A Flag Decomposition for Hierarchical Datasets

Nathan Mankovich
Universitat de València

Ignacio Santamaria
Universidad de Cantabria

Gustau Camps-Valls
Universitat de València

Tolga Birdal
Imperial College London

Abstract

Flag manifolds encode nested sequences of subspaces and serve as powerful structures for various computer vision and machine learning applications. Despite their utility in tasks such as dimensionality reduction, motion averaging, and subspace clustering, current applications are often restricted to extracting flags using common matrix decomposition methods like the singular value decomposition. Here, we address the need for a general algorithm to factorize and work with hierarchical datasets. In particular, we propose a novel, flag-based method that decomposes arbitrary hierarchical real-valued data into a hierarchy-preserving flag representation in Stiefel coordinates. Our work harnesses the potential of flag manifolds in applications including denoising, clustering, and few-shot learning.

1. Introduction

Hierarchical structures are fundamental across a variety of fields: they shape taxonomies and societies [25], allow us to study 3D objects [26], underpin neural network architectures [59], and form the backbone of language [31]. They reflect parts-to-whole relationships [55] and how our world organizes itself compositionally [49]. However, when handling hierarchies in data, we often resort to the temptation to *flatten* them for simplicity, losing essential structure and context in the process. This tendency is evident in standard dimensionality reduction techniques, like principal component analysis, which ignore any hierarchy the data contains.

In this work, we advocate for an approach rooted in flags to preserve the richness of hierarchical linear subspaces. A flag [33, 36] represents a sequence of nested subspaces with increasing dimensions, denoted by its type or signature $(n_1, n_2, \ldots, n_k; n)$, where $n_1 < n_2 < \ldots < n_k < n$. For instance, a flag of type $(1, 2; 3)$ describes a line within a plane in $\mathbb{R}^3$. By working in flag manifolds—structured spaces of such nested subspaces—we leverage the full complexity of hierarchical data. Flag manifolds have already shown promise in extending traditional methods like Principal Component Analysis (PCA) [36, 47, 53], Independent Component Analysis (ICA) [41–45], generalizing subspace learning [54], and Self-Organizing Maps (SOM) [32]. They enable robust representations for diverse tasks: averaging motions [33], modeling variations in face illumination [11, 33], parameterizing 3D shape spaces [9], and clustering subspaces for video and biological data [33–35, 38].

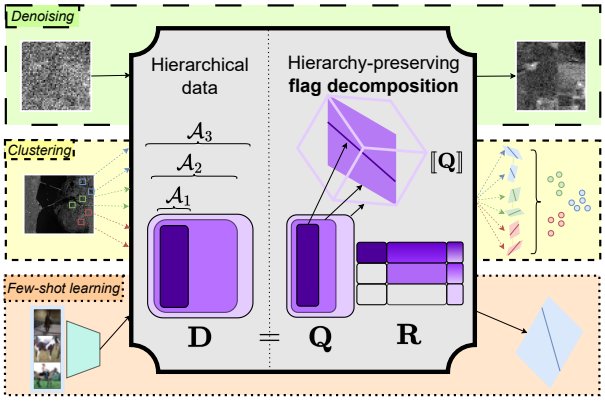


Figure 1. A flag decomposition (center) is used for a hierarchy-preserving flag representation and reconstruction. This decomposition separates a data matrix $\mathbf{D}$ with an associated hierarchy of column indices $\mathcal{A}_1 \subset \mathcal{A}_2 \subset \mathcal{A}_3$ into Stiefel coordinates $\mathbf{Q}$ for a flag $[\![\mathbf{Q}]\!]$, a block-upper triangular matrix $\mathbf{R}$, and a permutation matrix $\mathbf{P}$ (not pictured). Example applications include denoising (green), clustering (yellow), and few-shot learning (orange).

Our main contribution is a Flag Decomposition (FD) specifically designed to preserve hierarchical structures within data (see Fig. 1). First, we formalize the notion of hierarchies in data and the definition of a flag (§2). Next, we provide a practical algorithm for deriving FDs (§3) and outline multiple promising applications (§4). Then we demonstrate its robustness in clustering tasks involving noisy and outlier-contaminated simulated data (§5). Our approach outperforms standard methods, such as Singular Value Decomposition (SVD), in clustering and denoising hyperspec-

tral satellite images. Finally, we show that using flags as prototypes in a few-shot framework improves classification accuracy on benchmark datasets. Final remarks are in (§6) and formal proofs are in the suppl. material. Our implementation is https://github.com/nmank/FD.

2. Preliminaries

We begin by formalizing hierarchical datasets. Then, build to a definition of flags by providing the necessary background in matrix spaces. For notation, italicized capital letters (*e.g.*, $\mathcal{A}$) denote sets, and boldface letters denote matrices and column vectors (*e.g.*, $\mathbf{X}$ and $\mathbf{x}_i$). $[\mathbf{X}]$ denotes the subspace spanned by the columns of $\mathbf{X}$.

Consider the data matrix $\mathbf{D} \in \mathbb{R}^{n\times p}$ with a hierarchy defined by the subsets of column indices

$$\emptyset = \mathcal{A}_0 \subset \mathcal{A}_1 \subset \mathcal{A}_2 \subset \cdots \subset \mathcal{A}_k = \{1, 2, \ldots, p\}. \quad (1)$$

Let $\mathbf{D}_{\mathcal{A}_i}$ be the matrix containing only columns of $\mathbf{D}$ in $\mathcal{A}_i$.

Definition 1 (Column hierarchy for $\mathbf{D}$)**.** *We call $\mathcal{A}_1 \subset \mathcal{A}_2 \subset \cdots \subset \mathcal{A}_k$ a column hierarchy for $\mathbf{D} \in \mathbb{R}^{n\times p}$ when*

$$\dim([\mathbf{D}_{\mathcal{A}_{i-1}}]) < \dim([\mathbf{D}_{\mathcal{A}_i}]) \text{ for } i = 1, 2, \ldots, k. \quad (2)$$

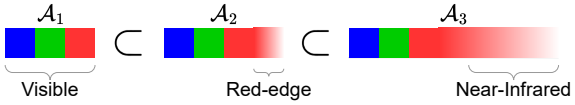
Given a column hierarchy for $\mathbf{D}$ [1], there is a natural correspondence between column and subspace hierarchies

$$\begin{array}{lccccccc} \text{columns:} & \mathcal{A}_1 & \subset & \mathcal{A}_2 & \subset & \cdots & \subset & \mathcal{A}_k \\ \text{subspaces:} & [\mathbf{D}_{\mathcal{A}_1}] & \subset & [\mathbf{D}_{\mathcal{A}_2}] & \subset & \cdots & \subset & [\mathbf{D}_{\mathcal{A}_k}]. \end{array}$$

These hierarchies can include coarse-to-fine neighborhoods (*e.g.*, Example 2.1), spectral hierarchies (*e.g.*, Example 2.2), and feature representations (*e.g.*, Example 2.3).

Example 2.1 (Neighborhood Hierarchy)**.** *Consider $(p_i \times p_i)$ concentric RGB image patches increasing in size with $i = 1, 2, 3$. We store the i^{th} image patch in $\mathbf{D} \in \mathbb{R}^{3\times p_i^2}$. $\mathcal{A}_1$ contains the column indices of the smallest image patch in $\mathbf{D}$, $\mathcal{A}_2$ contains those of the next smallest patch, and $\mathcal{A}_3$ the largest patch. This results in the neighborhood column hierarchy $\mathcal{A}_1 \subset \mathcal{A}_2 \subset \mathcal{A}_3$ for the data matrix $\mathbf{D}$.*

Example 2.2 (Spectral Hierarchy)**.** *Let $\mathbf{D} \in \mathbb{R}^{n\times p}$ be a hyperspectral image with n pixels and p bands. A hierarchy is imposed on the bands by grouping wavelengths into:*



Example 2.3 (Feature Hierarchy)**.** *Consider a feature extractor (*e.g.*, deep network) admitting the following decomposition: $f_\Theta = f_\Theta^{(2)} \circ f_\Theta^{(1)} : \mathbb{R}^N \to \mathbb{R}^n$ where $f_\Theta^{(1)} : \mathbb{R}^N \to \mathbb{R}^n$. s samples $\boldsymbol{x}_1, \cdots, \boldsymbol{x}_s$ are used to obtain*

$$\mathbf{D} = \left[f_\Theta^{(1)}(\boldsymbol{x}_1) \,|\, \cdots \,|\, f_\Theta^{(1)}(\boldsymbol{x}_s) \,|\, f_\Theta(\boldsymbol{x}_1) \,|\, \cdots |\, f_\Theta(\boldsymbol{x}_s) \right].$$

[1] A similar, complex, and well-studied notion of hierarchical matrices is H-matrices [7].

Since information flows from $f_\Theta^{(1)}$ to $f_\Theta^{(2)}$, it is natural to assume that features extracted by $f_\Theta^{(1)}$ span a subspace of the features extracted by f_Θ. Therefore, we propose the hierarchy $\{1, 2, \ldots, s\} \subset \{1, 2, \ldots, 2s\}$.

Next, we build a mathematical formalization of flags.

Definition 2 (Matrix spaces)**.** *The **orthogonal group** $O(n)$ denotes the group of distance-preserving transformations of a Euclidean space of dimension n:*

$$O(n) := \{\mathbf{M} \in \mathbb{R}^{n\times n} : \mathbf{M}^\top \mathbf{M} = \mathbf{M}\mathbf{M}^\top = \mathbf{I}\}. \quad (3)$$

*A **permutation matrix** is a square matrix $\mathbf{P} \in \mathbb{R}^{n\times n}$ where each column and each row contains only a single 1 value and the rest are 0. $\mathbf{DP}$ permutes the columns of $\mathbf{D}$. An important property of permutation matrices is $\mathbf{P}^{-1} = \mathbf{P}^\top$. The **Stiefel manifold** $St(k, n)$, a.k.a. the set of all orthonormal k-frames in $\mathbb{R}^n$, can be represented as the quotient: $St(k, n) = O(n)/O(n-k)$. A point on the Stiefel manifold is parameterized by a tall-skinny $n \times k$ real matrix with orthonormal columns,* i.e. *$\mathbf{X} \in \mathbb{R}^{n\times k}$ where $\mathbf{X}^\top\mathbf{X} = \mathbf{I}$. The **Grassmannian**, $Gr(k, n)$ represents the set of all k-dimensional subspaces of $\mathbb{R}^n$. Each point in $Gr(k, n)$ can be identified with an equivalence class of matrices in the Stiefel manifold, where two matrices are equivalent if their columns span the same subspace. We represent $[\mathbf{X}] \in Gr(k, n)$ using the Stiefel coordinates $\mathbf{X} \in St(k, n)$.*

Definition 3 (Flag)**.** *Let $\mathcal{V}_i$ be an n_i-dimensional subspace of a vector space $\mathcal{V}$ of dimension n. A* flag *of type $(n_1, n_2, \ldots, n_k; n)$, is the nested sequence of subspaces*

$$\emptyset \subset \mathcal{V}_1 \subset \mathcal{V}_2 \subset \cdots \subset \mathcal{V}_k \subset \mathcal{V}. \quad (4)$$

The *flag manifold* $\mathcal{FL}(n_1, n_2, \ldots, n_k; n)$ is a Riemannian manifold and the collection of all flags of type, a.k.a. signature, $(n_1, n_2, \ldots, n_k; n)$ [52, 60]. The first empty subspace, $\emptyset$ with dimension $n_0 = 0$, is now mentioned for completeness but will be dropped from notation and implicit from here on. In this work, we will work with real flags, namely $\mathcal{V} = \mathbb{R}^n$.

Remark 1 (Flag manifold as a quotient of groups)**.** *Ye et al. [60, Proposition 4.10] prove that $\mathcal{FL}(n_1, \ldots, n_k; n)$ is diffeomorphic to the quotient space $St(n_k, n)/(O(m_1) \times O(m_2) \times \cdots \times O(m_k))$ where $m_i = n_i - n_{i-1}$. This fact gives a Stiefel manifold coordinate representation of a flag.*

Definition 4 (Stiefel coordinates for flags [60])**.** *A flag is represented by $\mathbf{X} = [\mathbf{X}_1|\mathbf{X}_2|\cdots|\mathbf{X}_k] \in St(n_k, n)$ where $\mathbf{X}_i \in \mathbb{R}^{n\times m_i}$. Specifically, $\mathbf{X}$ represents the flag*

$$[\![\mathbf{X}]\!] = [\mathbf{X}_1] \subset [\mathbf{X}_1, \mathbf{X}_2] \subset \cdots \subset [\mathbf{X}_1, \mathbf{X}_2, \ldots \mathbf{X}_k] \subset \mathbb{R}^n \quad (5)$$

We say $[\![\mathbf{X}]\!]$ is a flag of type (or signature) $(n_1, n_2, \ldots, n_k; n)$ and $[\mathbf{X}_1, \mathbf{X}_2, \ldots \mathbf{X}_i]$ denotes the span of $[\mathbf{X}_1|\mathbf{X}_2|\cdots|\mathbf{X}_i]$ (for $i = 1, 2, \ldots, k$).

Table 1. Computing the chordal distance on Steifel, Grassmann, and flag manifolds using matrix representatives. $\|\cdot\|_F$ is Frobenius norm.

Representation / Manifold	$\mathbf{X}, \mathbf{Y} \in St(n_k, n)$	$[\mathbf{X}], [\mathbf{Y}] \in Gr(n_k, n)$	$[\![\mathbf{X}]\!], [\![\mathbf{Y}]\!] \in \mathcal{FL}(1, \dots, n_k; n)$
Chordal distance	$\|\mathbf{X} - \mathbf{Y}\|_F$	$\frac{1}{\sqrt{2}}\|\mathbf{X}\mathbf{X}^\top - \mathbf{Y}\mathbf{Y}^\top\|_F$	$\sqrt{\frac{1}{2}\sum_{i=1}^{k} \|\mathbf{X}_i\mathbf{X}_i^\top - \mathbf{Y}_i\mathbf{Y}_i^\top\|_F^2}$

Given the tall-skinny orthonormal matrix representatives $\mathbf{X}, \mathbf{Y} \in \mathbb{R}^{n \times n_k}$, we also utilize their *chordal distances* as given in Tab. 1. The chordal distance on the Stiefel manifold measures the 2-norm of the vectorized matrices. In contrast, the Grassmannian chordal distance measures the 2-norm of the vector of sines of the principal angles [6] between the subspaces through the Frobenius norm of the projection matrices [12]. The chordal distance on the flag manifold [48] arises from the fact that it is a closed submanifold of $\mathrm{Gr}(m_1, n) \times \cdots \times \mathrm{Gr}(m_k, n)$ as shown by Ye et al. [60, Proposition 3.2]. This distance is similar to the Grassmannian chordal distance between subsequent pieces of the flags (*e.g.*, $[\mathbf{X}_i]$ and $[\mathbf{Y}_i]$ for $i = 1, \dots, k$).

3. Flag Decomposition (FD)

We will now introduce our novel Flag Decomposition (FD) that, given $\mathbf{D}$, outputs a hierarchy-preserving flag $[\![\mathbf{Q}]\!]$. From this point on, $[\cdot, \cdot, \cdot]$ denotes column space and $[\cdot | \cdot | \cdot]$ block matrices. Let $\mathcal{B}_i = \mathcal{A}_i \setminus \mathcal{A}_{i-1}$ be the difference of sets for $i = 1, 2, \dots, k$ and $\mathbf{B}_i = \mathbf{D}_{\mathcal{B}_i}$ so that $[\mathbf{D}_{\mathcal{A}_i}] = [\mathbf{B}_1, \mathbf{B}_2, \dots, \mathbf{B}_i]$. We define the permutation matrix $\mathbf{P}$ so $\mathbf{B} = [\mathbf{B}_1 | \mathbf{B}_2 | \cdots | \mathbf{B}_k] = \mathbf{D}\mathbf{P}$. Also, denote the projection matrix onto the null space of $[\mathbf{Q}_i]$ with $\mathbf{Q}_i \in St(m_i, n)$ as $\mathbf{\Pi}_{\mathbf{Q}_i^\perp} = \mathbf{I} - \mathbf{Q}_i\mathbf{Q}_i^\top$. We use $n_0 = 0$, $\mathcal{A}_0 = \emptyset$, and $\mathbf{\Pi}_{\mathbf{Q}_0^\perp} = \mathbf{I}$.

Definition 5 (Hierarchy-preserving flags)**.** *A flag $[\![\mathbf{X}]\!] \in \mathcal{FL}(n_1, n_2, \dots, n_k; n)$ is said to preserve the hierarchy of $\mathbf{D}$ if $[\mathbf{D}_{\mathcal{A}_i}] = [\mathbf{X}_1, \mathbf{X}_2, \dots, \mathbf{X}_i]$ for each $i = 1, 2, \dots, k$.*

If $\mathcal{A}_1 \subset \mathcal{A}_2 \subset \cdots \subset \mathcal{A}_k$ is a column hierarchy for $\mathbf{D}$, then a hierarchy-preserving flag results in the three, equivalent, nested sequences of subspaces

$$\begin{array}{ccccccc} [\mathbf{D}_{\mathcal{A}_1}] & \subset & [\mathbf{D}_{\mathcal{A}_2}] & \subset & \cdots & \subset & [\mathbf{D}_{\mathcal{A}_k}] \\ [\mathbf{B}_1] & \subset & [\mathbf{B}_1, \mathbf{B}_2] & \subset & \cdots & \subset & [\mathbf{B}_1, \mathbf{B}_2, \dots, \mathbf{B}_k] \\ [\mathbf{X}_1] & \subset & [\mathbf{X}_1, \mathbf{X}_2] & \subset & \cdots & \subset & [\mathbf{X}_1, \mathbf{X}_2, \dots, \mathbf{X}_k]. \end{array}$$

SVD and QR decomposition can recover flags from data with certain, limited column hierarchies (see suppl. material for details). However, when faced with a more complex column hierarchy, both QR and SVD cannot recover the entire hierarchy-preserving flag (see Fig. 2).

These examples motivate a generalized decomposition of $\mathbf{D}$ that outputs a hierarchy-preserving flag. In particular, unlike in QR decomposition, $\mathbf{D}$ can be rank-deficient (*e.g.*, $\mathrm{rank}(\mathbf{D}) < p$); and unlike the SVD, we can decompose into flags of type $(n_1, n_2, \dots, n_k; n)$ with $n_k \leq p$.

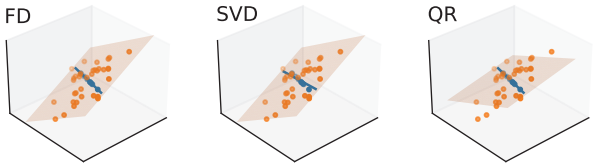


Figure 2. We recover a flag from $\mathbf{D}$ with hierarchy $\mathcal{A}_1 \subset \mathcal{A}_2$. Columns of $\mathbf{D}$ are plotted as points with $\mathcal{A}_1$ in blue and $\mathcal{A}_2 \setminus \mathcal{A}_1$ in orange. FD is the only method that recovers the flag (line inside plane). SVD correctly recovers the plane but not the line whereas QR only recovers the line and the plane misses the orange points.

Definition 6 (**Flag Decomposition (FD)**)**.** *Let $\mathbf{D} \in \mathbb{R}^{n \times p}$ be data with the hierarchically nested sequence of column indices $\mathcal{A}_1 \subset \mathcal{A}_2 \subset \cdots \subset \mathcal{A}_k$. A flag decomposition of type $(n_1, n_2, \cdots, n_k; n)$ is the matrix factorization*

$$\mathbf{D} = \mathbf{Q}\mathbf{R}\mathbf{P}^\top \tag{6}$$

where the block structures are

$$\mathbf{Q} = [\underbrace{\mathbf{Q}_1}_{n \times m_1} | \underbrace{\mathbf{Q}_2}_{n \times m_2} | \cdots | \underbrace{\mathbf{Q}_k}_{n \times m_k}] \in \mathbb{R}^{n \times n_k}, \tag{7}$$

$$\mathbf{R} = \begin{bmatrix} \mathbf{R}_{11} & \mathbf{R}_{12} & \cdots & \mathbf{R}_{1k} \\ \mathbf{0} & \mathbf{R}_{22} & \cdots & \mathbf{R}_{2k} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \cdots & \mathbf{R}_{kk} \end{bmatrix} \in \mathbb{R}^{n_k \times p}, \tag{8}$$

$$\mathbf{P} = [\mathbf{P}_1 | \mathbf{P}_2 | \cdots | \mathbf{P}_k] \in \mathbb{R}^{p \times p}. \tag{9}$$

Here, $\mathbf{Q}$ corresponds to the Stiefel coordinates for the hierarchy-preserving flag $[\![\mathbf{Q}]\!] \in \mathcal{FL}(n_1, n_2, \dots, n_k; n)$ with $m_i = n_i - n_{i-1}$ and $n_k \leq p$, $\mathbf{R}$ is a block upper-triangular matrix, and $\mathbf{P}$ is a permutation matrix so that $\mathbf{Q}\mathbf{R} = \mathbf{D}\mathbf{P}$.

We now use Prop. 1 to determine when we can recover a hierarchy-preserving flag from data and then we use Prop. 2 to show how to construct $\mathbf{R}$ and $\mathbf{P}$ from this flag. Finally, we combine Prop. 1 and 2 to define when we can find an FD (see Prop. 3) and investigate its uniqueness (see Prop. 4).

Proposition 1. *Suppose $\mathcal{A}_1 \subset \mathcal{A}_2 \subset \cdots \subset \mathcal{A}_k$ is a column hierarchy for $\mathbf{D}$. Then there exists $\mathbf{Q} = [\mathbf{Q}_1 | \mathbf{Q}_2 | \cdots | \mathbf{Q}_k]$ that are coordinates for the flag $[\![\mathbf{Q}]\!] \in \mathcal{FL}(n_1, n_2, \dots, n_k; n)$ where $n_i = \mathrm{rank}(\mathbf{D}_{\mathcal{A}_i})$ that satisfies $[\mathbf{Q}_i] = [\mathbf{\Pi}_{\mathbf{Q}_{i-1}^\perp} \cdots \mathbf{\Pi}_{\mathbf{Q}_1^\perp} \mathbf{B}_i]$ and the **projection property** (for $i = 1, 2 \dots, k$):*

$$\mathbf{\Pi}_{\mathbf{Q}_i^\perp} \mathbf{\Pi}_{\mathbf{Q}_{i-1}^\perp} \cdots \mathbf{\Pi}_{\mathbf{Q}_1^\perp} \mathbf{B}_i = 0. \tag{10}$$

Proof. Define (for $i = 2, 3, \ldots, k$) the projector onto the null space of $[\mathbf{Q}_1, \mathbf{Q}_2, \ldots, \mathbf{Q}_i]$, as $\mathbf{\Pi}_{\mathbf{Q}_{:i}^\perp} = \mathbf{I} - \mathbf{Q}_{:i}\mathbf{Q}_{:i}^\top$. We use this to define $\mathbf{C}_i = \mathbf{\Pi}_{\mathbf{Q}_{:i-1}^\perp}\mathbf{B}_i$ and $\mathbf{Q}_i \in St(m_i, n)$ so that $[\mathbf{Q}_i] = [\mathbf{C}_i]$. Then we use mathematical induction to show results ending in Eq. (10) and $\mathbf{Q}_i \in St(m_i, n)$ with $m_i = n_i - n_{i-1}$ where $n_i = \mathrm{rank}(\mathbf{D}_{\mathcal{A}_i})$. □

The simplest methods for recovering $\mathbf{Q}$ so that $[\mathbf{Q}_i] = [\mathbf{\Pi}_{\mathbf{Q}_{i-1}^\perp} \cdots \mathbf{\Pi}_{\mathbf{Q}_1^\perp}\mathbf{B}_i]$ include the left singular vectors from the truncated SVD and the $\mathbf{Q}$ matrix from the QR decomposition with pivoting. We will now use $\mathbf{Q}$ and the projection property to construct $\mathbf{R}$ and $\mathbf{P}$ for the FD.

Proposition 2. *Suppose $\mathcal{A}_1 \subset \mathcal{A}_2 \subset \cdots \subset \mathcal{A}_k$ is a column hierarchy for $\mathbf{D}$. Then there exists some hierarchy-preserving $[\![\mathbf{Q}]\!] \in \mathcal{FL}(n_1, n_2, \ldots, n_k; n)$ (with $n_i = \mathrm{rank}(\mathbf{D}_{\mathcal{A}_i})$) that satisfies the projection property of $\mathbf{D}$ and can be used for a flag decomposition of $\mathbf{D}$ with*

$$\mathbf{R}_{i,j} = \begin{cases} \mathbf{Q}_i^\top \mathbf{\Pi}_{\mathbf{Q}_{i-1}^\perp} \cdots \mathbf{\Pi}_{\mathbf{Q}_1^\perp}\mathbf{B}_i, & i = j \\ \mathbf{Q}_i^\top \mathbf{\Pi}_{\mathbf{Q}_{i-1}^\perp} \cdots \mathbf{\Pi}_{\mathbf{Q}_1^\perp}\mathbf{B}_j, & i < j \end{cases}, \tag{11}$$

$$\mathbf{P}_i = \left[\mathbf{e}_{b_{i,1}} \,|\, \mathbf{e}_{b_{i,2}} \,|\, \cdots \,|\, \mathbf{e}_{b_{i,|\mathcal{B}_i|}}\right] \tag{12}$$

where $\{b_{i,j}\}_{j=1}^{|\mathcal{B}_i|} = \mathcal{B}_i$ and $\mathbf{e}_b$ is the $b_{i,j}{}^{\text{th}}$ standard basis vector.

Proof sketch. This is proved using the previous proposition. □

Proposition 3. *A data matrix $\mathbf{D}$ admits a flag decomposition of type $(n_1, n_2, \cdots, n_k; n)$ if and only if $\mathcal{A}_1 \subset \mathcal{A}_2 \subset \cdots \subset \mathcal{A}_k$ is a column hierarchy for $\mathbf{D}$.*

Proof sketch. We use Prop. 1 and 2 and the definition of a column hierarchy for $\mathbf{D}$. Details in suppl. material. □

Therefore, any $\mathbf{D}$ with an associated column hierarchy admits a hierarchy-preserving FD. Now we state a uniqueness result for the FD.

Proposition 4 (Block rotational ambiguity)**.** *Given the FD $\mathbf{D} = \mathbf{QRP}^\top$, any other Stiefel coordinates for the flag $[\![\mathbf{Q}]\!]$ produce an FD of $\mathbf{D}$ (via Prop. 2). Furthermore, different Stiefel coordinates for $[\![\mathbf{Q}]\!]$ produce the same objective function values in Eq. (13) and Eq. (14) (for $i = 1, \cdots, k$).*

Proof sketch. Notice $\mathbf{Q}_i\mathbf{Q}_i^T = (\mathbf{Q}_i\mathbf{M}_i)(\mathbf{Q}_i\mathbf{M}_i)^\top$ for any $\mathbf{Q}_i \in St(m_i, n)$ and $\mathbf{M}_i \in O(m_i)$. See our suppl. material for details. □

3.1. Flag recovery

In this section, we introduce an approach for recovering the FD $\mathbf{D} = \mathbf{QRP}^\top$ from a given, corrupted version of the dataset, $\tilde{\mathbf{D}}$ and the column hierarchy $\mathcal{A}_1 \subset \mathcal{A}_2 \subset \cdots \subset \mathcal{A}_k$ for $\mathbf{D}$. We call recovering $[\![\mathbf{Q}]\!]$ from $\tilde{\mathbf{D}}$ and $\mathcal{A}_1 \subset \mathcal{A}_2 \subset \cdots \subset \mathcal{A}_k$ the *flag recovery*.

Recall that any $[\![\mathbf{Q}]\!]$ satisfying the projection property of $\mathbf{D}$ can be used for a FD (see Prop. 2). However, since we only have access to $\tilde{\mathbf{D}}$, we may not be able to satisfy this property. As a remedy, we try to get as close as possible to satisfying the projection property by optimizing for $[\![\mathbf{Q}]\!]$ such that $\mathbf{\Pi}_{\mathbf{Q}_i^\perp} \cdots \mathbf{\Pi}_{\mathbf{Q}_1^\perp}\tilde{\mathbf{B}}_i \approx \mathbf{0}$ for each $i = 1, 2, \ldots, k$. We minimize this cost column-wise to solve the problem in maximum generality. Specifically, we propose the following minimization:

$$[\![\mathbf{Q}]\!] = \underset{[\![\mathbf{X}]\!] \in \mathcal{FL}(n_1, n_2, \ldots, n_k; n)}{\arg\min} \sum_{i=1}^{k} \sum_{j \in \mathcal{B}_i} \|\mathbf{\Pi}_{\mathbf{X}_i^\perp} \cdots \mathbf{\Pi}_{\mathbf{X}_1^\perp}\tilde{\mathbf{d}}_j\|_r^q \tag{13}$$

for $r \geq 0$, $q > 0$. Choosing small r and q (*e.g.*, $r = 0$ and $q = 1$) would result in a robust flag recovery, optimal for recovering $\mathbf{D}$ in the presence of outlier columns in $\tilde{\mathbf{D}}$. This problem is difficult, even after restricting q and r, so we address the iterative optimization for each $\mathbf{Q}_i$ for $i = 1$, then $i = 2$, and so on until $i = k$.

$$\mathbf{Q}_i = \underset{\mathbf{X} \in St(m_i, n)}{\arg\min} \sum_{j \in \mathcal{B}_j} \|\mathbf{\Pi}_{\mathbf{X}^\perp} \mathbf{\Pi}_{\mathbf{Q}_{i-1}^\perp} \cdots \mathbf{\Pi}_{\mathbf{Q}_1^\perp}\tilde{\mathbf{d}}_j\|_r^q. \tag{14}$$

The solution to the case where $r = q = 2$ is obtained by the first m_i left singular vectors of $\mathbf{\Pi}_{\mathbf{Q}_{i-1}^\perp} \cdots \mathbf{\Pi}_{\mathbf{Q}_1^\perp}\tilde{\mathbf{D}}_{\mathcal{B}_j}$. In general, solving Eq. (14) for some i recovers $\mathbf{Q}_i$ whose columns form a basis for a m_i dimensional subspace in $\mathbb{R}^n$. Although outputting a truncated basis via QR with pivoting or rank-revealing QR decompositions would offer faster alternatives to SVD for solving Eq. (14), SVD offers more reliable subspace recovery [10]. Thus, we use SVD-based algorithms and leave QR methods for future work.

For cases where $\tilde{\mathbf{D}}$ has outlier columns, we use an L_1 penalty, *i.e.*, $q = 1$, and introduce an **IRLS-SVD solver**[2], a simple method that resembles IRLS algorithms for subspace recovery [15, 27–29, 34, 57, 61]. In practice, we implement a vanilla IRLS-SVD algorithm which could further be made faster and provably convergent using tools from [1, 4, 23, 24, 58]. We leave more advanced solvers, as well as working with other values of r and q (e.g., $r = 0$ [30]), for future work.

3.2. Flag-BMGS

We now propose Flag-BMGS, an algorithm for finding FD and its robust version, Robust FD (RFD). Our algorithm is inspired by the Block Modified Gram-Schmidt (BMGS) procedure [3, 19]. Modified Gram-Schmidt (MGS) is a more numerically stable implementation of the classical Gram-Schmidt orthogonalization. BMGS runs an MGS algorithm on block matrices, iteratively projecting and orthonormalizing matrices rather than vectors, to output a QR

[2] IRLS denotes iteratively reweighted least squares.

Table 2. A summary of GS algorithms and their properties.

Algorithm	GS	MGS	BMGS	Flag-BMGS
Stable	✗	✓	✓	✓
Block-wise	✗	✗	✓	✓
Hier.-pres.	✗	✗	✗	✓

decomposition. In contrast, we use Flag-BMGS on a data matrix with a column hierarchy to produce a hierarchy-preserving FD. We summarize the properties of Gram-Schmidt variants in Tab. 2.

Flag-BMGS operates by first generating a permutation matrix $\mathbf{P}$ (see Eq. (12)) to extract the matrix $\mathbf{B} = \mathbf{D}\mathbf{P}^\top$, using the column hierarchy. Then each iteration $i = 1, 2, \ldots, k$ constructs $\mathbf{\Pi}_{\mathbf{Q}_{i-1}^\perp} \cdots \mathbf{\Pi}_{\mathbf{Q}_1^\perp} \mathbf{B}_i$, solves an optimization of the form Eq. (14), and then constructs each $\mathbf{R}_{i,j}$ for $j \leq i$ (see Eq. (11)). In experiments, we call FD the output of Flag-BMGS using SVD with $r = q = 2$ and Robust FD (RFD) the iterative variant using $r = 2$ and $q = 1$ to solve Eq. (14). Algorithm details are in suppl. material.

Stability results and the search for more optimal algorithms, such as those using block Householder transformations [17] are left to future work. Many other block matrix decompositions exist and a brief discussion of such a low-rank block matrix decomposition [46] can be found in suppl. material.

On the flag type. Flag type is an input to Flag-BMGS. *Detecting* or selecting an adapted flag type from data rather than relying on a heuristic choice, is recently addressed by Szwagier *et al*. in principal subspace analysis [53]. The FD model does not benefit from this advance because it preserves hierarchies rather than directions of maximum variance. We now discuss methods for estimating flag type.

Assuming full access to $\mathbf{D}$, the flag type is $(n_1, n_2, \ldots, n_k; n)$ where $n_i = \operatorname{rank}(\mathbf{D}_{\mathcal{A}_i})$ (see Prop. 1). Yet, the data can be corrupted, *i.e*., we observe only $\tilde{\mathbf{D}} = \mathbf{D} + \boldsymbol{\epsilon}$ ($\boldsymbol{\epsilon}$ denotes random noise) instead of the true $\mathbf{D}$. This leads to an estimation problem of the flag type of the FD assuming access to $\tilde{\mathbf{D}}$ and the true (known) column hierarchy for $\mathbf{D}$.

A naive approach to address the problem of flag type estimation for our FD is to run the FD along with a singular value truncation in each SVD. Methods for truncating the SVs include the *elbow* and *Gavish-Dohono* [13, 16]. In this work, given a column hierarchy and $\tilde{\mathbf{D}}$ (but not $\mathbf{D}$), we choose a flag type where $n_k < \operatorname{rank}(\tilde{\mathbf{D}})$ and input it to FD. In doing so, the output of FD forms a reduced-rank approximation of $\mathbf{D}$ denoted $\hat{\mathbf{D}} = \mathbf{Q}\mathbf{R}\mathbf{P}^\top$.

A promising future research direction involves exploring smarter truncation methods for extracting the flag type of $\mathbf{D}$ under specific contamination criteria.

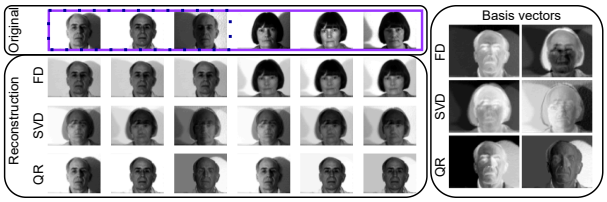


Figure 3. Images from the YFB [5] are flattened and horizontally stacked into $\mathbf{D}$. We use the hierarchy with the images of the first subject as $\mathcal{A}_1$ and all images as $\mathcal{A}_2$. We run FD (flag type $(1, 2)$) and baselines (rank 2). FD is the only method to correctly reconstruct the subjects. We plot the basis vectors (eigenfaces) on the right and find FD extracts basis elements that most closely resemble the subjects.

4. Applications

Before moving on to experimental results, we specify applications of Flag Decomposition (FD), which enables reconstruction in the presence of data corruption, visualization, classification, and a novel prototype and distance for few-shot learning.

4.1. Reconstruction

Consider the matrix $\mathbf{D}$ with an associated column hierarchy $\mathcal{A}_1 \subset \mathcal{A}_2 \subset \cdots \subset \mathcal{A}_k$. Suppose we have a corrupted version $\tilde{\mathbf{D}}$ and the feature hierarchy is known a priori. We use FD to recover $\mathbf{D}$ from $\tilde{\mathbf{D}}$. For example, $\tilde{\mathbf{D}}$ could be a (pixels $\times$ bands) flattened hyperspectral image with a hierarchy on the bands (see Example 2.2 and Fig. 1) with outlier bands or additive noise. Another example includes a hierarchy of subjects: with images of subject 1 inside those of subjects 1 & 2 (see Fig. 3). A reconstruction using FD respects this hierarchy by preserving the identities of the two subjects.

Suppose FD is computed by running Flag-BMGS on $\tilde{\mathbf{D}}$ to output $\mathbf{Q}, \mathbf{R}, \mathbf{P}$. Then we reconstruct $\hat{\mathbf{D}} = \mathbf{Q}\mathbf{R}\mathbf{P}^\top \approx \mathbf{D}$. This is a *low-rank* reconstruction of $\tilde{\mathbf{D}}$ when $n_k < \operatorname{rank}(\tilde{\mathbf{D}})$. *Unlike other reconstruction algorithms this application preserves the column hierarchy.*

4.2. Leveraging the geometry of flags

Consider a collection of $\mathcal{D} = \{\mathbf{D}^{(j)} \in \mathbb{R}^{n \times p_j}\}_{j=1}^N$ with column hierarchies $\mathcal{A}_1^{(j)} \subset \mathcal{A}_2^{(j)} \subset \cdots \subset \mathcal{A}_k^{(j)}$. For example, $\mathcal{D}$ could be a collection of (band $\times$ pixel) hyperspectral image patches. After choosing one flag type $(n_1, n_2, \ldots, n_k; n)$ with $n_k \leq \min(p_1, \ldots, p_N)$, we can use Flag-BMGS with this flag type on each $\mathbf{D}^{(j)}$ to extract the collection of flags $\mathcal{Q} = \{[\![\mathbf{Q}^{(j)}]\!]\}_{j=1}^N \subset \mathcal{FL}(n_1, n_2, \ldots, n_k; n)$. Now, we can use chordal distance on the flag manifold $\mathcal{FL}(n_1, n_2, \ldots, n_k; n)$ or the product of Grassmannians $Gr(m_1, n) \times Gr(m_2, n) \times \cdots \times Gr(m_k, n)$ to build an $N \times N$ distance matrix and run multi-dimensional scaling (MDS) [22] to visualize $\mathcal{D}$ or k-nearest neighbors [39] to cluster $\mathcal{D}$ (see Fig. 1). Other clustering algorithms like k-means can also be implemented with

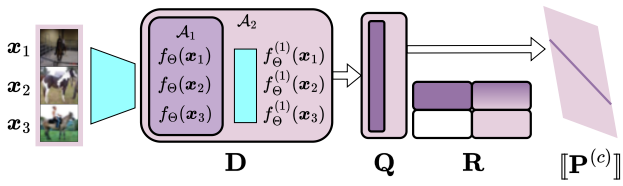


Figure 4. To perform few shot-learning, we embed all $s = 3$ shots $(\boldsymbol{x}_1, \boldsymbol{x}_2, \boldsymbol{x}_3)$ of one class into one flag using FD. The light shapes are pre-trained and frozen feature extractors.

means on products of Grassmannians (*e.g.*, [11, 14, 34, 37]) or chordal flag averages (*e.g.*, [33]). Additionally, we can generate intermediate flags by sampling along geodesics between flags in $\mathcal{Q}$ using tools like `manopt` [8, 56] for exploration of the flag manifold between data samples.

4.3. Few-shot learning

In few-shot learning, a model is trained on very few labeled examples, a.k.a. 'shots' from each class to make accurate predictions. Suppose we have a pre-trained feature extractor $f_\Theta : \mathcal{X} \to \mathbb{R}^n$, parameterized by Θ. In few-shot learning, the number of classes in the training set is referred to as 'ways.' We denote the feature representation of s shots in class c as $f_\Theta(\boldsymbol{x}_{c,1}), f_\Theta(\boldsymbol{x}_{c,2}), \ldots, f_\Theta(\boldsymbol{x}_{c,s})$ where $\boldsymbol{x}_{c,i} \in \mathcal{X}$. The 'support' set is the set of all shots from all classes. A few-shot learning architecture contains a method for determining a class representative (a.k.a. 'prototype') in the feature space ($\mathbb{R}^n$) for each class using its shots. A test sample ('query') is then passed through the encoder, and the class of the nearest prototype determines its class. Overall, a classical few-shot learning architecture is comprised of three (differentiable) pieces: (1) a mapping of shots in the feature space to prototypes, (2) a measure of distance between prototypes and queries, and (3) a loss function for fine-tuning the pre-trained encoder.

Flag classifiers. Take a feature extractor that decomposes into $k \geq 2$ functions. Specifically, $f_\Theta = f_\Theta^{(k)} \circ \cdots \circ f_\Theta^{(1)} : \mathcal{X} \to \mathbb{R}^n$, where each $f_\Theta^{(i)}$ maps to $\mathbb{R}^n$. We can generalize Example 2.3 to construct a k-part hierarchical data matrix. For simplicity, we consider the case where $k = 2$. After constructing a data matrix with a corresponding column hierarchy, we use Flag-BMGS to represent the support of one class, c, in the feature space as $[\![\mathbf{Q}^{(c)}]\!] \in \mathcal{FL}(n_1, n_2; n)$ (see Fig. 4). Now, each subspace $[\mathbf{Q}_1^{(c)}]$ and $[\mathbf{Q}_2^{(c)}]$ represents the features extracted by $f_\Theta^{(1)}$ and f_Θ, respectively.

Given a flag-prototype $[\![\mathbf{Q}^{(c)}]\!] \in \mathcal{FL}(n_1, n_2; n)$ and query $\left\{ f_\Theta^{(1)}(\boldsymbol{x}), f_\Theta(\boldsymbol{x}) \right\} \subset \mathbb{R}^n$, we measure distance between the them as

$$\left\| \mathbf{\Pi}_{\mathbf{Q}_1^{(c)\perp}} f_\Theta^{(1)}(\boldsymbol{x}) \right\|_2^2 + \left\| \mathbf{\Pi}_{\mathbf{Q}_2^{(c)\perp}} f_\Theta(\boldsymbol{x}) \right\|_2^2 \qquad (15)$$

Table 3. Metrics for evaluating simulations. LRSE stands for Log Relative Squared Error, and $\|\cdot\|_F$ is the Frobenius norm. $[\![\mathbf{X}]\!]$ represents true flag, $\mathbf{D}$ the true data, $[\![\hat{\mathbf{X}}]\!]$ the estimated flag, and $\hat{\mathbf{D}}$ the reconstructed data.

Metric (↓)	Formula
Chordal Distance	$\sqrt{\sum_{i=1}^k m_i - \operatorname{tr}(\mathbf{X}_i^T \hat{\mathbf{X}}_i \hat{\mathbf{X}}_i^T \mathbf{X}_i))}$
LRSE	$10 \log_{10} \left(\|\mathbf{D} - \hat{\mathbf{D}}\|_F^2 / \|\mathbf{D}\|_F^2 \right)$

where $\mathbf{\Pi}_{\mathbf{Q}_i^{(c)\perp}} = \mathbf{I} - \mathbf{Q}_i^{(c)} \mathbf{Q}_i^{(c)^\top}$ for $i = 1, 2$. This is proportional to the squared chordal distance on $\mathcal{FL}(1, 2; n)$ when the matrix $\left[f_\Theta^{(1)}(\boldsymbol{x}) | f_\Theta(\boldsymbol{x}) \right]$ is in Stiefel coordinates.

Flag classifiers are *fully differentiable* enabling fine-tuning of the feature extractor with a flag classifier loss. We leave this as an avenue for future work.

5. Results

We run three simulations in Sec. 5.1 to test the capacity of FD and RFD for flag recovery and reconstruction for noisy and outlier-contaminated data. In Sec. 5.2 we visualize clusters of flag representations for hierarchically structured $\mathbf{D}$ matrices using FD. We highlight the utility of FD for denoising hyperspectral images in Sec. 5.3. We cluster FD-extracted flag representations of hyperspectral image patches via a pixel hierarchy in Sec. 5.4. Finally, in Sec. 5.5, we apply flag classifiers to three datasets for classification.

Baselines. While more modern, task-specific algorithms may exist as baselines for each experiment, our primary objective is to demonstrate the effectiveness of FD and RFD (computed using Flag-BMGS) compared to the de facto standards, SVD and QR, in the context of hierarchical data. SVD is a standard denoising method. Two common flag extraction algorithms are SVD [11, 33, 34, 53] and QR [33]. In Sec. 5.5 we compare our results to two standard prototypes (e.g., means and subspaces) for classification within the few-shot learning paradigm.

Metrics. In the additive noise model $\tilde{\mathbf{D}} = \mathbf{D} + \boldsymbol{\epsilon}$, we compute the signal-to-noise ratio (SNR) in decibels (dB) as

$$\operatorname{SNR}(\mathbf{D}, \boldsymbol{\epsilon}) = 10 \log_{10} \left(\|\mathbf{D}\|_F^2 / \|\boldsymbol{\epsilon}\|_F^2 \right). \qquad (16)$$

A negative SNR indicates more noise than signal, and a positive SNR indicates more signal than noise. The rest of our metrics are in Tab. 3.

5.1. Reconstruction Under Corruption

For both experiments, we generate $\mathbf{X} \in St(10, 4)$ that represents $[\![\mathbf{X}]\!] \in \mathcal{FL}(2, 4; 10)$. Then we use $\mathbf{X}$ to build a data matrix $\mathbf{D} \in \mathbb{R}^{10 \times 40}$ with the feature hierarchy $\mathcal{A}_1 \subset \mathcal{A}_2 = \{1, 2, \cdots, 20\} \subset \{1, 2, \cdots, 40\}$. We generate $\tilde{\mathbf{D}}$ as either

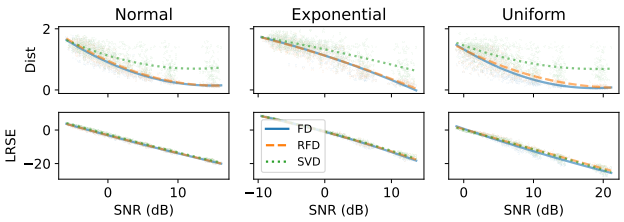


Figure 5. FD & RFD improve flag recovery while maintaining accurate reconstructions. SNR is Eq. (16). LSRE & Dist are in Tab. 3 with *Dist* as the chordal distance. Best fit lines are quadratic.

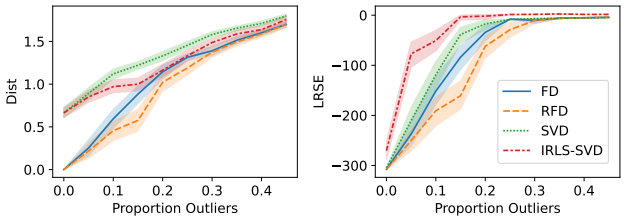


Figure 6. FD and RFD improve flag recovery and reconstruction error over SVD and IRLS-SVD with RFD. LSRE & Dist are defined in Tab. 3 with *Dist* as the chordal distance.

$\mathbf{D}$ with additive noise or $\mathbf{D}$ with columns replaced by outliers. Our goal is to recover $[\![\mathbf{X}]\!]$ and $\mathbf{D} = \mathbf{XRP}^\top$ from $\tilde{\mathbf{D}}$ using FD and RFD with a flag type of $(2, 4; 10)$, and the first 4 left singular vectors from SVD. We evaluate the estimated $[\![\hat{\mathbf{X}}]\!]$ and $\hat{\mathbf{D}}$ using Tab. 3.

Additive noise. We contaminate with noise by $\tilde{\mathbf{D}} = \mathbf{D} + \boldsymbol{\epsilon}$ where $\boldsymbol{\epsilon}$ is sampled from either a mean-zero normal, exponential, or uniform distribution of increasing variance. FD and RFD improve flag recovery over SVD and produce similar reconstruction errors (see Fig. 5).

Robustness to outliers. We construct $\tilde{\mathbf{D}}$ to contain outlier columns. The inlier columns of $\tilde{\mathbf{D}}$ form the flag-decomposable $\mathbf{D} = \mathbf{XRP}^\top$ with the flag $[\![\mathbf{X}]\!]$. FD and RFD outperform SVD and IRLS-SVD, with RFD providing the most accurate flag recovery and inlier reconstructions (see Fig. 6).

5.2. MDS Clustering

We generate 60 $\mathbf{D}$ matrices in 3 clusters, each with 20 points. Then we add normally-distributed noise to generate 60 $\tilde{\mathbf{D}}$ matrices (see suppl. material). We compute the SNR for each $\tilde{\mathbf{D}}$ via Eq. (16) and find the mean SNR for the 60 matrices is -4.79 dB, indicating that significant noise has been added to each $\mathbf{D}$. This experiment aims to find the method that best clusters the $\tilde{\mathbf{D}}$ matrices.

We use SVD (with 4 left singular vectors) and FD (with flag type $(2, 4; 10)$) on each of the 60 $\tilde{\mathbf{D}}$ matrices to recover the flag representations. Then the chordal distance is used to generate distance matrices. Finally, MDS visualizes these data in 2 dimensions. Our additional baseline is run on the Euclidean distance matrix between the flattened $\tilde{\mathbf{D}}$ matrices. We find that FD produces a distance matrix and MDS with most defined clusters in Fig. 7.

5.3. Hyperspectral Image Denoising

We consider denoising images captured by the AVIRIS hyperspectral sensor. Two hyperspectral images are used for model evaluation: the KSC and Indian Pines datasets [2]. KSC is a (512×614) image with 176 bands and Indian Pines is a (145×145) image with 200 bands. We run two experiments, one on each image, by randomly selecting a 50×50 square and flattening it to generate $\mathbf{D} \in \mathbb{R}^{2500 \times p}$ (pixels as rows and bands as columns). Then, we add mean-zero Gaussian noise of increasing variance to obtain $\tilde{\mathbf{D}}$, on which we run our FD and SVD to denoise. LRSE is measured between $\mathbf{D}$ and the denoised reconstruction $\hat{\mathbf{D}}$ (see Tab. 3) to determine the quality of the reconstruction.

For our FD, we specify a flag of type $(8, 9, 10; 2500)$, and SVD is run using the first 10 left singular vectors. The hierarchy used as input to our algorithm mirrors the spectrum hierarchy (see Example 2.2) by assigning $\mathcal{A}_1$ to the first 40 bands, $\mathcal{A}_2$ to the first 100 bands, and $\mathcal{A}_3$ to all the bands. We find in Fig. 8 that FD consistently improves HSI denoising over SVD. When testing exponential and uniform noise, FD and SVD produce similar quality denoising.

5.4. Hyperspectral Image Clustering

We now showcase image patch clustering using the KSC hyperspectral image. The data was pre-processed to remove low SNR and water absorption bands, then split into 3×3 patches of pixels from the same class. Each patch is translated into a $\mathbf{D} \in \mathbb{R}^{176 \times 9}$ (bands as rows and pixels as columns) with the hierarchy described in Example 2.1

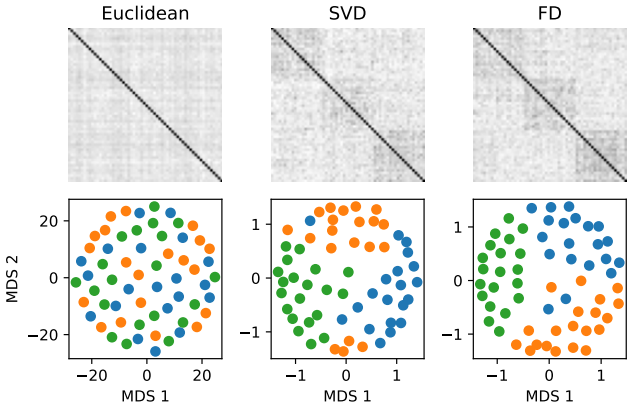


Figure 7. **(Top row)** distance matrices using Euclidean distance (Euclidean) and chordal distance between flags (SVD and FD). **(Bottom row)** 2D representation of the data colored by cluster via MDS applied to the distance matrix.

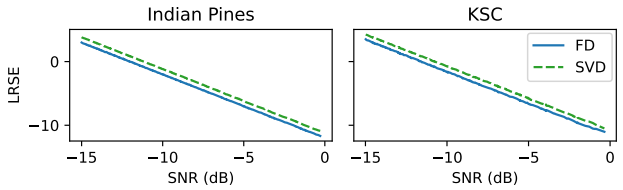


Figure 8. FD improves hyperspectral image denoising over SVD (see SNR Eq. (16), LRSE Tab. 3).

with $\mathcal{A}_1$ as the center pixel. A flag recovery method is run on each $\mathbf{D}$ to extract a flag of type $(1, 8; 176)$. Then, we compute a chordal distance matrix between the collection of flags. Finally, we classify these flags using k-nearest neighbors. We compare FD to QR and SVD in Fig. 9 and find that FD produces the highest classification accuracy for the number of nearest neighbors between 6 and 24.

Instead of using chordal distance between flags to measure distance, we use a sum of Grassmannian chordal distances. We hypothesize that this is a more suited distance for this example because it is more robust to outlier pixels. Given $[\![\mathbf{X}]\!], [\![\mathbf{Y}]\!] \in \mathcal{FL}(1, 8; 176)$, we use a chordal distance on the product of Grassmannians that takes advantage of the embedding of $\mathcal{FL}(1, 8; 176)$ in $Gr(1, 176) \times Gr(7, 176)$. See our suppl. material for details.

5.5. Few-shot Learning

We deploy FD in few-shot learning using an Alexnet [21], pre-trained on ImageNet, as the feature extractor $f_\Theta : \mathcal{X} \to \mathbb{R}^{4096}$, admitting the representation $f_\Theta = f_\Theta^{(1)} \circ f_\Theta^{(2)}$ where the range of both $f_\Theta^{(1)}$ and $f_\Theta^{(2)}$ is $\mathbb{R}^{4096}$. We use the feature hierarchy outlined in Example 2.3 and the procedure in Sec. 4.3 to map the support of one class to a flag prototype using FD (see Fig. 4). The distance between a query point and a flag prototype is Eq. (15). We call this pipeline a flag classifier. Our baselines include Euclidean [51] and subspace [50] classifiers. No fine-tuning is used to optimize the feature extractor in any experiments.

Our two baseline methods, Euclidean and subspace, use means and subspaces as prototypes. Specifically, prototypical networks [51] are a classical few-shot architecture that uses averages for prototypes and Euclidean distance between prototypes and queries. On the other hand, subspace classifiers from adaptive subspace networks [50] use subspaces as prototypes and measure distances between prototypes and queries via projections of the queries onto the prototype. Building upon these baseline methods, we use a flag-based approach (see Figs. 1 and 4). For a fair comparison, baselines stack features extracted by $f_\Theta^{(1)}$ and f_Θ.

We evaluate flag classifiers on EuroSat [18], CIFAR-10 [20], and Flowers102 [40] datasets, and report the average classification accuracy in Tab. 4 over 20 random trials each containing 100 evaluation tasks with 10 query images and 5 ways per task. We find that flag classifiers perform similarly to subspace classifiers and improve classification accuracy in two cases. Further results are in suppl. material.

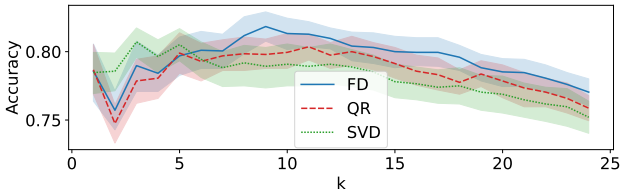


Figure 9. k-nearest neighbors *classification accuracies* (↑) using chordal distance matrices derived from flag representations of 3×3 image patches of the KSC dataset over 20 random trials with a $70\% - 30\%$ training-validation split.

Table 4. *Classification accuracy (↑)* with s shots, 5 ways, and 100 evaluation tasks each containing 10 query images, averaged over 20 random trials. Flag types for 'Flag' are $(s-1, 2(s-1))$ and the subspace dimension is $s-1$.

s	Dataset	Flag	Euc.	Subsp.
3	EuroSat	**77.7**	76.7	77.6
	CIFAR-10	**59.6**	58.6	**59.6**
	Flowers102	**90.2**	88.2	**90.2**
5	EuroSat	**81.8**	80.7	**81.8**
	CIFAR-10	**65.2**	**65.2**	**65.2**
	Flowers102	**93.2**	91.4	**93.2**
7	EuroSat	**83.9**	82.6	83.8
	CIFAR-10	68.0	**68.6**	68.1
	Flowers102	**94.5**	92.7	**94.5**

6. Conclusion

We introduced Flag Decomposition (FD), a novel matrix decomposition that uses flags to preserve hierarchical structures within data. We further proposed Flag-BMGS to robustly find this decomposition even under noise and outlier contamination and studied its properties. With this algorithm, FD augments the machine learning arsenal by providing a robust tool for working with hierarchical data, applicable in tasks like denoising, clustering, and few-shot learning, as demonstrated by our evaluations.

Limitations & future work. Our FD framework is a first step to hierarchy-aware decompositions and leaves ample room for future study. For example, Flag-BMGS is prone to the instabilities of Gram-Schmidt and requires a flag type and column hierarchy as inputs. In the future, we will improve algorithms for faster and more stable computation. Next, we plan to automate the flag type detection and explore fine-tuning a feature extractor for few-shot learning with flags. We will also investigate directly learning (latent) hierarchical structures from data.

Acknowledgements. N. Mankovich thanks Homer Durand, Gherardo Varando, Claudio Verdun, and Bernardo Freitas Paulo da Costa for enlightening conversations on flag manifolds and their applications. N. Mankovich and G. Camps-Valls acknowledge support from the project "Artificial Intelligence for complex systems: Brain, Earth, Climate, Society" funded by the Department of Innovation, Universities, Science, and Digital Society, code: CIPROM/2021/56. This work was also supported by the ELIAS project (HORIZON-CL4-2022-HUMAN-02-02, Grant No. 101120237), the THINKINGEARTH project (HORIZON-EUSPA-2022-SPACE-02-55, Grant No. 101130544), and the USMILE project (ERC-SyG-2019, Grant No. 855187). T. Birdal acknowledges support from the Engineering and Physical Sciences Research Council [grant EP/X011364/1]. T. Birdal was supported by a UKRI Future Leaders Fellowship [grant number MR/Y018818/1] as well as a Royal Society Research Grant RG/R1/241402. The work of I. Santamaria was partly supported under grant PID2022-137099NB-C43 (MADDIE) funded by MICIU/AEI /10.13039/501100011033 and FEDER, UE.

References

[1] Khurrum Aftab, Richard Hartley, and Jochen Trumpf. Generalized Weiszfeld algorithms for lq optimization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37(4):728–745, 2014. 4

[2] Tatyana V. Bandos, Lorenzo Bruzzone, and Gustavo Camps-Valls. Classification of hyperspectral images with regularized linear discriminant analysis. *IEEE Transactions on Geoscience and Remote Sensing*, 47(3):862–873, 2009. 7

[3] Jesse L Barlow. Block modified Gram–Schmidt algorithms and their analysis. *SIAM Journal on Matrix Analysis and Applications*, 40(4):1257–1290, 2019. 4

[4] Amir Beck and Shoham Sabach. Weiszfeld's method: Old and new results. *Journal of Optimization Theory and Applications*, 164:1–40, 2015. 4

[5] Peter N. Belhumeur, Joao P Hespanha, and David J. Kriegman. Eigenfaces vs. fisherfaces: Recognition using class specific linear projection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(7):711–720, 1997. 5

[6] Ake Bjorck and Gene H Golub. Numerical methods for computing angles between linear subspaces. *Mathematics of computation*, 27(123):579–594, 1973. 3

[7] Steffen Börm, Lars Grasedyck, and Wolfgang Hackbusch. Introduction to hierarchical matrices with applications. *Engineering analysis with boundary elements*, 27(5):405–422, 2003. 2

[8] Nicolas Boumal, Bamdev Mishra, P-A Absil, and Rodolphe Sepulchre. Manopt, a Matlab toolbox for optimization on manifolds. *JMLR*, 15(1):1455–1459, 2014. 6

[9] Ioana Ciuclea, Alice Barbora Tumpach, and Cornelia Vizman. Shape spaces of nonlinear flags. In *International Conference on Geometric Science of Information*, pages 41–50. Springer, 2023. 1

[10] James W Demmel. *Applied numerical linear algebra*. SIAM, 1997. 4

[11] Bruce Draper, Michael Kirby, Justin Marks, Tim Marrinan, and Chris Peterson. A flag representation for finite collections of subspaces of mixed dimensions. *Linear Algebra and its Applications*, 451:15–32, 2014. 1, 6

[12] Alan Edelman, Tomás A Arias, and Steven T Smith. The geometry of algorithms with orthogonality constraints. *SIAM Journal on Matrix Analysis and Applications*, 20(2):303–353, 1998. 3

[13] Antonella Falini. A review on the selection criteria for the truncated SVD in Data Science applications. *Journal of Computational Mathematics and Data Science*, 5, 2022. 5

[14] P Thomas Fletcher, Suresh Venkatasubramanian, and Sarang Joshi. The geometric median on riemannian manifolds with application to robust atlas estimation. *NeuroImage*, 45(1): S143–S152, 2009. 6

[15] Vaibhav Garg, Ignacio Santamaria, David Ramirez, and Louis L Scharf. Subspace averaging and order determination for source enumeration. *IEEE Transactions on Signal Processing*, 67(11):3028–3041, 2019. 4

[16] Matan Gavish and David L. Donoho. The optimal hard threshold for singular values is $\frac{4}{\sqrt{3}}$. *IEEE Transactions on Information Theory*, 60(8):5040–5053, 2014. 5

[17] Vincent Griem and Sabine Le Borne. A block Householder–based algorithm for the QR decomposition of hierarchical matrices. *SIAM Journal on Matrix Analysis and Applications*, 45(2):847–874, 2024. 5

[18] Patrick Helber, Benjamin Bischke, Andreas Dengel, and Damian Borth. Eurosat: A novel dataset and deep learning benchmark for land use and land cover classification. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 12(7):2217–2226, 2019. 8

[19] William Jalby and Bernard Philippe. Stability analysis and improvement of the block Gram–Schmidt algorithm. *SIAM journal on scientific and statistical computing*, 12(5):1058–1073, 1991. 4

[20] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. *Technical report*, 2009. 8

[21] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25, 2012. 8

[22] Joseph B Kruskal and Myron Wish. *Multidimensional scaling*. Sage, 1978. 5

[23] Christian Kümmerle and Claudio M Verdun. A scalable second order method for ill-conditioned matrix completion from few samples. In *International Conference on Machine Learning*, pages 5872–5883. PMLR, 2021. 4

[24] Christian Kümmerle, Claudio Mayrink Verdun, and Dominik Stöger. Iteratively reweighted least squares for basis pursuit with global linear convergence rate. *Advances in Neural Information Processing Systems*, 34:2873–2886, 2021. 4

[25] David Lane. Hierarchy, complexity, society. In *Hierarchy in natural and social sciences*, pages 81–119. Springer, 2006. 1

[26] Zhiying Leng, Tolga Birdal, Xiaohui Liang, and Federico Tombari. Hypersdfusion: Bridging hierarchical structures in language and geometry for enhanced 3d text2shape generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 19691–19700, 2024. 1

[27] Gilad Lerman and Tyler Maunu. Fast, robust and non-convex subspace recovery. *Information and Inference: A Journal of the IMA*, 7(2):277–336, 2018. 4

[28] Gilad Lerman and Tyler Maunu. An overview of robust subspace recovery. *Proceedings of the IEEE*, 106(8), 2018.

[29] Gilad Lerman, Michael B McCoy, Joel A Tropp, and Teng Zhang. Robust computation of linear models by convex relaxation. *Foundations of Computational Mathematics*, 15: 363–410, 2015. 4

[30] Guangcan Liu, Zhouchen Lin, Shuicheng Yan, Ju Sun, Yong Yu, and Yi Ma. Robust recovery of subspace structures by low-rank representation. *IEEE transactions on pattern analysis and machine intelligence*, 35(1):171–184, 2012. 4

[31] Robert E Longacre. *Hierarchy in language*. Mouton, 1966. 1

[32] Xiaofeng Ma, Michael Kirby, and Chris Peterson. Self-organizing mappings on the flag manifold with applications to hyper-spectral image data analysis. *Neural Computing and Applications*, 34(1):39–49, 2022. 1

[33] Nathan Mankovich and Tolga Birdal. Chordal averaging on flag manifolds and its applications. In *ICCV*, pages 3881–3890, 2023. 1, 6

[34] Nathan Mankovich, Emily J King, Chris Peterson, and Michael Kirby. The flag median and FlagIRLS. In *CVPR*, pages 10339–10347, 2022. 4, 6

[35] Nathan Mankovich, Helene Andrews-Polymenis, David Threadgill, and Michael Kirby. Module representatives for refining gene co-expression modules. *Physical Biology*, 20 (4):045001, 2023. 1

[36] Nathan Mankovich, Gustau Camps-Valls, and Tolga Birdal. Fun with Flags: Robust principal directions via flag manifolds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024. 1

[37] Nathan J Mankovich. *Subspace and Network Averaging for Computer Vision and Bioinformatics*. PhD thesis, Colorado State University, 2023. 6

[38] Tim Marrinan, J Ross Beveridge, Bruce Draper, Michael Kirby, and Chris Peterson. Finding the subspace mean or median to fit your need. In *CVPR*, 2014. 1

[39] Tim Marrinan, P-A Absil, and Nicolas Gillis. On a minimum enclosing ball of a collection of linear subspaces. *Linear Algebra and its Applications*, 625:248–278, 2021. 5

[40] Maria-Elena Nilsback and Andrew Zisserman. Automated flower classification over a large number of classes. In *2008 Sixth Indian conference on computer vision, graphics & image processing*, pages 722–729. IEEE, 2008. 8

[41] Yasunori Nishimori, Shotaro Akaho, and Mark D Plumbley. Riemannian optimization method on generalized flag manifolds for complex and subspace ICA. In *AIP Conference Proceedings*, pages 89–96. American Institute of Physics, 2006. 1

[42] Yasunori Nishimori, Shotaro Akaho, and Mark D Plumbley. Riemannian optimization method on generalized flag manifolds for complex and subspace ICA. In *AIP Conference*, 2006.

[43] Yasunori Nishimori, Shotaro Akaho, and Mark D Plumbley. Riemannian optimization method on the flag manifold for independent subspace analysis. In *International conference on independent component analysis and signal separation*, pages 295–302. Springer, 2006.

[44] Yasunori Nishimori, Shotaro Akaho, Samer Abdallah, and Mark D Plumbley. Flag manifolds for subspace ICA problems. In *ICASSP*, pages IV–1417. IEEE, 2007.

[45] Yasunori Nishimori, Shotaro Akaho, and Mark D Plumbley. Natural conjugate gradient on complex flag manifolds for complex independent subspace analysis. In *International Conference on Artificial Neural Networks*. Springer, 2008. 1

[46] Frank Ong and Michael Lustig. Beyond low rank+ sparse: Multiscale low rank matrix decomposition. *IEEE journal of selected topics in signal processing*, 10(4), 2016. 5

[47] Xavier Pennec. Barycentric subspace analysis on manifolds. *Annals of Statistics*, 46(6A), 2018. 1

[48] Renaud-Alexandre Pitaval and Olav Tirkkonen. Flag orbit codes and their expansion to Stiefel codes. In *IEEE Information Theory Workshop*, pages 1–5. IEEE, 2013. 3

[49] Stanley N Salthe. *Evolving hierarchical systems: their structure and representation*. Columbia University Press, 1985. 1

[50] Christian Simon, Piotr Koniusz, Richard Nock, and Mehrtash Harandi. Adaptive subspaces for few-shot learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4136–4145, 2020. 8

[51] Jake Snell, Kevin Swersky, and Richard Zemel. Prototypical networks for few-shot learning. *Advances in neural information processing systems*, 30, 2017. 8

[52] Tom Szwagier and Xavier Pennec. Rethinking the Riemannian logarithm on flag manifolds as an orthogonal alignment problem. In *International Conference on Geometric Science of Information*, pages 375–383. Springer, 2023. 2

[53] Tom Szwagier and Xavier Pennec. The curse of isotropy: from principal components to principal subspaces, 2024. 1, 5, 6

[54] Tom Szwagier and Xavier Pennec. Nested subspace learning with flags, 2025. 1

[55] Mohammad Reza Hosseinzadeh Taher, Michael B Gotway, and Jianming Liang. Representing part-whole hierarchies in foundation models by learning localizability composability and decomposability from anatomy via self supervision. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11269–11281, 2024. 1

[56] James Townsend, Niklas Koep, and Sebastian Weichwald. Pymanopt: A Python toolbox for optimization on manifolds using automatic differentiation. *arXiv preprint arXiv:1603.03236*, 2016. 6

[57] Manolis Tsakiris and René Vidal. Dual principal component pursuit. *Journal of Machine Learning Research*, pages 1—-50, 2018. 4

[58] Claudio Mayrink Verdun, Oleh Melnyk, Felix Krahmer, and Peter Jung. Fast, blind, and accurate: Tuning-free sparse regression with global linear convergence. In *The Thirty Seventh Annual Conference on Learning Theory*, pages 3823–3872. PMLR, 2024. 4

[59] Zhicheng Yan, Hao Zhang, Robinson Piramuthu, Vignesh Jagadeesh, Dennis DeCoste, Wei Di, and Yizhou Yu. Hd-cnn: hierarchical deep convolutional neural networks for large scale visual recognition. In *Proceedings of the IEEE international conference on computer vision*, 2015. 1

[60] Ke Ye, Ken Sze-Wai Wong, and Lek-Heng Lim. Optimization on flag manifolds. *Mathematical Programming*, 194(1): 621–660, 2022. 2, 3

[61] Teng Zhang and Gilad Lerman. A novel m-estimator for robust PCA. *The Journal of Machine Learning Research*, 15 (1):749–808, 2014. 4