) + O(rmn)$	Columns (rows) sampled from a probability distribution based on leverage scores computed from the leading k' right (left) singular vectors. k' is a rank parameter
LS-D CUR	$O(k'mn) + O(rmn)$	Columns (rows) with largest leverage scores computed from the leading k' right (left) singular vectors. k' is a rank parameter
DEIM CUR	$O(kmn)$	Columns (rows) chosen using DEIM point selection algorithm on the leading k right (left) singular vectors
QR CUR	$O(mn^2)$	Columns (rows) chosen using pivoted QR of $\mathbf{X}$ ($\mathbf{C}^{\mathrm{T}}$)
Rank-k SVD	$O(kmn)$	–

The complexity given is for $\mathbf{X} \in \mathbb{R}^{m\times n}$, letting c be the number of columns in $\mathbf{C}$, r the number of rows of $\mathbf{R}$, and $c = r = k$. We assume that $m < n$ and $c, r \leq m$. For each CUR method, $\mathbf{U} = \mathbf{C}^{+}\mathbf{X}\mathbf{R}^{+}$.

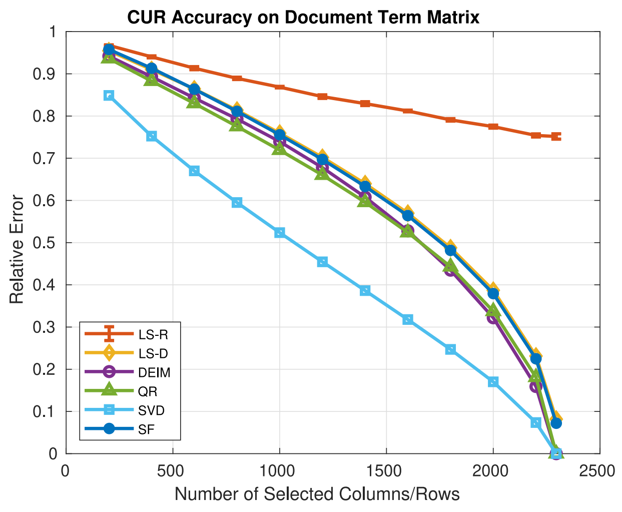

FIGURE 1
Relative error of CUR approximations and the rank-k SVD on a document-term matrix. The rank of the SVD approximation is the same as the number of selected columns/rows for the CUR approximations.

5 Numerical experiments

We demonstrate our CUR algorithm on two datasets: (1) a document-term matrix, and (2) a gene expression dataset. CUR has been previously applied to these types of datasets, e.g., [1, 2, 9] for document-term matrices and [1, 11] for gene expression data. We compare performance of our CUR algorithm to that of (1) the leverage score CUR [1] (the randomized version and deterministic variant), (2) the DEIM CUR [2], (3) the QR CUR variant described in Sorenson and Embree [2], and (4) the low-rank SVD. In the remainder of this article, we denote our CUR algorithm as SF CUR (for surrogate functional CUR), and the deterministic (randomized) leverage score CUR as LS-D (LS-R) CUR. For a brief summary of these methods, see Table 2, and for more details, see Section 2. While each CUR method selects $\mathbf{C}$ and $\mathbf{R}$ separately and allows the user to select c and r, the LS-R CUR is not deterministic. We include results from the LS-R CUR in comparisons of accuracy and computation time since this method is often compared to in the literature, but exclude it from feature selection performance experiments. Comparisons with the SVD are included as a baseline.

Experiments were performed in MATLAB R2023b on the University of Virginia's High-Performance Computing system, Rivanna. We used one (CPU) node, using 8 cores of an Intel(R) Xeon(R) Gold 6248 CPU at 2.50 GHz, and 72 GB of RAM. Code for all experiments performed in this study is provided at https://github.com/klinehan1/cur_feature_selection.

5.1 Document-term matrix

This first experiment serves to compare the accuracy and computation time of the SF CUR with those of other CUR algorithms and the SVD on a document-term matrix, $\mathbf{T} \in \mathbb{R}^{2,389\times 21,238}$. Accuracy is given by the relative error in the Frobenius norm of each approximation, e.g., $\|\mathbf{T} - \mathbf{CUR}\|_F/\|\mathbf{T}\|_F$. $\mathbf{T}$ is sparse with 0.23% non-zero entries and was downloaded and created from the 20 Newsgroups dataset [21] using the scikit-learn package [22]. The documents include the training set documents for the four recreation categories; headers, footers, quotes, and a list of English stop words were removed from the text. The documents were vectorized using TFIDF and the resulting matrix rows were normalized using the ℓ_2 norm.[3] $\mathbf{T}$ is a rank-deficient matrix; rank($\mathbf{T}$) = 2, 295.

Figure 1 presents the relative error and Figure 2 presents the computation time for each CUR approximation in which $c = r$ and c, r vary over {200, 400, ..., 2,200, 2,295}, and for the rank-k SVD in which $k = c = r$. Since the LS-R CUR is randomized, we ran five experiments for each value of c, r and reported the average relative error and time with the standard deviations given by error bars. We note that due to sampling, the LS-R CUR may have chosen a number of columns and/or rows slightly more or less than c and/or r. For both LS-D and LS-R CUR, the rank parameter for leverage score computation was 10.

In general, the SF CUR and LS-D CUR achieve similar relative errors, as do the DEIM CUR and QR CUR, which achieve lower relative errors than those of the SF CUR and LS-D CUR. However, for $c, r \geq 400$, the DEIM CUR has greater computation time than the QR CUR. For $c, r \geq 1,600$, the DEIM CUR has computation times larger than 100 s as compared to computation times of <20 s for all values of c, r for the QR CUR. The LS-R CUR has the smallest computation time for $c, r \geq 400$, but does not perform well in relative error as c, r increase. Clearly, the SVD achieves the lowest relative error and has computation times approximately 10 s. The

3 The data were downloaded using the function sklearn.datasets.fetch_20newsgroups and vectorized using sklearn.feature_extraction.text.TfidfVectorizer. The processing was completed as described using options in these functions. The stop word list used was the built-in list provided in scikit-learn, and the TFIDF calculations were based on the default parameter settings.

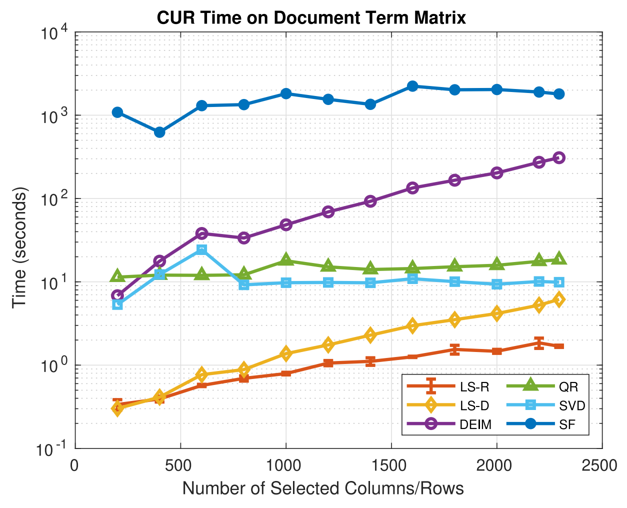

FIGURE 2
Computation time of CUR approximations and the rank-*k* SVD on a document-term matrix. The rank of the SVD approximation is the same as the number of selected columns/rows for the CUR approximations.

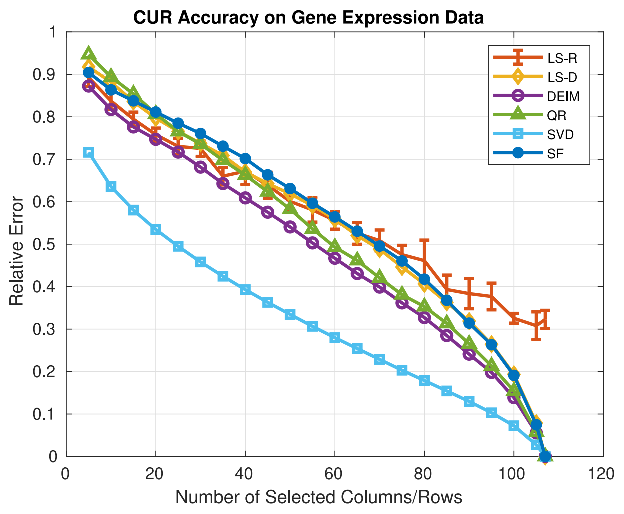

FIGURE 3
Relative error of CUR approximations and the rank-*k* SVD on gene expression data. The rank of the SVD approximation is the same as the number of selected columns/rows for the CUR approximations.

LS-D CUR is relatively fast with computation times less than those of the SF CUR, DEIM CUR, QR CUR, and SVD. The SF CUR is the slowest of the algorithms, with computation times that are generally larger than 1,000 s (16.67 min). While the SF CUR is the most expensive algorithm in terms of computation time, we will demonstrate its effectiveness as a feature selection method in the next experiment.

5.2 Gene expression data

We compare the relative error and feature selection performance of the SF CUR algorithm with that of other complementary CUR algorithms and the SVD on the National Institutes of Health (NIH) gene expression dataset, GSE10072 [23]. We repeat and extend the experiment of Sorenson and Embree [2] who compared the performance of the DEIM CUR and the LS-D CUR on this dataset by (1) calculating the error of each CUR approximation for varying values of c, r, and (2) assessing if the top 15 probes selected by each CUR algorithm separate the patients into those with and without a tumor. We extend this experiment by adding (1) relative error results for the SF CUR, QR CUR, and SVD, (2) computation times for each matrix approximation, (3) metrics to compare the overall probe selection of each matrix approximation method, and (4) results when selecting the top 5, 10, ..., 100 probes. We also include relative error and computation times for the LS-R CUR on this dataset, but exclude it from the probe selection comparison since it is not deterministic.

The GSE10072 dataset, $\mathbf{G} \in \mathbb{R}^{22{,}283\times 107}$, contains gene expression data for 107 patients, of which 58 have a lung tumor and 49 do not. All entries of $\mathbf{G}$ are positive, and larger entries represent a greater reaction to a probe. Each row of $\mathbf{G}$ is centered using its mean. We approximate $\mathbf{G}^{\mathrm{T}}$ (so that probes, i.e., columns, are selected first in the SF algorithm) and again use the Frobenius norm for the relative error calculation (instead of $\|\mathbf{G}-\mathbf{CUR}\|_2$ as in Sorenson and Embree [2]). To assess how well a probe separates the patients into two classes, the number of patients in each class with a (mean-centered) entry in $\mathbf{G}^{\mathrm{T}}$ greater than one for that probe is counted. As mentioned in Sorenson and Embree [2], there are 23 probes for which at least 30 patients with a tumor have an entry greater than one, and 95 probes for which at least 30 patients without a tumor have an entry greater than one. No probe is included in both of these sets.

Figure 3 presents the relative error and Figure 4 presents the computation time for each CUR and SVD approximation. Values of c, r vary over $\{5, 10, ..., 105, 107\}$ and for each CUR approximation $c = r$. For the rank-k SVD, $k = c = r$. We report the average relative error and computation time over five runs for the LS-R CUR, along with the corresponding standard deviations. For both LS-D and LS-R CUR, the rank parameter for leverage score computation was 2. The SF CUR and LS-D CUR have similar relative errors for all values of c, r; however, the SF CUR generally takes about 5 s to compute, whereas the LS-D CUR generally takes about 0.1 s. The QR CUR achieves lower relative error than the SF CUR for $c, r \geq 20$, and the DEIM CUR achieves the lowest relative error of the CUR approximations for every c, r value. The LS-R CUR has an average relative error lower than that of the SF CUR for $c, r \leq 65$. The SF CUR takes the longest to compute, while the other methods have relatively similar computation times under 0.5 s. These trends are fairly similar to the relative error and computation time trends seen in the previous experiments of Section 5.1.

Next, we determine probe selection performance for each CUR and SVD approximation. For each CUR approximation, we set c to the corresponding number of probes, i.e., 5, 10, ..., 100, and report the selected probes (i.e., columns of $\mathbf{G}^{\mathrm{T}}$). For the rank-k SVD, we perform PCA using a rank-2 SVD since the two classes (tumor and no tumor) are separated well when the data are projected onto the

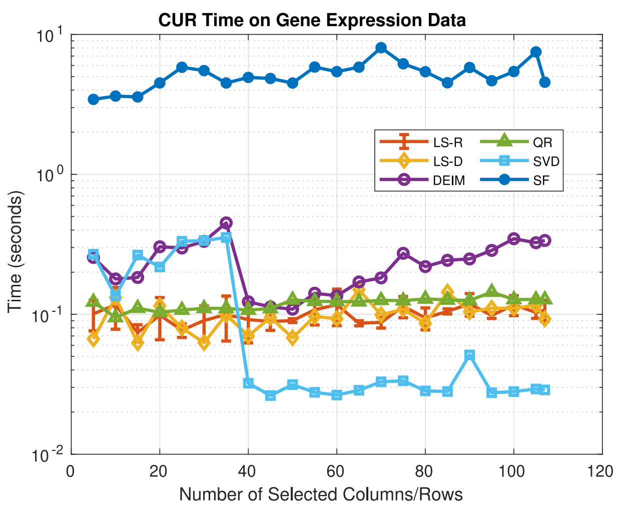

FIGURE 4
Time of CUR approximations and the rank-k SVD on gene expression data. The rank of the SVD approximation is the same as the number of selected columns/rows for the CUR approximations.

leading two principal axes [2],[4] and then select the c probes that have the largest correlation (in absolute value) with either the first or second principal component. To compare the probe selection performance of the five methods, we compute the absolute value of the difference between the number of entries greater than one in $\mathbf{G}^T$ for patients with and without a tumor for each selected probe in each method. See Table 3 for an example of probe selection results using $c = 15$.[5] To quantify the performance of each method, we calculate the median and mean, and standard deviation of the c differences, reported in Tables 4, 5, respectively.

The probes selected by SF CUR and LS-D CUR perform very well in separating patients with a tumor from those without a tumor, as they have larger median and mean differences and smaller standard deviations of differences than the probes selected by the other methods. The probes selected by SVD also perform fairly well in this task due to their fairly large median and mean differences, but they exhibit standard deviations that are generally double those of the SF CUR and LS-D CUR probes. The probes selected by DEIM CUR and QR CUR perform poorly in median and mean difference and exhibit standard deviations that are generally double those of the SF CUR and LS-D CUR probes as well.

While the SF CUR and LS-D CUR methods achieve very similar results,[6] the SF CUR outperforms the LS-D CUR in this experiment. For three out of 20 values of c, the probes selected by the two methods produce equal values for the median and mean difference, and standard deviation of differences (since they select

4 Sorenson and Embree [29] for this result; however, this was withdrawn from the arXiv.

5 For this particular example, SF CUR and LS-D CUR returned the exact same set of 15 probes, hence these results are reported together in Table 3a.

6 These two methods select many of the same probes for each value of c. See Supplementary Table S1.

TABLE 3 Probe selection results for $c = 15$.

Probe	# of Entries > 1 in $\mathbf{G}^T$		
	No tumor	Tumor	|Diff|
(a) SF CUR and LS-D CUR			
210081_at	45	2	43
214387_x_at	48	6	42
211735_x_at	48	5	43
209875_s_at	2	50	48
205982_x_at	48	5	43
209613_s_at	47	2	45
215454_x_at	46	0	46
210096_at	44	6	38
204712_at	43	5	38
203980_at	44	2	42
219230_at	38	2	36
209612_s_at	46	2	44
214135_at	47	3	44
205866_at	39	0	39
205200_at	39	0	39
(b) DEIM CUR			
210081_at	45	2	43
214895_s_at	3	8	5
209156_s_at	6	5	1
211653_x_at	1	18	17
214777_at	3	27	24
219612_s_at	17	17	0
204304_s_at	4	16	12
203824_at	4	17	13
204748_at	14	18	4
201909_at	34	21	13
214774_x_at	0	34	34
211074_at	4	7	3
210096_at	44	6	38
204475_at	0	27	27
214612_x_at	0	16	16
(c) QR CUR			
214387_x_at	48	6	42
201909_at	34	21	13
206239_s_at	2	30	28
205725_at	33	7	26
219612_s_at	17	17	0
203290_at	29	19	10
213831_at	28	18	10

(Continued)

TABLE 3 (Continued)

Probe	# of Entries > 1 in G^T		
	No tumor	Tumor	|Diff|
213674_x_at	8	17	9
204475_at	0	27	27
201884_at	0	30	30
209278_s_at	0	21	21
204885_s_at	12	18	6
207430_s_at	6	9	3
205476_at	11	16	5
209309_at	7	17	10
(d) SVD			
204931_at	36	0	36
206702_at	34	0	34
201540_at	43	0	43
209074_s_at	45	1	44
206742_at	38	0	38
205200_at	39	0	39
219213_at	19	0	19
204894_s_at	29	0	29
213103_at	5	0	5
206209_s_at	36	0	36
219719_at	13	0	13
204719_at	42	0	42
208981_at	22	0	22
210081_at	45	2	43
204396_s_at	36	0	36

Probes are listed in ranked order (1–15) for LS-D CUR, DEIM CUR, QR CUR, and SVD methods since these methods return selected columns in a ranked order.

the same exact probes). However, the SF CUR probes achieve a median difference greater than or equal to that of the LS-D CUR probes for all values of *c*, and a mean difference greater than or equal to that of the LS-D CUR probes for 12 out of 20 values of *c*. Additionally, the standard deviation of differences for the SF CUR probes is less than that of the LS-D CUR probes for 15 out of 20 values of *c*.

6 Protein expression discriminant analysis with CUR

Finally, we present a novel application of CUR as a feature selection method for a clustering algorithm on protein expression data. We modify the clustering analysis of Higuera et al. [14] in which discriminant proteins that critically affect learning in wild type and trisomic mice were discovered in biologically relevant pairwise class comparisons with clustering provided by SOMs and feature selection provided by the Wilcoxon rank-sum test. Specifically, we demonstrate the use and effectiveness of CUR as the feature selection method in a subset of these computational experiments on the same dataset used by Higuera et al. [24]. We compare the performance of multiple CUR algorithms in this application.

6.1 Prior computational experiments

In this section, we provide a summary of the dataset used and computational experiments (MATLAB R2011b) performed in Higuera et al. [14].

6.1.1 Data

The protein expression data used by Higuera et al. was created in prior research [25, 26]. The data were measured from two groups of mice, control and trisomic. Each mouse was exposed to one option from each of two treatments:

1. Context fear conditioning (CFC), an associative learning assessment task, of either context-shock (CS) or shock-context (SC), and
2. an injection of memantine, a drug known to treat learning impairment, or saline.

The CFC task consisted of placing a mouse in a novel cage and allowing it to explore. The context-shock option involves giving the mouse an electric shock after a few minutes of cage exploration, whereas the shock-context option involves an immediate shock to the mouse before exploration. Control mice given the context-shock option will learn an association between the cage and shock, whereas those given the shock-context option will not. However, trisomic mice given the context-shock option will not learn the association between the cage and shock. Thus, the second treatment, an injection of memantine or saline, is given prior to the CFC task. Trisomic mice injected with memantine before the CFC context-shock task will learn the association as the control mice do. Learning in control mice is not affected by memantine injection. Table 6 summarizes the eight classes of mice and presents class size and type of learning. For the remainder of this study, we will refer to classes by their names in Table 6.

Data consist of expression levels for 77 proteins measured from the brains of the 72 mice represented in Table 6. Each protein was measured 15 times for each mouse, giving a total of 1,080 measurements of 77 proteins, resulting in a data matrix that is 1,080 $\times$ 77. For each of the 1,080 observations, the mouse ID and class of the mouse that produced the measurements are also provided. The dataset is available in the supporting information of Higuera et al. [14] and in the University of California, Irvine Machine Learning Repository [24].

Missing data arises as a consequence of the protein measurement process. Higuera et al. processed the data by (1) removing an outlier mouse with mainly missing data, (2) filling in missing entries, and (3) normalizing the data. For any mice in class *c* missing data for protein *p*, the missing entries were replaced with the average expression level of protein *p* from the reported (non-missing) entries for mice in class *c*. Min-max normalization

TABLE 4 Comparison of probe selection methods using the metrics of median (MDN) and mean difference between classes.

# of Probes	SF		LS-D		DEIM		QR		SVD	
	MDN	Mean	MDN	Mean	MDN	Mean	MDN	Mean	MDN	Mean
5	**43**	**43.80**	**43**	**43.80**	17	18	26	21.80	38	39
10	**43**	**42.80**	**43**	**42.80**	12.50	13.20	19.50	19.50	36	32.30
15	**43**	**42**	**43**	**42**	13	16.67	10	16	36	31.93
20	**42.50**	41.75	**42.50**	**42.10**	13	15.80	10	14.55	36	31.95
25	**42**	**40.68**	**42**	40.32	14	16.28	13	15.56	34	31.32
30	**40**	39.20	**40**	**39.30**	13.50	15.57	13	15.87	34	30.67
35	**39**	38.66	**39**	**38.69**	14	15.86	13	16.40	31	29.83
40	**38**	38.28	**38**	**38.30**	14.50	16.15	16.50	17.25	30	29.43
45	**38**	**38.16**	**38**	38.11	16	16.89	17	17.73	31	30.02
50	**37**	**37.94**	36.50	37.76	15.50	15.92	16.50	16.96	30	29.36
55	**36**	**37.42**	**36**	37.31	14	15.02	16	16.62	29	28.49
60	**36**	37.12	**36**	**36.78**	14.50	14.93	14.50	16.10	29	27.63
65	**36**	**36.69**	**36**	36.52	14	14.65	16	16	29	27.55
70	**36**	**36.31**	**36**	36.17	15.50	15.01	16	15.89	28	27
75	**36**	**36.13**	**36**	35.95	14	15.05	16	16.05	28	26.61
80	**35.50**	**35.84**	35	35.64	13	14.49	15.50	16.03	27.50	26.08
85	**35**	**35.54**	**35**	35.44	13	14.25	15	16.07	28	26.04
90	**35**	**35.20**	**35**	35.14	13	14.12	14	15.97	28	25.89
95	**35**	**34.95**	**35**	34.82	13	13.66	14	16.06	28	25.88
100	**34**	**34.63**	**34**	34.49	12.50	13.46	14	16.06	26.50	25.62

A greater difference implies better performance, i.e., better separation of the tumor and no tumor classes by the selected probes. The greatest median and mean differences are bolded in each row.

was then applied to each column of the data, i.e., for each protein. When we analyzed the raw data, we could not identify the outlier mouse and suspected that the data for this mouse had already been removed from the dataset before it was posted for download. We found that 1.7% of entries in the data were missing, with only six out of 77 proteins missing 20 or more measurements out of the 1,080 total. We also discovered that two columns of the raw data are equal; these columns correspond to the proteins ARC_N and pS6_N. These columns are clearly the same after the data are processed as well, and thus make the matrix of protein expression data rank deficient.

6.1.2 Methodology

We first give a high-level overview of the methodology and then follow with more details. SOMs and the Wilcoxon rank-sum test[7] were used to discover discriminant proteins in biologically relevant pairwise class comparisons. An SOM is an unsupervised neural network clustering method that can identify the topology and distribution of data such that clusters that exist close together in the topology should cluster similar data points. It is useful for dimension reduction and provides a 2D visualization of the data. An SOM is used in this case to cluster mice with similar protein expression levels to discover protein expression patterns among classes. The data provided to the SOM included the protein expression data, but not the class of each mouse. The Wilcoxon rank-sum test is then used to find discriminant proteins between pairs of SOM "class-specific clusters" of mice (details below). This non-parametric test checks for equal medians between two independent samples of data, which are not necessarily of the same size.

Higuera et al. used this method on (1) data from the four control classes, (2) data from the four trisomic classes, and (3) a mixture of the two (c-CS-s, c-CS-m, t-CS-m, t-CS-s, and c-SC-s). We explored feature selection with CUR instead of the Wilcoxon rank-sum test on data from the four control classes only. The computational experiments by Higuera et al. on the four control classes are detailed below.

Initially, ten 7×7 SOMs are computed on the processed data from the four control classes, a 570×77 dataset. Since SOM neuron weights are initialized randomly, each SOM instance will most likely be different. The average quantization error, q, is measured for each SOM as

$$q = \frac{1}{n} \sum_{i=1}^{n} \|\mathbf{d}_i - \mathbf{w}_{BMU(\mathbf{d}_i)}\|_2,$$

7 Also called the Mann–Whitney U-test.

TABLE 5 Standard deviation results for probe selection methods.

# of Probes	SF	LS-D	DEIM	QR	SVD
5	**2.39**	**2.39**	16.73	15.94	4.36
10	**3.16**	**3.16**	12.89	12.91	11.98
15	**3.36**	**3.36**	13.80	12.17	11.93
20	3.34	**2.94**	12.49	11.20	10.68
25	**3.94**	4.80	11.62	10.64	9.76
30	5.07	**4.97**	11.17	10.39	10.65
35	**5.21**	5.23	10.75	11.14	10.40
40	**4.99**	5.02	10.87	11.26	10
45	**4.88**	4.90	10.54	11.13	10.22
50	**4.68**	4.86	10.58	11.24	10.44
55	**4.86**	4.97	10.62	11.27	10.37
60	**4.89**	5.11	10.30	10.96	10.89
65	**5.02**	5.14	10.15	11.02	10.54
70	**5.05**	5.22	10.35	10.74	10.36
75	**5.04**	5.16	10.67	11.03	10.33
80	**5.02**	5.16	10.63	11.05	11.10
85	**5.06**	5.19	10.46	11.43	11.30
90	**5.13**	5.19	10.32	11.46	11.32
95	**5.11**	5.30	10.24	11.31	11.26
100	**5.23**	5.39	10.10	11.24	11.42

The smallest standard deviation of differences is bolded in each row.

where n is the number of observations, $\mathbf{d}_i \in \mathbb{R}^{77}$ is an observation (a row of the data matrix), and $\mathbf{w}_{BMU(\mathbf{d}_i)} \in \mathbb{R}^{77}$ is the weight vector of the Best Matching Unit (BMU) or closest neuron to $\mathbf{d}_i$. The SOM with the smallest average quantization error is then used to identify "class-specific clusters," defined in Higuera et al. [14] as "(i) two or more adjacent [neurons] that contain mice of the same class and no mice from other classes, or (ii) a single [neuron] that contains $\geq 80\%$ (or ≥ 12 of 15) of the measurements of one mouse and no measurements of mice from any other class." While the class of each mouse was not used in the learning process of the SOM, the class of each mouse is used to determine the class-specific clusters. Two class-specific clusters can be compared using the weight vectors of the neurons included in the cluster and 77 instances of the Wilcoxon rank-sum test, one for each protein (each neuron weight vector is length 77). For example, to compare levels of the protein in column 5 of the dataset, the Wilcoxon rank-sum test would use two samples: one created from the fifth element of each neuron weight vector for the neurons in the first class-specific cluster, and one created from the fifth element of each neuron weight vector for the neurons in the second class-specific cluster. Those proteins for which the Wilcoxon rank-sum test returns a p-value of <0.05 are considered to be the discriminant proteins between the two class-specific clusters and thus the two classes they represent.

A new 7×7 SOM is then created using data for the discriminant proteins only (a $570 \times k$ matrix where k is the number of discriminant proteins). Class-specific clusters are identified in this SOM. The discriminant proteins are validated through a qualitative analysis of this SOM, which includes the number of mixed class neurons and the number of observations in mixed class neurons as metrics to determine how well the discriminant proteins clustered the data.

In particular, Higuera et al. found the common discriminant proteins in four pairwise comparisons involving successful learning, c-CS-s vs. c-SC-s, c-CS-m vs. c-SC-m, c-CS-m vs. c-SC-s, and c-CS-s vs. c-SC-m, and then used these results in two other computational experiments.

1. Experiment 1 discriminant proteins: The union of those between c-CS-m and c-CS-s and the common discriminant proteins between the four successful learning comparisons.
2. Experiment 2 discriminant proteins: The union of those between c-SC-m and c-SC-s and the common discriminant proteins between the four successful learning comparisons.

Each experiment produced an SOM that was qualitatively analyzed to validate the selection of discriminant proteins. In the following two subsections of this article, we describe how we used CUR as a feature selection method in this methodology and provide results for its use in Experiments 1 and 2.

6.2 Feature selection using CUR

To use CUR as a feature selection method between two class-specific clusters, we construct a matrix $\mathbf{D}$ that contains pairwise differences between neuron weight vectors in opposite clusters. For example, if cluster A contains a neurons (call their weight vectors $\mathbf{A}_1, ..., \mathbf{A}_a$) and cluster B contains b neurons (call their weight vectors $\mathbf{B}_1, ..., \mathbf{B}_b$), then $\mathbf{D} \in \mathbb{R}^{ab \times 77}$, and

$$\mathbf{D}(j + b(i-1), :) = \mathbf{A}_i - \mathbf{B}_j,$$

for $i \in [a]$ and $j \in [b]$. We then compute 77 CUR approximations of $\mathbf{D}$; one for each possible number of columns to select for the matrix $\mathbf{C}$, i.e., $1, 2, ..., 77$.[8] For this particular application, we are only interested in the subset of columns selected for $\mathbf{C}$, and in each CUR approximation that we use, the columns are chosen first, independently of the rows. Hence, we could select any number of rows for the matrix $\mathbf{R}$. We chose to use all rows and set $\mathbf{R} = \mathbf{D}$. To select a CUR approximation from the 77 calculated, we compute the Akaike information criterion (AIC) [27] and the Bayesian information criterion (BIC) [28] for each CUR model as given in the formulas below. Let $\mathbf{D} \in \mathbb{R}^{m \times 77}$, and $\mathbf{CUR}$ be the CUR

8 Since two columns of the raw protein expression data are equal, $\mathbf{D}$ also has this property. Thus, the SF CUR may fail to produce a selection of columns for particular values of the number of columns to select. In these cases, we ignore these values of the number of columns to select.

TABLE 6 Classes of 72 total mice.

Class name	Genotype	CFC (stimulation to learn)	Injection	Learning	Class size
c-CS-s	Control	Context-shock (yes)	Saline	Normal	9
c-CS-m	Control	Context-shock (yes)	Memantine	Normal	10
c-SC-s	Control	Shock-context (no)	Saline	None	9
c-SC-m	Control	Shock-context (no)	Memantine	None	10
t-CS-s	Trisomic	Context-shock (yes)	Saline	Failed	7
t-CS-m	Trisomic	Context-shock (yes)	Memantine	Rescued	9
t-SC-s	Trisomic	Shock-context (no)	Saline	None	9
t-SC-m	Trisomic	Shock-context (no)	Memantine	None	9

approximation to **D**, where $\mathbf{C} \in \mathbb{R}^{m\times c}$. Then,

$$\text{AIC} = 2(mc+3) + 77m\ln\left(\frac{\|\mathbf{D}-\mathbf{CUR}\|_F^2}{77m}\right),$$

and

$$\text{BIC} = (mc+3)\ln(77m) + 77m\ln\left(\frac{\|\mathbf{D}-\mathbf{CUR}\|_F^2}{77m}\right).$$

The CUR model with the lowest AIC score and the CUR model with the lowest BIC score are selected. The columns chosen to be in the matrix **C** of each CUR correspond to the discriminant proteins. We then train two SOMs: the first using the discriminant proteins from the CUR model with the lowest AIC and the second using the discriminant proteins from the CUR model with the lowest BIC. Hence, using CUR for feature selection will result in two possibilities for the set of discriminant proteins, which can be compared through a qualitative assessment of the SOMs trained on each set.

6.3 Results

We repeated Experiments 1 and 2 by Higuera et al. with two exceptions: (1) we used CUR as the feature selection process instead of the Wilcoxon rank-sum test, and (2) each time we needed to use an SOM, we trained 10 SOMs and chose the one with the minimum number of mixed-class neurons. If multiple SOMs had the minimum number of mixed-class neurons, we chose the SOM with the minimum number of observations in mixed-class neurons. We focused on these metrics based on their importance in the qualitative analysis of the discriminant protein SOMs in Higuera et al. [14]. We compared the performance of four CUR algorithms in this feature selection task, SF CUR, LS-D CUR, DEIM CUR, and QR CUR, as well as that of the Wilcoxon rank-sum test. In all experiments, the LS-D CUR rank parameter for leverage score computation was 2. All experiments were run in MATLAB R2023b on a laptop with 16GB RAM and a 2.20 GHz Intel Core i7-1360P processor.[9]

9 The SOM implementation in MATLAB's Deep Learning Toolbox was used with the default parameters except the SOM size.

Figure 5 presents the SOM using all 77 proteins. Neurons are labeled with the classes of mice they contain in sorted order, i.e., the first class listed is the majority class, and the number of observations contained in each neuron is indicated. Neurons are colored by their majority class—c-CS-m: yellow, c-CS-s: green, c-SC-m: tan, c-SC-s: orange—and a bold outline of a neuron represents class-specific cluster membership—c-CS-m class cluster: red outline, c-CS-s class cluster: green outline, c-SC-m class cluster: brown outline, c-SC-s class cluster: black outline. This color scheme is based on that in Higuera et al. [14].

We define a mixed-CS-class neuron as a mixed-class neuron that includes either c-CS-m or c-CS-s observations, and a mixed-SC-class neuron as a mixed-class neuron that includes either c-SC-m or c-SC-s observations. As a reference for comparison in the results of Experiments 1 and 2, the SOM in Figure 5 has five mixed-CS-class neurons, which contain 84 observations, and four mixed-SC-class neurons, which contain 54 observations. The goal of Experiment 1 is to select discriminant proteins such that when an SOM is trained on the discriminant protein data only, the SOM improves, i.e., has a smaller number of mixed-CS-class neurons and observations contained within those neurons, as compared to the SOM in Figure 5 that was trained on all protein data. The goal of Experiment 2 is similar, except that the discriminant protein SOM should have a smaller number of mixed-SC-class neurons and observations contained within those neurons. Since each CUR algorithm results in two potential sets of discriminant proteins (one for the CUR with minimum AIC and one for the CUR with minimum BIC), we present results for nine feature selection methods: (1) Wilcoxon rank-sum test, (2-3) SF CUR AIC/BIC, (4-5) LS-D CUR AIC/BIC, (6-7) DEIM CUR AIC/BIC, and (8-9) QR CUR AIC/BIC.

6.3.1 Experiment 1

For each feature selection method, we (1) identified the common discriminant proteins between the four successful learning comparisons, c-CS-s vs. c-SC-s, c-CS-m vs. c-SC-m, c-CS-m vs. c-SC-s, and c-CS-s vs. c-SC-m, and (2) identified the discriminant proteins between the c-CS-m and c-CS-s classes. The union of these two sets of proteins is the set of discriminant proteins. We present the results of Experiment 1 in Table 7. In addition, Figure 6 contains the discriminant protein SOMs for the

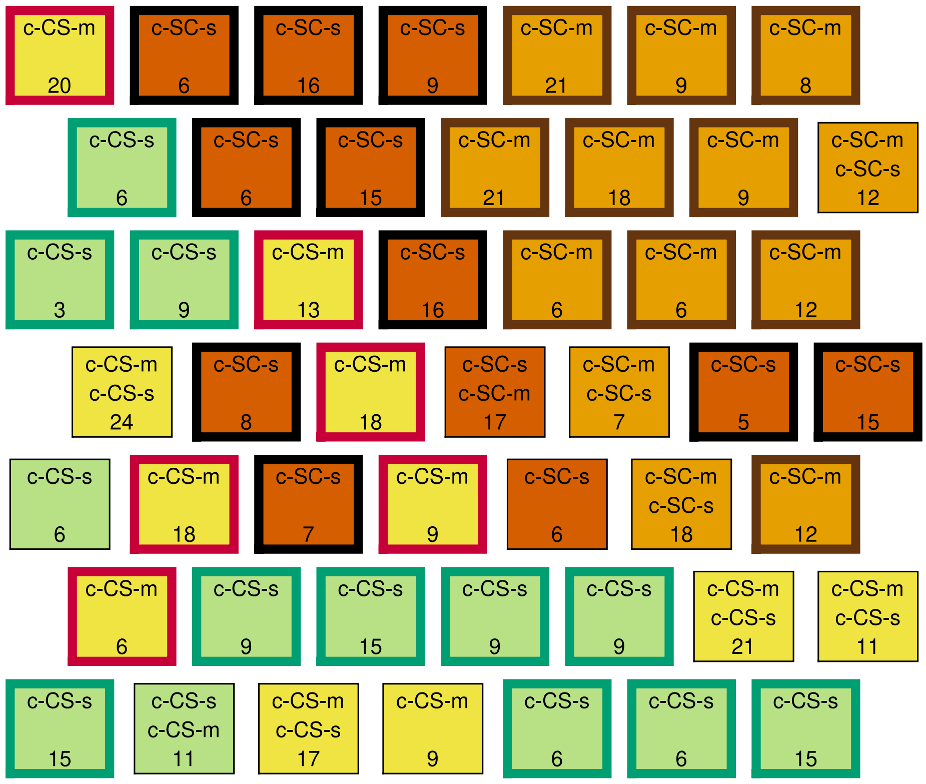

FIGURE 5
SOM using all 77 proteins.

TABLE 7 Experiment 1 results.

Feature selection method	# Mixed-CS neurons	# Observations in mixed-CS neurons	# Discriminant proteins
None	5	84	–
Wilcoxon rank-sum test	**2**	**19**	15
SF CUR—AIC	5	90	35
SF CUR—BIC	7	91	21
LS-D CUR—AIC	7	85	49
LS-D CUR—BIC	6	105	28
DEIM CUR—AIC	5	71	20
DEIM CUR—BIC	7	90	18
QR CUR—AIC	3	47	18
QR CUR—BIC	7	77	16

The minimum number of mixed-CS-class neurons and observations are in bold.

Wilcoxon rank-sum test and each CUR algorithm using the AIC model selection criteria. Since the CUR algorithms using the AIC criteria generally outperformed those using the BIC criteria, the discriminant protein SOMs for the CUR algorithms using the BIC model selection criteria are found in Supplementary Figure S1.

The Wilcoxon rank-sum test performs the best of the feature selection methods, resulting in two mixed-CS-class neurons and 19 observations in those neurons, which is by far the minimum number of observations in mixed-CS-class neurons. It is interesting to note that not only does the Wilcoxon rank-sum test perform the best, but it also selects the fewest number of discriminant proteins. Amongst CUR algorithms, the QR CUR—AIC performs the best, resulting in only three mixed-CS-class neurons containing 47 observations. All CUR algorithms except the QR CUR—AIC, QR CUR—BIC, and DEIM CUR—AIC perform worse than the baseline of no feature selection. Although the QR CUR—BIC

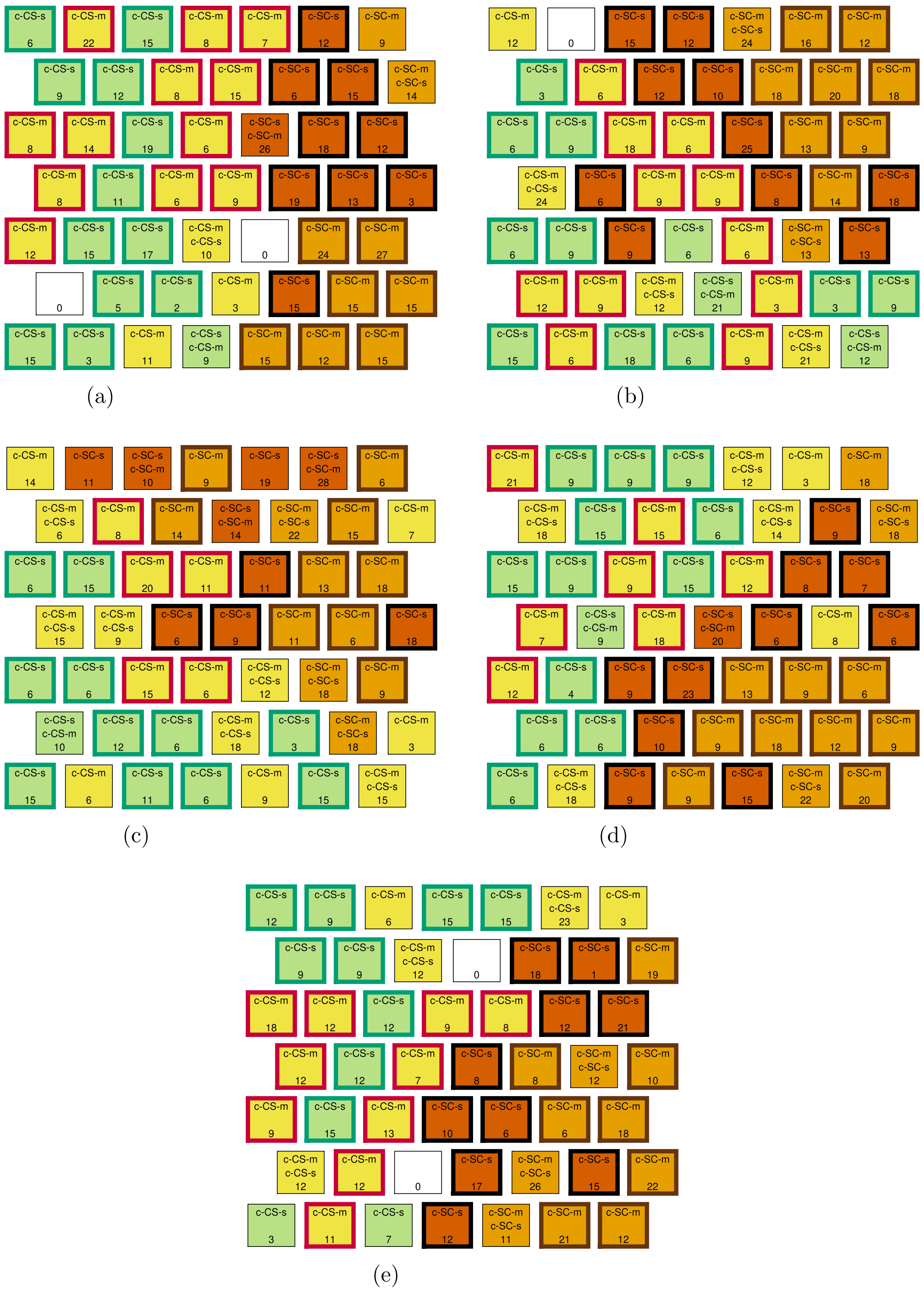

FIGURE 6
Experiment 1 discriminant protein SOMs for the Wilcoxon rank-sum test and CUR algorithms using the AIC model selection criteria. **(a)** Wilcoxon rank-sum test. **(b)** SF CUR, AIC. **(c)** LS-D CUR, AIC. **(d)** DEIM CUR, AIC. **(e)** QR CUR, AIC.

results are mixed: there are two more mixed-CS-class neurons, but these neurons contain fewer observations than the baseline. While CUR-based feature selection did not perform as well as the Wilcoxon rank-sum test for this experiment, we will see a different result in Experiment 2.

6.3.2 Experiment 2

For each feature selection method, we (1) again identified the common discriminant proteins between the four successful learning comparisons, c-CS-s vs. c-SC-s, c-CS-m vs. c-SC-m, c-CS-m vs. c-SC-s, and c-CS-s vs. c-SC-m, and (2) identified the discriminant proteins between the c-SC-m and c-SC-s classes. The union of these two sets of proteins is the set of discriminant proteins. Results for Experiment 2 are presented in Table 8. The discriminant protein SOMs for the Wilcoxon rank-sum test and CUR algorithms using the AIC model selection criteria are given in Figure 7. The discriminant protein SOMs for CUR algorithms using the BIC model selection criteria are given in Supplementary Figure S2.

The best-performing feature selection methods are DEIM CUR—AIC and DEIM CUR—BIC, each resulting in two mixed-SC-class neurons containing 42 observations. The Wilcoxon rank-sum test also performs well, resulting in two mixed-SC-class neurons containing 47 observations. While the SF CUR—AIC results in three mixed-SC-class neurons, it does achieve the minimum number of observations in mixed-SC-class neurons with 37; however, it selects 49 discriminating proteins, which is at least 20 more than all other feature selection methods. The QR CUR—AIC performs relatively well; however, not as well as the Wilcoxon rank-sum test, since it results in three mixed-SC-class neurons containing 52 observations. The SF CUR—BIC, LS-D CUR—AIC, and LS-D CUR—BIC perform poorly compared to the other feature selection methods. In addition, these three methods and the QR CUR—BIC perform worse than the baseline of no feature selection.

The results of Experiments 1 and 2 demonstrate that CUR-based feature selection can be an effective feature selection method for this application, but the results are data and CUR algorithm-dependent. Nonetheless, we demonstrated that DEIM-CUR is an excellent option for feature selection in Experiment 2.

7 Conclusion

We have presented SF CUR, a novel CUR matrix approximation using convex optimization with supporting theory and numerical experiments. Specifically, the SF CUR algorithm uses the surrogate functional [13] to solve the convex optimization problems that arise in the method. To the best of our knowledge, this is the only CUR method using convex optimization that solves for **C** and **R** separately and allows the user to choose the number of columns and rows for inclusion in **C** and **R**, respectively. In addition, we extended the theory of the surrogate functional technique to apply to SF CUR. We numerically demonstrated the use of SF CUR on sparse and dense data. Specifically, we (1) calculated its relative error and computation time on a document-term dataset and a gene expression dataset, and (2) used it as a feature selection method on the gene expression dataset to classify patients as those with or without a tumor, as in Sorenson and Embree [2]. We compared SF CUR performance on these numerical tasks to the SVD and other complementary CUR approximations. We found that while the SVD provides the optimal approximation to each dataset, SF CUR performed the best in selecting probes to separate patient classes on the gene expression dataset, with LS-D CUR a close second in performance. However, the computational time of the SF CUR is about three orders of magnitude higher than that of the LS-D CUR and at least one order of magnitude higher than that of the other CUR approximations used in this feature selection experiment. The computational time of the SF CUR is a current limitation of the method, hence we recommend using SF CUR on small to medium datasets for which computational resources and time constraints are not an issue.

We also presented a novel application of CUR to determine discriminant proteins when clustering protein expression data in an SOM. These computational experiments were based on those in Higuera et al. [14], with the exception that CUR was used

TABLE 8 Experiment 2 results.

Feature selection method	# Mixed-SC neurons	# Observations in mixed-SC neurons	# Discriminant proteins
None	4	54	–
Wilcoxon rank sum test	**2**	47	27
SF CUR—AIC	3	**37**	49
SF CUR—BIC	7	99	27
LS-D CUR—AIC	6	89	29
LS-D CUR—BIC	8	109	28
DEIM CUR—AIC	**2**	42	26
DEIM CUR—BIC	**2**	42	26
QR CUR—AIC	3	52	25
QR CUR—BIC	4	67	20

The minimum number of mixed-SC-class neurons and observations are in bold.

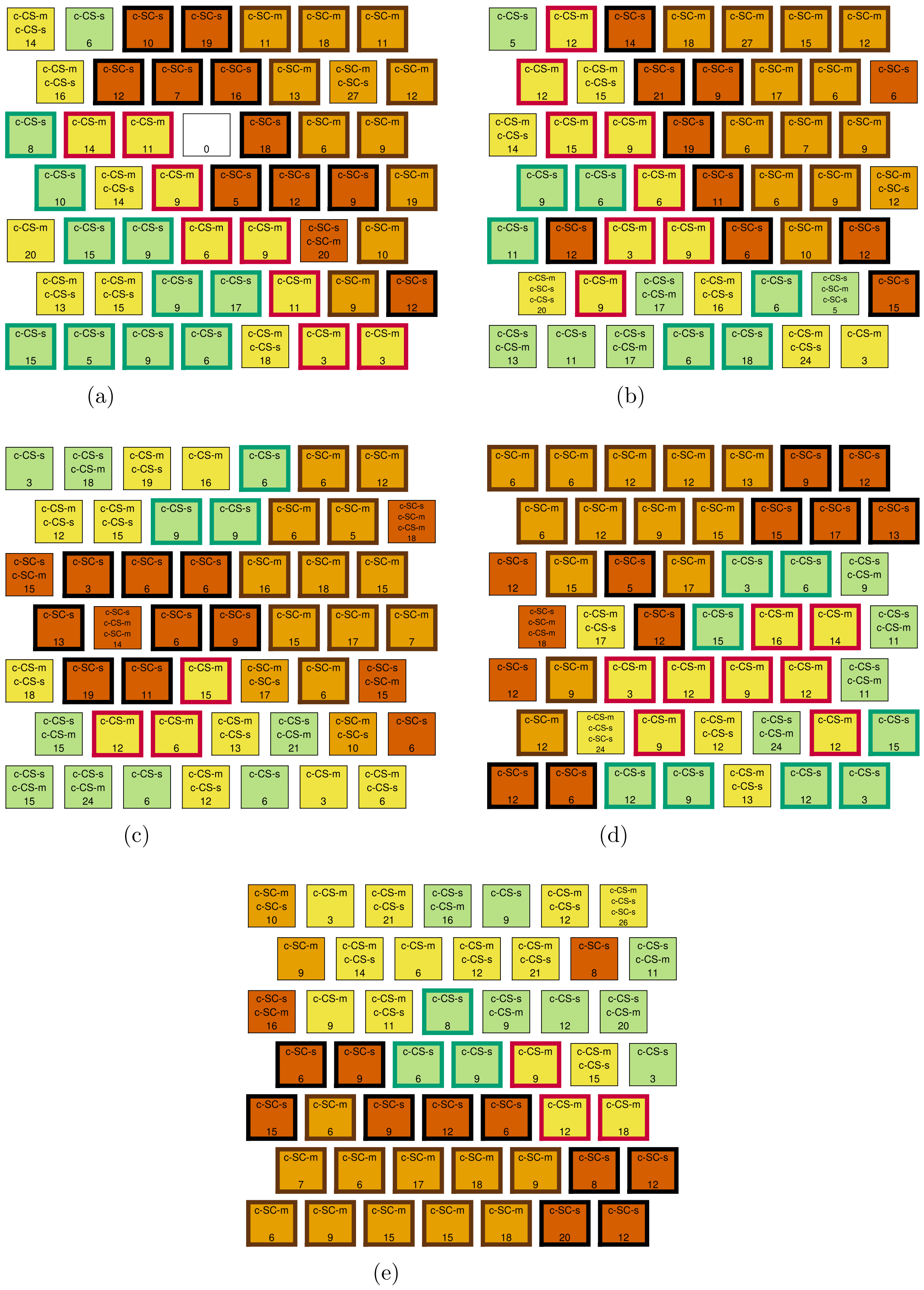

FIGURE 7
Experiment 2 discriminant protein SOMs for the Wilcoxon rank-sum test and CUR algorithms using the AIC model selection criteria. **(a)** Wilcoxon rank-sum test. **(b)** SF CUR, AIC. **(c)** LS-D CUR, AIC. **(d)** DEIM CUR, AIC. **(e)** QR CUR, AIC.

as the feature selection method instead of the Wilcoxon rank-sum test. We compared the performance of SF CUR with that of the Wilcoxon rank-sum test and other complementary CUR approximations. We performed two computational experiments and found that performance varied between datasets and CUR algorithms. While CUR-based feature selection performance was generally poor in Experiment 1, multiple CUR algorithms performed well in Experiment 2. In fact, DEIM CUR—AIC and DEIM CUR—BIC were the best-performing feature selection methods in Experiment 2. In addition, to the best of our knowledge, this was the first use of CUR on protein expression data.

While we have shown the effectiveness of CUR as a feature selection method in numerical experiments on gene and protein expression datasets, it is interesting to note that CUR's performance in these applications is data and algorithm-dependent. SF CUR and LS-D CUR performed very well on the gene expression dataset, whereas DEIM CUR and QR CUR did not. QR CUR—AIC performed moderately well in Experiment 1 on the protein expression dataset, but all other CUR algorithms performed poorly, and DEIM CUR performed very well in Experiment 2 on the protein expression dataset, whereas the LS-D, for example, did not. This naturally leads to the research question of how to choose the CUR algorithm for a given dataset and/or task. We have begun to explore CUR algorithm performance in terms of dataset properties such as sparsity and spectrum, but do not yet have any conclusive results. Future work may include further investigation into why certain CUR algorithms perform better than others in particular applications or on particular datasets, which can hopefully lead to a CUR algorithm selection framework. Other potential areas for future work include generalizing the SF CUR objective function as mentioned in Section 3.3 and SF CUR implementation speed-ups.

Data availability statement

The original contributions presented in the study are included in the article/Supplementary material, further inquiries can be directed to the corresponding author.

Author contributions

KL: Conceptualization, Formal analysis, Methodology, Software, Validation, Visualization, Writing – original draft, Writing – review & editing. RB: Conceptualization, Formal analysis, Methodology, Supervision, Writing – review & editing.

Funding

The author(s) declare that financial support was received for the research and/or publication of this article. Radu Balan was partially supported by NSF under DMS-2108900.

Acknowledgments

The authors acknowledge Research Computing at the University of Virginia for providing computational resources that have contributed to the results reported within this article (URL: https://rc.virginia.edu) and thank Sallie Keller for her support in the early stages of the research.

Conflict of interest

The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Generative AI statement

The author(s) declare that no Gen AI was used in the creation of this manuscript.

Supplementary material

The Supplementary Material for this article can be found online at: https://www.frontiersin.org/articles/10.3389/fams.2025.1632218/full#supplementary-material

References

1. Mahoney MW, Drineas P. CUR matrix decompositions for improved data analysis. *Proc Nat Acad Sci.* (2009) 106:697–702. doi: 10.1073/pnas.0803205106

2. Sorenson DC, Embree M. A DEIM induced CUR factorization. *SIAM J Sci Comput.* (2016) 38:A1454–82. doi: 10.1137/140978430

3. Liu Y, Shao J. High dimensionality reduction using CUR matrix decomposition and auto-encoder for web image classification. In: Qiu G, Lam KM, Kiya H, Xue XY, Kuo CCJ, Lew MS, editors. *Advances in Multimedia Information Processing - PCM 2010.* Berlin, Heidelberg: Springer Berlin Heidelberg (2010). p. 1–12. doi: 10.1007/978-3-642-15696-0_1

4. Esmaeili A, Joneidi M, Salimitari M, Khalid U, Rahnavard N. Two-way spectrum pursuit for CUR decomposition and its application in joint column/row subset selection. In: *2021 IEEE 31st International Workshop on Machine Learning for Signal Processing (MLSP).* Gold Coast: IEEE (2021). p. 1–6. doi: 10.1109/MLSP52302.2021.9596233

5. Li C, Wang X, Dong W, Yan J, Liu Q, Zha H. Joint active learning with feature selection via CUR matrix decomposition. *IEEE Trans Pattern Anal Mach Intell*. (2019) 41:1382–96. doi: 10.1109/TPAMI.2018.2840980

6. Hamm K, Huang L. Perspectives on CUR decompositions. *Appl Comput Harmon Anal*. (2020) 48:1088–99. doi: 10.1016/j.acha.2019.08.006

7. Goreinov SA, Tyrtyshnikov EE, Zamarashkin NL. A theory of pseudoskeleton approximations. *Linear Algebra Appl*. (1997) 261:1–21. doi: 10.1016/S0024-3795(97)80059-6

8. Drineas P, Kannan R, Mahoney MW. Fast Monte Carlo algorithms for matrices III: computing a compressed approximate matrix decomposition. *SIAM J Comput*. (2006) 36:184–206. doi: 10.1137/S0097539704442702

9. Drineas P, Mahoney MW, Muthukrishnan S. Relative-error *CUR* matrix decompositions. *SIAM J Matrix Anal Appl*. (2008) 30:844–81. doi: 10.1137/07070471X

10. Stewart GW. Four algorithms for the efficient computation of truncated pivoted QR approximations to a sparse matrix. *Nume Math*. (1999) 83:313–23. doi: 10.1007/s002110050451

11. Mairal J, Jenatton R, Obozinski G, Bach F. Convex and network flow optimization for structured sparsity. *J Mach Learn Res*. (2011) 12:2681–720.

12. Bien J, Xu Y, Mahoney MW. CUR from a sparse optimization viewpoint. In: *Proceedings of the 23rd International Conference on Neural Information Processing Systems - Volume 1. NeurIPS'10*. Red Hook, NY: Curran Associates Inc. (2010). p. 217–25.

13. Daubechies I, Defrise M, De Mol C. An iterative thresholding algorithm for linear inverse problems with a sparsity constraint. *Commun Pure Appl Math*. (2004) 57:1413–57. doi: 10.1002/cpa.20042

14. Higuera C, Gardiner KJ, Cios KJ. Self-organizing feature maps identify proteins critical to learning in a mouse model of Down syndrome. *PLoS ONE*. (2015) 10:e0129126. doi: 10.1371/journal.pone.0129126

15. Dong Y, Martinsson PG. Simpler is better: a comparative study of randomized pivoting algorithms for CUR and interpolative decompositions. *Adv Comput Math*. (2023) 49:66. doi: 10.1007/s10444-023-10061-z

16. Ida Y, Kanai S, Fujiwara Y, Iwata T, Takeuchi K, Kashima H. Fast deterministic CUR matrix decomposition with accuracy assurance. In: Daumé III H, Singh A, editors. *Proceedings of the 37th International Conference on Machine Learning, Vol. 119*. PMLR (2020). p. 4594–603.

17. Peng Z, Luo M, Li J, Liu H, Zheng Q. ANOMALOUS: a joint modeling approach for anomaly detection on attributed networks. In: Lang J, editor. *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI-18*. International Joint Conferences on Artificial Intelligence Organization (2018). p. 3513–9. doi: 10.24963/ijcai.2018/488

18. Grant M, Boyd S. *CVX: Matlab Software for Disciplined Convex Programming, version 2.1*. (2014). Available online at: http://cvxr.com/cvx (Accessed January 24, 2024).

19. Grant M, Boyd S. Graph implementations for nonsmooth convex programs. In: Blondel V, Boyd S, Kimura H, editors. *Recent Advances in Learning and Control. Lecture Notes in Control and Information Sciences*. Cham: Springer-Verlag Limited (2008). p. 95–110. doi: 10.1007/978-1-84800-155-8_7

20. Parikh N, Boyd S. Proximal algorithms. *Found Trends Optim*. (2014) 1:127–239. doi: 10.1561/2400000003

21. Rennie J. *20 Newsgroups*. (2008). Available online at: http://people.csail.mit.edu/jrennie/20Newsgroups/ (Accessed March 7, 2024).

22. Pedregosa F, Varoquaux G, Gramfort A, Michel V, Thirion B, Grisel O, et al. Scikit-learn: machine learning in Python. *J Mach Learn Res*. (2011) 12:2825–30.

23. Landi M, Dracheva T, Rotunno M, Figueroa J, Liu H, Dasgupta A, et al. Gene expression signature of cigarette smoking and its role in lung adenocarcinoma development and survival. *PLoS ONE*. (2008) 3:e1651. doi: 10.1371/journal.pone.0001651

24. Higuera C, Gardiner K, Cios K. Mice protein expression. *UCI Machine Learning Repository*. (2015). doi: 10.24432/C50S3Z

25. Ahmed MM, Dhanasekaran AR, Block A, Tong S, Costa ACS, Gardiner KJ. Protein Profiles associated with context fear conditioning and their modulation by memantine. *Mol Cell Proteom*. (2014) 13:919–37. doi: 10.1074/mcp.M113.035568

26. Ahmed MM, Dhanasekaran AR, Block A, Tong S, Costa ACS, Stasko M, et al. Protein dynamics associated with failed and rescued learning in the Ts65Dn mouse model of down syndrome. *PLoS ONE*. (2015) 10:1–25. doi: 10.1371/journal.pone.0119491

27. Akaike H. Information theory and an extension of the maximum likelihood principle. In: Petrov BN, Csáki F, editors. *2nd International Symposium on Information Theory*. Budapest, Hungary: Akadémia Kiadó. (1973). p. 267–81.

28. Schwarz G. Estimating the dimension of a model. *Ann Stat*. (1978) 6:461–4. doi: 10.1214/aos/1176344136

29. Kundu A, Nambirajan S, Drineas P. Identifying influential entries in a matrix. *arXiv*. (2013). [Preprint]. arXiv:1310.3556. doi: 10.48550/arXiv.1310.3556