Principle Component Trees and Their Persistent Homology

Ben Kizaric[1,3], **Daniel Pimentel-Alarcón** [2,3]

[1]Department of Electrical Engineering, University of Wisconsin-Madison
[2]Department of Biostatistics, University of Wisconsin-Madison
[3]Wisconsin Institute For Discovery
benkizaric@gmail.com, pimentelalar@wisc.edu

Abstract

Low dimensional models like PCA are often used to simplify complex datasets by learning a *single* approximating subspace. This paradigm has expanded to *union of subspaces* models, like those learned by subspace clustering. In this paper, we present *Principal Component Trees* (PCTs), a graph structure that generalizes these ideas to identify mixtures of components that together describe the subspace structure of high-dimensional datasets. Each node in a PCT corresponds to a principal component of the data, and the edges between nodes indicate the components that must be *mixed* to produce a subspace that approximates a portion of the data. In order to construct PCTs, we propose two angle-distribution hypothesis tests to detect subspace clusters in the data. To analyze, compare, and select the best PCT model, we define two persistent homology measures that describe their shape. We show our construction yields two key properties of PCTs, namely ancestral orthogonality and non-decreasing singular values. Our main theoretical results show that learning PCTs reduces to PCA under multivariate normality, and that PCTs are efficient parameterizations of intersecting union of subspaces. Finally, we use PCTs to analyze neural network latent space, word embeddings, and reference image datasets.

1 Introduction

Learning a low-dimensional structure that approximates a high-dimensional dataset is a fundamental problem in science and engineering —e.g., learning an approximating subspace using *principal component analysis* (PCA) (Pearson 1901; Hotelling 1933; Jolliffe 2002). This paper introduces a new approximating structure, which we call *principal component trees* (PCT), of which PCA and generalized PCA (also known as subspace clustering (Elhamifar and Vidal 2013)) are special cases.

PCT builds on the union of subspaces model, which assumes that each data sample lies near one of several low-dimensional subspaces. Subspace clustering is then the task of learning the structure of these subspaces and has shown effectiveness in modeling many kinds of data (Vidal, Tron, and Hartley 2008; Mahmood and Pimentel-Alarcón 2022; Li, Zhao, and Zhu 2023). Baked into the majority of subspace clustering algorithms, such as the effective Sparse Subspace Clustering (SSC) method (Elhamifar and Vidal 2013), is the assumption that the subspaces containing the data are disjoint. That is, all points near the span of one subspace are not near the span of any other subspace.

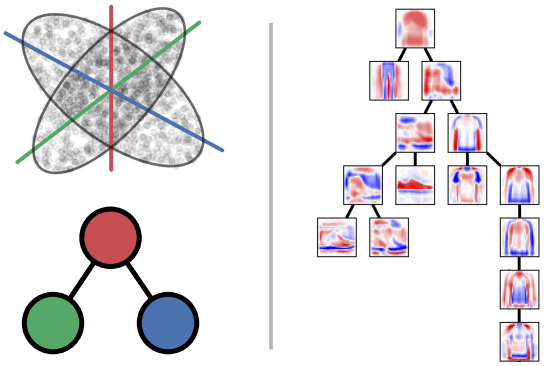


Figure 1: (Left) Two 2D subspace clusters with a 1D intersection and the corresponding Principal Component Tree. (Right) A PCT trained on Fashion MNIST.

It is this disjoint assumption that motivates Principal Component Trees, which can represent a single subspace, multiple disjoint subspaces, and multiple intersecting subspaces. In essence, a PCT is a graph summarizing the hierarchical structure of the principal components in the data. Each node of a PCT corresponds to a single principal component, defined by a principal vector and a singular value. Each edge of a PCT links principal vectors that, together, define a set of subspaces that contain the data. Each of these subspaces is spanned by a *branch* of the PCT, which is the set of nodes that lie on the path between each leaf of the PCT and its root. Branches that share nodes correspond to intersecting subspaces (see Figure 1).

Our algorithm to learn a Principal Component Tree analyzes the structure of the data dynamically, recursively adding child nodes, depending on the structure of the data. Each node will have more than one child only if we detect *disjoint* subspace clusters. We make this determination with a non parametric hypothesis test which compares the distribution of pairwise angles of the data points to the known distribution of uniform angles. If no disjoint structure is detected at any node, the PCT will result in a degenerate tree.

In this case, PCT reduces to PCA. Similarly, the union of subspaces model used by classical subspace clustering is the special case of PCT where a disjoint structure is detected at the root and nowhere else. In general, data following a multivariate normal distribution will yield *tall* trees, and data exhibiting subspace clustered structure will yield *wide* trees.

To quantitatively measure tree structure, we turn to tools in persistent homology that measure properties of graphs as they are filtered into sub-graphs. Specifically, we perform a *height filtration* that measures how many nodes are below a given height, and a *width filtration*, which measures the width of the tree as we prune the least significant nodes. These measures yield bounded, monotonic curves that summarize the subspace structure of the PCT, and can be used to perform hypothesis testing between PCTs. Our experimental results show the effectiveness of both summaries on synthetic and real-world data, including multilingual word embeddings and the latent space of neural networks.

Our main theoretical contribution shows that any distribution that can be parameterized using PCA or generalized PCA can be equivalently represented with a PCT with as many or fewer parameters. Our second theoretical contribution relates to our non parametric hypothesis test. We are able to show that with high probability (w.h.p.) our test will accurately detect clusters of correlated data, which we treat as a proxy for subspace structure. As a corollary of this result, we show that w.h.p. our algorithm will result in a degenerate tree (i.e., will produce the same result as PCA) whenever data is best represented by a single subspace.

Similar to PCA, our PCT enjoys several favorable properties. Firstly, the principal vector of each node is orthogonal to all of its ancestors and descendants, making projection of data onto each node of the tree efficient and decomposable. We also show that the singular value of a node is strictly less than or equal to that of its parent and ancestors. In standard PCA, the components corresponding to the largest singular values are the *subset* of components enabling optimal projection approximation of the datapoints. We prove that the monotonically decreasing singular values gives PCT a similar property, that the optimal approximating *subtree* of a PCT is the subtree with nodes with largest singular values. The decreasing singular values are also essential to enable the aforementioned width filtration topological measure.

2 Related Work

Subspace Clustering As mentioned, the task of subspace clustering is to partition a set of data such that each partition lies on the span of a low dimensional subspace. While some works are principally interested in the clustering / partitioning of points, such as in Motion Segmentation (Vidal, Tron, and Hartley 2008) and Hyperspectral Imaging (Li, Zhao, and Zhu 2023), others are more interested in using the learned subspace structure for downstream tasks. For example, subspace clustering has shown to be effective in matrix completion problems (Fan and Udell 2019), or as a feature extraction method (Hu et al. 2023; Kizaric and Pimentel-Alarcón 2022). Methods for performing subspace clustering utilize sparse self-expressiveness (Elhamifar and Vidal 2013), expectation maximization (Lipor et al. 2021), deep learning (Zhu et al. 2019), and fusion over the grassmannian (Mahmood and Pimentel-Alarcón 2022).

An important distinction in subspace clustering techniques is the difference between *axis-parallel* and *linear* methods. In axis-parallel methods such as CLIQUE (Agrawal et al. 1998) and (Yuan et al. 2013) the subspaces are assumed be be spanned by a discrete subset of the variables / features of the dataset. In linear methods, the subspaces are allowed to span any linear combination of the variables, in a continuous space known formally as the Grassmanian. As illustrated in figure 1, the nodes of Principal Component Trees represent an arbitrary direction in the data, and as such is a generalization of linear subspace clustering. Although it is comparatively well-studied, the discreet sample space in axis-parallel regimes enables the use of an entirely different class of algorithms, so we focus our comparisons only on linear methods.

Of particular interest to this paper are hierarchical subspace clustering techniques. These approaches come in top-down (Rafiezadeh Shahi et al. 2020; Wu et al. 2015) and bottom-up (Menon, Muthukrishnan, and Kalyani 2020) flavors. (Rafiezadeh Shahi et al. 2020) and (Wu et al. 2015) are similar to Principal Component Trees in that they are top-down methods which recursively partition the space into lower dimensional subspaces, and the termination of node splitting is based on a reconstruction error criteria.

While the construction of PCTs is fundamentally a hierarchical subspace clustering algorithm, it is the only one whose nodes represent a single basis, and that can generalize to PCA, disjoint subspaces, and intersecting subspaces. Similarly named similarity search methods like (McCartin-Lim, McGregor, and Wang 2012) and (Wichert 2009) exist, but these methods don't use subspace clustering.

Angle Distributions Established results on the distribution of pairwise angles in high dimensional space are important to PCT's adaptive splitting construction. These results were first established in (Cai, Fan, and Jiang 2013) and used to estimate intrinsic dimensionality in (Thordsen and Schubert 2022). (Menon, Muthukrishnan, and Kalyani 2020) uses pairwise angle distributions in subspace clustering, but for merging subspaces from the bottom-up, unlike our top-down approach. Hypothesis tests of angle uniformity are central to directional statistics, and are explored in (García-Portugués, Navarro-Esteban, and Cuesta-Albertos 2023).

Dictionary Learning and Sparse Coding Primarily a compression and denoising technique, Dictionary Learning and Sparse Coding seek to find a dictionary $\mathbf{V}$ and sparse codes $\mathbf{U}$ in order to minimize $||\mathbf{X} - \mathbf{UV}||_F + \alpha||\mathbf{U}||_1$. Implementations utilize autoencoders (Ng et al. 2011), alternating optimization (Tovsić and Frossard 2011), and other techniques. Structured Sparse Coding places additional constraints on $\mathbf{U}$ and $\mathbf{V}$ in order to form advantageous structures. For example, (Jenatton et al. 2010) force the non-zero coefficients to form a *pre-defined* tree hierarchy, and its optimization procedure promotes orthogonality in the branches of dictionary atoms in the tree. The atoms in (Szlam, Gregor, and LeCun 2012) are placed into overlapping groups which are learned from data, but don't form a tree.

Topological Data Analysis & Persistent Homology Be-

cause of its ability to discover robust patterns in arbitrary distributions like networks and point clouds, topological data analysis (TDA) has emerged as a promising approach in medical imaging and other areas (Chazal and Michel 2021). Persistent Homology is a sub-field of TDA that focuses on measuring properties of graphs as they are filtered down (Pun, Lee, and Xia 2022). While persistent homology is most commonly used to construct Birth-Death decomposition charts, Rips Filtration (Bauer 2021) and Graph Filtration (Lee et al. 2012) are robust approaches that enable theoretically sound statistical inference. The key property of Graph Filtration is that it tracks the first two Betti numbers of the graph, which are monotonic over edge filtration.

Of particular relevance to our analysis of PCTs are persistent homology based characterizations of tree structure. (Chung et al. 2015) build minimum spanning trees from structural covariance networks then apply graph filtration to create curves used to compare network structure. Persistent Homology has also been applied to tree-structured data. Both (Li et al. 2017) and (Kanari et al. 2016) filter the binary tree structure of neurons and brain arteries based on height / distance to the neuron's cell body. Persistent structures are then described by a birth-death composition chart. Our analysis of PCTs borrows this *height filtration* approach, but instead creates monotonic curves as in (Chung et al. 2015).

3 Principal Component Tree

Consider a data matrix $\mathbf{X}$ whose columns $\{\mathbf{x}\}$ represent samples in $\mathbb{R}^D$. A *principal component tree* $\mathcal{T}$ to model data $\mathbf{X}$ is defined by a set of nodes $\mathcal{N}_i$, $i \in \{1 \dots |\mathcal{T}|\}$, and a set of directed edges $\mathcal{E}_{ij}$ indicating that node $\mathcal{N}_i$ is the parent of node $\mathcal{N}_j$. Given node $\mathcal{N}_i$, we denote its (unique) parent as $\mathcal{P}_i$, its set of ancestors as $\mathcal{A}_i$, with $\mathcal{N}_i \in \mathcal{A}_i$, its set of children as $\mathcal{C}_i$, and its set of descendants as $\mathcal{D}_i$. We use $\mathcal{S}_i$ to denote the set of siblings of $\mathcal{N}_i$, defined as the set of nodes with the same parent as $\mathcal{N}_i$.

A node $\mathcal{N}_i$ represents two objects: (a) its associated *principal component*, given by a unit-length vector $\mathbf{v}_i \in \mathbb{R}^D$, and (b) its *singular value* $\sigma_i \geq 0$. Much like PCA, each node's $\mathbf{v}_i$ represents a direction of variation in the data, and its σ_i is proportional to the amount of variation along this direction. However, unlike PCA, these quantities describe variation directions of specific subsets of the data, not just the data as a whole. These subsets of data and their particular variation directions are determined by the nodes and their lineage. To see this, define the *ancestral basis* associated to node $\mathcal{N}_i$ as

$$\mathbf{V}_i \;:=\; \{\mathbf{v}_i \,|\, \mathcal{N}_i \in \mathcal{A}_i\}.$$

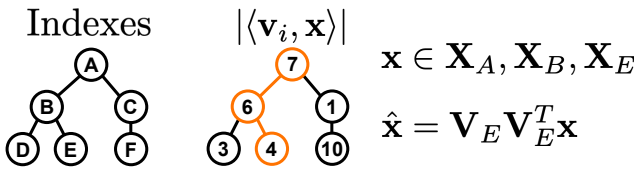


Figure 2: Sample tree and data assignments for one point. $\mathbf{x}$ "chooses" nodes greedily, starting from the root down.

This basis spans a subspace that approximates a subset of the samples in $\mathbf{X}$. Specifically, we assign each node of the tree a subset of the data $\mathbf{X}_i$, defined recursively as:

$$\mathbf{X}_i \;:=\; \left\{ \mathbf{x} \in \mathbf{X}_{\mathcal{P}_i} \;\middle|\; \arg\max_{j \in \mathcal{S}_i} |\langle \mathbf{v}_j, \mathbf{x} \rangle| = i \right\}.$$

Intuitively, the subset of columns assigned to node $\mathcal{N}_i$ is the subset of columns that were assigned to its parent, and that are closer to the principal component of $\mathcal{N}_i$ than to that of any of its siblings. See Figure 2 for a one sample example.

Notice that under these definitions, the bases and data assignments of ancestors and descendants are nested, i.e., for any $\mathcal{N}_j \in \mathcal{A}_i$,

$$\text{span}(\mathbf{V}_j) \subseteq \text{span}(\mathbf{V}_i) \;\text{ and }\; \mathbf{X}_j \supseteq \mathbf{X}_i$$

That is, the basis for an ancestor is a subspace of the basis for all its descendants, and the data assigned to an ancestor is a superset of the data assigned to its descendants.

Approximating Data

We will now show how Principal Component Trees can be used to approximate data in a similar fashion to PCA. That is, to create an approximation $\hat{\mathbf{X}}$ of $\mathbf{X}$. Whereas PCA approximations can be computed by a single matrix operation, PCT approximations are a union of approximations given individual nodes of their respective assigned data. For a single node, its *node approximation* and *node residual* are:

$$\hat{\mathbf{X}}_i = \mathbf{V}_i \mathbf{V}_i^T \mathbf{X}_i \quad \text{and} \quad \mathbf{R}_i := \mathbf{X}_i - \hat{\mathbf{X}}_i$$

respectively. The node approximation is simply a node's assigned data projected onto its anscestral basis. Ordinary least squares projection would require that we also compute $(\mathbf{V}_i \mathbf{V}_i^T)^{-1}$, but our tree construction actually gives an orthonormal basis for all $\mathbf{V}_i$ (Property 6.1). The node residuals are used extensively in our tree construction; the descendants of $\mathcal{N}_i$ are learned to best approximate $\mathbf{R}_i$. Intuitively, the number of children of $\mathcal{N}_i$ is the number of *disjoint* subspace clusters detectable in $\mathbf{R}_i$ and the principal components of those children are the first singular vectors of those disjoint subspaces.

To compute the approximation of the entire dataset, we make use of the fact that for leaves of the tree (nodes with no children), data assignments are disjoint. Furthermore, the subspaces defined by leaf nodes are not nested. More exactly, for two leaf nodes $\mathcal{N}_\ell$ and $\mathcal{N}_g$:

$$\text{span}(\mathbf{V}_\ell) \not\subseteq \text{span}(\mathbf{V}_g) \quad \text{and} \quad \mathbf{X}_\ell \cap \mathbf{X}_g = \emptyset$$

Now that we have single node approximations and have shown that data is disjointly partitioned between leaf nodes, we can compute the full approximation of some data $\mathbf{X}$ as:

$$\hat{\mathbf{X}} := \bigcup_{\ell=1}^{|\mathcal{N}_L|} \hat{\mathbf{X}}_\ell$$

Where $\mathcal{N}_\mathcal{L}$ is the set of all leaves of the tree.

4 Subspace Structure Test

Testing the Presence of Subspace Clusters

Key to our construction of the Principal Component Tree is our adaptive splitting algorithm, which builds the tree top to bottom, expanding node $\mathcal{N}_i$ with either (a) a single child at a time, if its data residuals $\mathbf{R}_i$ show no sign of subspace structure, or (b) with multiple children, if we detect sufficient subspace structure in $\mathbf{R}_i$.

To make this determination we use the Cramér-von Mises test to compare the distribution of the angles of the residuals $\mathbf{R}_i$ with the known angle distribution under uniformity on the hypersphere. We motivate the use of this test as follows. If data is subspace clustered, there will be directions in the space that are crowded with points and directions that are comparatively empty. This non-uniform use of the space will cause the distribution of angles between points of subspace clustered data to differ from the known distribution for uniform points. We treat the degree of angle non-uniformity as a proxy for the degree of subspace clusteredness.

This measure of subspace clusteredness is summarized in a single p-value p where a small p indicates subspace clustering. To compute this p-value, we require three things. 1) The empirical cumulative distribution function (CDF) $\hat{\mathbb{P}}_D(C)$ which describes the observed distribution of angles of our D-dimensional residuals. 2) The CDF $\mathbb{P}_D(C)$ which defines the expected distribution of angles under the null hypothesis of no subspace structure i.e. uniformity. 3) The Cramér-von Mises test statistic Q which quantifies the difference between $\hat{\mathbb{P}}_D(C)$ and $\mathbb{P}_D(C)$, and therefore the degree of subspace clustering.

For notational clarity, we henceforth discuss the test as applied to a generic data matrix $\mathbf{X}$ whose columns $\{\mathbf{x}_1 \dots \mathbf{x}_k \dots \mathbf{x}_j \dots \mathbf{x}_N\}$ represent samples in $\mathbb{R}^D$.

To perform this test, we first measure the *angle distribution* of our data. More precisely, we measure the distribution of absolute cosine similarities between pairs of points of $\mathbf{X}$. Taking all pairs, we compute the $\frac{N(N-1)}{2}$ angles of $\mathbf{X}$,

$$\angle\mathbf{X} := \left\{ \frac{|\langle \mathbf{x}_k, \mathbf{x}_j \rangle|}{||\mathbf{x}_k||\,||\mathbf{x}_j||} \;\middle|\; k < j \right\} \quad \angle x_i \in [0,1].$$

We use *absolute* cosine similarity because subspaces pass *through* the origin. All points on a line through the origin to should be deemed similar, regardless of whether they are on the same side of the origin. We then describe the distribution of angles via the empirical distribution function,

$$\hat{\mathbb{P}}_D(C) := \frac{1}{N(N-1)} \sum_{i=1}^{N(N-1)} \mathbb{1}\{\angle x_i \le C\}$$

In order to determine if our data $\mathbf{X}$ exhibits subspace clusteredness, we need to compare the observed distribution of angles to an expected null distribution. In this test, our null hypothesis is that our data has *angular uniformity*; that the unit length normalized data is sampled uniformly from the hypersphere embedded in $\mathbb{R}^D$. For example, the isotropic gaussian distribution has angular uniformity.

Thankfully, the expected distribution of pairwise angles on the hypershpere has already been derived by (Cai, Fan, and Jiang 2013) for standard angles $\theta \in [0, 2\pi]$ and (Thordsen and Schubert 2022) for absolute cosine similarity $C \in [0,1]$. The latter is a simple transformation of the univariate beta distribution, whose CDF is:

$$\mathbb{P}_D(C) = 2\,\text{Beta}_{\text{CDF}}\left(\tfrac{1+C}{2};\ \alpha = \tfrac{D-1}{2}, \beta = \tfrac{D-1}{2}\right) - 1.$$

With the observed and expected/null distributions in hand, we can finally compare them via the Cramér-von Mises test. This test quantifies the distance between two distributions into the test statistic

$$Q = N(N-1) \int_0^1 \left(\mathbb{P}_D(C) - \hat{\mathbb{P}}(C_D)\right)^2 dC.$$

The known domain of C allows us to bound the otherwise improper integral of the CvM statistic. Computation of the final p-value p is given in (CSöRgő and Faraway 1996), equation 1.8.

Practical Considerations. For simplicity and interpretability, we leave three important implementation details about this and the following test to the technical appendix. 1) In order to reasonably compare angle distributions to those of hyper*spheres*, we first whiten $\mathbf{X}$ before taking the angle distribution. 2) This *explicit* whitening makes the expected angle distribution $\mathbb{P}_D(C)$ dependent on N, so we used monte-carlo simulation to estimate $\mathbb{P}_D(C)$ for $N < 100D$. 3) Whenever comparing the subspace clusteredness of two data matrices, we first project both to a common dimension.

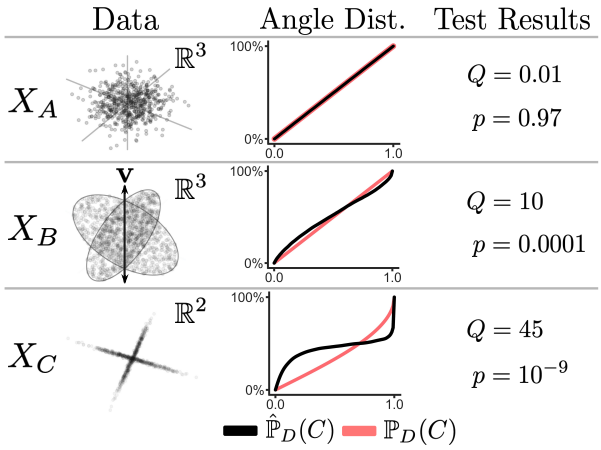


Figure 3: Three datasets, their angle distribution, and subspace clustering test results. $\mathbf{X}_A \in \mathbb{R}^3$: An isotropic normal distribution. $\mathbf{X}_B \in \mathbb{R}^3$: A data lying on 2 2D subspaces, with a 1d intersection, labeled $\mathbf{v}$. $\mathbf{X}_C \in \mathbb{R}^2$: A union of 2 1D disjoint subspaces.

Testing an Intersection of Subspace Clusters

In the previous subsection, we state that we will split a node $\mathcal{N}_i$ in two if we detect subspace clusters in its residuals $\mathbf{R}_i$. This is a simplification; if we only do this, we may split nodes *prematurely* and leave our PCT construction unable to find intersections of subspaces. For example, take $\mathbf{X}_B$ and $\mathbf{X}_C$ in Figure 2. Both $\mathbf{X}_B$ and $\mathbf{X}_C$ have a very small p-value, small enough to correctly conclude there is subspace

Algorithm 1: CLUSTER-TEST

Input: $\mathbf{X}_{D\times K}$
Output: p: The p-value of the Cramér-von Mises test; the probability that $\mathbf{X}$ has angular uniformity. Sufficiently low p indicates subspace clustering.

Let $r = \texttt{RANK}(\mathbf{X})$.
Let $\mathbf{X}_w = \texttt{WHITEN}(\mathbf{X})$.
Let $\mathbf{X}_{unit} = \frac{\mathbf{x}_k}{||\mathbf{x}||_k}$ for $\mathbf{x}_k \in \mathbf{X}_w$.
Let $\mathbf{A}_{K\times K} = |\mathbf{X}_{unit}\mathbf{X}_{unit}^T|$
Let $\angle\mathbf{X} = \mathbf{A}_{ij}$ where $i > j$. $\{\angle x_i \in [0, 1]\}$
Let $p = \texttt{CVM-TEST}(\hat{P} = \angle\mathbf{X}, P = \mathbb{P}(C_r))$
RETURN p =0

Algorithm 2: EXPAND-NODE

Input: A node $\mathcal{N}_i$ and residuals $\mathbf{R}_i$.
Output: Either one or two child nodes of $\mathcal{N}_i$.

$\texttt{V1}(\mathbf{X}) \rightarrow \mathbf{v}_1, \sigma_1$ {$\mathbf{X}$'s 1_{st} singular vector and value}.
Let $\mathbf{v}_1, \sigma_1 = \texttt{V1}(\mathbf{R}_i)$.
Let $isClustered$ = $\texttt{CLUSTER-TEST}(\mathbf{R}_i) < \alpha_{test}$
Let $hasIntersection$ = $\texttt{INTERSECT-TEST}(\mathbf{R}_i, \mathbf{v}_1)$
if $isClustered$ **and not** $hasIntersection$ **then**
  $\{\mathbf{R}_{i,A}, \mathbf{R}_{i,B}\} \leftarrow \texttt{SUBSPACE-CLUSTER}(\mathbf{R}_i)$
  $\mathbf{v}_A, \sigma_A = \texttt{V1}(\mathbf{R}_{i,A})$; $\mathbf{v}_B, \sigma_B = \texttt{V1}(\mathbf{R}_{i,B})$
  RETURN $\texttt{NODE}(\mathbf{v}_A, \sigma_A, \mathcal{N}_i), \texttt{NODE}(\mathbf{v}_B, \sigma_B, \mathcal{N}_i)$
end if
RETURN $\texttt{NODE}(\mathbf{v}_1, \sigma_1, \mathcal{N}_i)$ =0

clustering in the data. However, if we split $\mathbf{X}_B$ right away, we would miss the fact that the two subspaces have a prominent 1D intersection (labeled $\mathbf{v}$).

In order to build PCTs with the capacity to represent intersecting subspaces, we now present a test that extends on the previous subsection. This test evaluates whether or not potential subspace clusters plausibly intersect along a given axis / basis $\mathbf{v}$. We propose that if some data $\mathbf{X}$ projected onto the null space of $\mathbf{v}$ appears *more* subspace clustered than the original data, then $\mathbf{v}$ is a plausible intersection of subspace clusters present in $\mathbf{X}$. More exactly, our test $\texttt{INTERSECT-TEST}(\mathbf{X}, \mathbf{v})$ returns `true` if

$$\texttt{CLUSTER-TEST}(\mathbf{X} - \mathbf{v}\mathbf{v}^T\mathbf{X}) < \texttt{CLUSTER-TEST}(\mathbf{X})$$

Back to the example of Figure 3, notice that $\mathbf{X}_C$ is actually the projection of $\mathbf{X}_B$ onto the null space of $\mathbf{v}$, and that $\mathbf{X}_C$ has an even smaller p-value than $\mathbf{X}_B$. In this case, we would conclude that $\mathbf{v}$ is a plausible subspace intersection of $\mathbf{X}_B$.

5 Methodology

Building a Principal Component Tree

Now that we have defined the structure of a Principal Component Tree in section 3 and two important building blocks in section 4, we can present our algorithm to construct a PCT. Put simply, we build a tree by selecting the *leaf* node with the largest residuals

$$\mathcal{N}_{next} := \arg\max_{i\in\mathcal{N}_\mathcal{L}} ||\mathbf{R}_i||_F$$

and expanding it by giving it one or two children. The process is illustrated in Figure 4. We start with an "empty" root that describes no variation in the data, and we expand nodes until we have reached our target size.

The two tests of subspace structure presented in section 4 are used to determine how many children to grant $\mathcal{N}_{next}$. In essence, if $\texttt{CLUSTER-TEST}(\mathbf{R}_{next})$ finds no subspace clustered structure in $\mathbf{R}_{next}$, we give it one child. We also give it one child if we deem that $\mathbf{v}$, the first singular vector of $\mathbf{R}_{next}$ is common to subspace clusters in $\mathbf{R}_{next}$, as determined by $\texttt{INTERSECT-TEST}(\mathbf{R}_{next}, \mathbf{v})$. This case is like "digging deeper" through the space in the hope that there are even more disjoint subspace clusters somewhere in the null space of $\mathbf{V}_{next}\cup\mathbf{v}$. We will only split a node in two when we believe that $\mathbf{R}_{next}$ lies on *disjoint* subspace clusters: When the data is sufficiently subspace clustered, and $\mathbf{v}$ is not an intersection. See algorithm 2 for a formal specification.

By only evaluating the leading singular vector of $\mathbf{R}_{next}$ as a potential subspace intersection of $\mathbf{R}_{next}$, we ensure three desireable properties: a) Only *strong* intersections will be detected, and not intersections that may arise from noise. b) The non-decreasing singular value property will be enforced. This property is needed to select the optimal subtree of $\mathcal{T}$, and for width-filtration. c) The PCT will reduce to PCA in the case of gaussian data.

The PCT construction algorithm uses a few hyperparameters. Firstly, we greedily build the tree up to a maximum number of nodes $|\mathcal{T}|_{max}$. There is no inherent maximum size of the tree, but the maximum height of the tree is the data dimension D. We also specify a significance level $\alpha_{test} \leq 0.05$ which specifies how confident the test must be that there is subspace clustered / angle non-uniformity structure in order to split a node in two. This parameter directly influences the shape of the tree; a smaller α_{test} means nodes are more likely to have only one child, resulting in taller trees. This parameter is used in theorem 6.1 to lower bound the probability that the PCT will reduce to PCA when trained on gaussian / elipical data. If $\alpha_{test} = 0$, no nodes will be split, and the PCT will always reduce to PCA.

Finally, when we split a node, we use an arbitrary binary subspace clustering algorithm to partition its residuals. We denote this $\mathcal{SC}(\mathbf{R}_i) \rightarrow \{\mathbf{R}_{i,A}, \mathbf{R}_{i,B}\}$. The basis vector and singular value of the two new nodes emerging from this split are the first singular values and vectors of $\mathbf{R}_{i,A}$ and $\mathbf{R}_{i,B}$. As a structure, the PCT supports arbitrary numbers of children, but we focus only on the binary case in this paper.

Persistent Homology Analysis

Thus far, we have defined the structure and properties of Principal Component Trees and outlined an algorithm to construct them using two novel hypothesis tests. In this section, we discuss how to analyze and compare the structure of PCTs so that we may use them as a tool to inspect the structure of high-dimensional data. To this end, we introduce two interpretable topological measures that describe the shape of

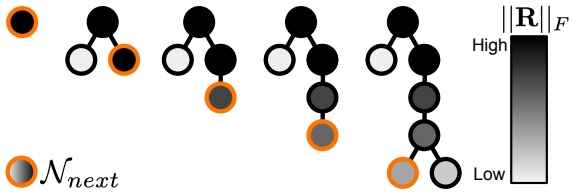


Figure 4: A sample PCT as it is constructed.

PCTs. Inspired by (Chung et al. 2015), both of these measures take the form of bounded, monotonic curves measuring the size or shape of a tree as nodes are pruned away.

As a guiding principal, the more subspace clustered $\mathbf{X}$ is, the more nodes will be split into multiple children, and the tree will be *wider*; it will have more leaves. Conversely, the tree will be *taller* the less subspace clustered $\mathbf{X}$ is. In the case where $\mathbf{X}$ follows a multivariate normal distribution, the tree will be *degenerate* and maximally tall. Our two topological measures quantify this intuition by defining curves that describe the height and width of the tree. Assume that the empty root node is left out of calculations.

Height Curves. The first topological measure of a PCT is the tree's *height curve*. Put simply, the height curve characterizes the distribution of node heights, similar to a CDF. For a tree $\mathcal{T}$, it is defined as:

$$H_{\mathcal{T}}(n) = \sum_{i=1}^{|\mathcal{T}|} \mathbb{1}\{h_i \leq n\}.$$

Intuitively, the height curve will be higher for more subspace clustered trees. More subspace clusters leads to more nodes with multiple children which leads to more nodes at lower heights. The height curve is lower bounded by $H_{\mathcal{T}}(n) = n$. This occurs only for degenerate / "PCA" PCTs, where there is exactly one node at each height. The height curve is upper bounded by $H_{\mathcal{T}}(n) = |T|$. Only trees where all $|T|$ nodes are children of the root will have this curve. In this case, the tree models a union of $|T|$ one dimensional disjoint subspace clusters.

Width Curves. We also propose *width curves* as a topological descriptor of PCT shape. For this curve, we measure tree width (number of leaves) of the subtree of $\mathcal{T}$ consisting of only the most significant nodes, according to their singular values. Formally, this subtree consisting of the n most significant nodes is

$$\mathcal{T}_{[n]} = \{\mathcal{N}_i | \sigma_i \geq \sigma_{(n)}\}, \{\mathcal{E}_{ij} | \sigma_i, \sigma_j \geq \sigma_{(n)}\} \qquad (1)$$

where $\sigma_{(n)}$ is the n_{th} largest singular value of nodes in $\mathcal{T}$. In property 6.3 we prove that $\mathcal{T}_{[n]}$ is the best approximating subtree of $\mathcal{T}$. With this subtree defined, a PCT's width curve is then

$$W_{\mathcal{T}}(n) = \sum_{i=1}^{|\mathcal{T}|} \mathbb{1}\{i \in \mathcal{T}_{[n]} \ \wedge \ \mathcal{C}_i \cap \mathcal{T}_{[n]} = \emptyset\}.$$

That is, the number of nodes who are both in $\mathcal{T}_{[n]}$ and leaves of $\mathcal{T}_{[n]}$. It may be easier to imagine recording the width of a tree as the least significant nodes are removed.

Similar to height curves, the width curves will be higher for more subspace clustered trees. Also like height curves, width curves are lower bounded by $W_{\mathcal{T}}(n) = 1$ and upper bounded $W_{\mathcal{T}}(n) = n$ for degenerate trees and maximally wide trees, respectively.

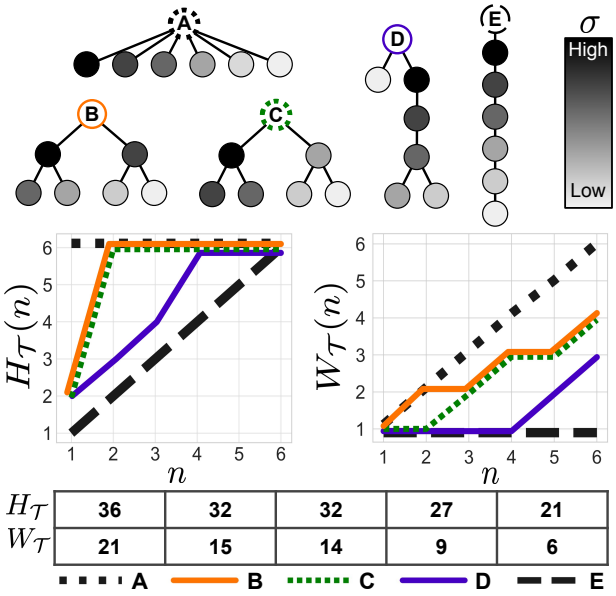


Figure 5: Five example PCTs with 6 nodes each, and corresponding height/width curves. Shading indicates singular value. A and E are maximally and minimally subspace clustered. B and C differ only in singular values, which is reflected in width curve only.

Area Under Curves We have just shown that $H_{\mathcal{T}}(n)$ and $W_{\mathcal{T}}(n)$ are monotonic and maximized under a large degree of subspace clusteredness. Therefore, the area under each curve is an effective one number summary of the subspace-clusteredness of the dataset used to construct a tree. The area under height and width curves are

$$H_{\mathcal{T}} = \sum_{n=1}^{|\mathcal{T}|} H_{\mathcal{T}}(n) \quad \text{and} \quad W_{\mathcal{T}} = \sum_{n=1}^{|\mathcal{T}|} W_{\mathcal{T}}(n).$$

In section 7 we perform hypothesis testing to show differences in subspace clusteredness between various datasets.

6 Main Theoretical Results

In this section we present two theoretical results that demonstrate the representational capacity of Principal Component Trees and the ability of their construction to perform PCA under correct conditions. Detailed proofs in the Appendix.

Theorem 6.1 *Any union of subspaces, or distribution lying on a union of subspaces can be equivalently represented by a PCT using the same or fewer parameters.*

Proof Sketch. We prove this result for a union of two subspaces. Any mixture of 2 d-dimensional subspaces will require $2d$ vector parameters. But if these subspaces have a c-dimensional intersection, we can a) find that intersection analytically, and b) create a PCT representing that same mixture with only $2d - c$ parameters. □

Theorem 6.2 *When presented data following a multivariate normal distribution, PCT construction will reduce to PCA, yielding a degenerate tree with probability at least* $(1-\alpha_{test})^D$.

Proof Sketch. If the data is truly multivariate normal, then all projection residuals will also be multivariate normal. Therefore, when we test for subspace clustering in $\mathbf{R}_i$ for any node, we will erroneously split a node with probability α_{test}. Repeating this test D times, we will never split a node with probability $(1-\alpha_{test})^D$. □

We also prove that our PCT learning algorithm leads to two simple but useful properties of PCTs. Firstly, nodes are orthogonal to their ancestors, making calculation of data approximations fast and decomposable. Secondly, node singular values decrease with height. Together these two properties are used in property 6.3, which provides a simple and efficient way to select the best approximating subtree of an existing tree by selecting its most significant nodes. This is like selecting the best principal components of PCA.

Property 6.1 *The singular vector of any node is orthogonal to those of its ancestors. Formally,*

$$\mathbf{v}_i \perp \mathbf{v}_j \;\; \forall j \in \mathcal{A}_i$$

as a consequence, all ancestral bases are orthonormal.

Proof Sketch. This property arises from the tree construction. As a singular vector of $\mathbf{R}_i$, Each $\mathbf{v}_i$ is somewhere in the null space of $\mathbf{v}_{\mathcal{P}_i}$, its parent's basis, so $\mathbf{v}_i \perp \mathbf{v}_{\mathcal{P}_i}$. By induction, it is orthogonal to all its ancestors. □

Property 6.2 *A node's singular value is less than or equal to its ancestors. Formally,*

$$\sigma_i \leq \sigma_j \;\; \forall j \in \mathcal{A}_i.$$

Proof Sketch. We show this by contradiction. Consider a node $\mathcal{N}_i$ and its parent $\mathcal{P}_i$. Let $\overline{\mathbf{R}_{\mathcal{P}_i}}$ be the residual used to create the parent $\mathcal{P}_i$. Both $\sigma_{\mathcal{P}_i}$ and σ_i are derived from $\overline{\mathbf{R}_{\mathcal{P}_i}}$, but $\sigma_{\mathcal{P}_i}$ is its singular value. If $\sigma_i > \sigma_{\mathcal{P}_i}$, it would violate the Eckart-Young theorem. Therefore, $\sigma_i \leq \sigma_{\mathcal{P}_i}$. By induction, $\sigma_i \leq \sigma_j, j \in \mathcal{A}_i$. □

Property 6.3 *The best size n approximating subtree of $\mathcal{T}$ is given by $\mathcal{T}_{[n]}$ (equation 1), i.e. selecting the n nodes with largest singular values. More exactly,*

$$\arg\min_{\mathcal{T}^* \subset \mathcal{T}} ||\mathbf{X} - \hat{\mathbf{X}}_{\mathcal{T}^*}||_F := \mathcal{T}_{[n]}$$

Proof Sketch. We prove this in three parts. 1) We show that property 6.1 means that the approximation error is inversely proportional to the sum of singular values in the tree. 2) We can therefore minimize the approximation error by selecting the nodes with largest singular values. 3) Property 6.2 implies that $\mathcal{T}_{[n]}$ will share the root of $\mathcal{T}$, and will have no *gaps*, that is $\mathcal{N}_i \in \mathcal{T}_{[n]} \implies \mathcal{N}_j \in \mathcal{T}_{[n]} \;\forall\; \mathcal{N}_j \in \mathcal{A}_i$. □

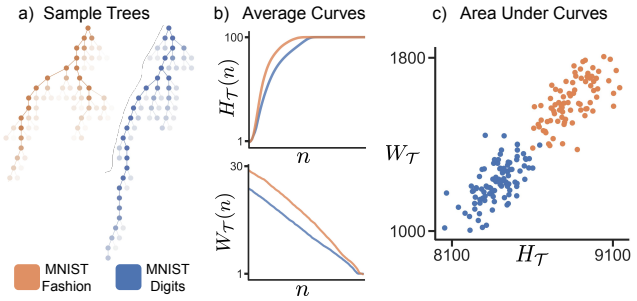


Figure 6: MNIST Digits vs Fashion. a): Example PCTs learned from each dataset. Singular values shaded. b) The average height & width curve from bootstrap sampling. c) Area under width & height curves among bootstrap samples.

7 Experiments

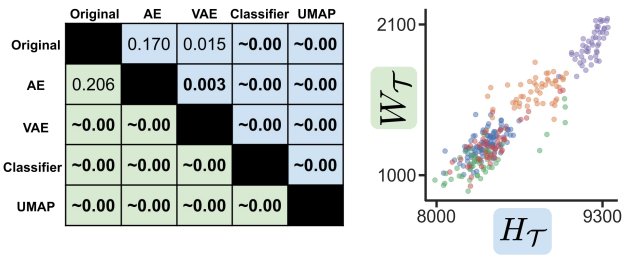
With our persistent homology measures of tree shape defined, we now analyze the structure of some real world high-dimensional datasets by comparing the structure of Principal Component Trees learned from those datasets. Our goal is not to discriminate between samples, but to conclude something about the clusteredness of the datasets themselves.

In order to do this, we: 1) take the datasets we wish to compare, and pre-process them by normalizing, but not completely whitening their singular values. This way, differences in PCT structure are purely the result of subspace structuredness, not anisotropic covariance. 2) Bootstrap sample each dataset into a number of equally sized sub-datasets. Differently sized bootstrap samples between datasets would impact the hypothesis test used to construct the trees. 3) Construct a PCT and calculate $H_\mathcal{T}$ and $W_\mathcal{T}$ for each bootstrap sample. 4) Compare $H_\mathcal{T}$ and $W_\mathcal{T}$ for each dataset using a wilcoxon rank-sum test. We also plot these metrics on scatter plots.

MNIST Digits vs Fashion. For our first experiment, we compare the widely known MNIST Digits and Fashion datasets (Deng 2012; Xiao, Rasul, and Vollgraf 2017). On paper, they are very similar: 60,000 32x32 images with 10 balanced classes each.

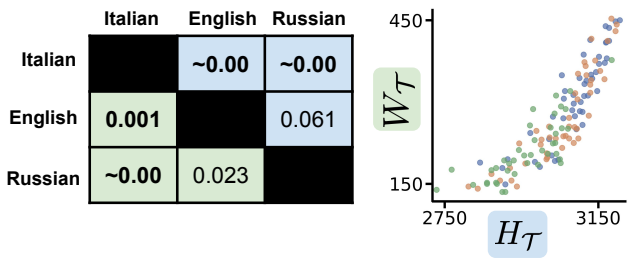
When we compare the subspace clusteredness of the digits and fashion datasets, the results are clear: MNIST Fashion is much more subspace clustered. The results in Figure 4 tell the story. Both the average height and width curve in 4b is higher for the fashion dataset, and this is reflected in areas under each curve. Anecdotally, the sample tree (4c) for MNIST digits shows a single dominant "branch" of nodes whereas the fashion tree has two dominant branches.

Neural Network Latent Space & Non-Linear Dimensionality Reduction "What do neural networks learn?" is a long running question in Machine Learning, and there is no shortage of answers in the literature. In this experiment, we answer this question by analyzing the latent space of different neural network architectures via Principal Component Trees. Specifically, we compare the latent space of an Multi Layer Perceptron classifier, simple auto encoder (AE), and variational auto encoder (VAE) which have been trained on the MNIST Digits dataset. We also include the digits after



	Original	AE	VAE	Classifier	UMAP
$H_\mathcal{T}$	8383	8430	8363	8821	9216
$W_\mathcal{T}$	1227	1216	1107	1677	1954

Figure 7: Subspace Clusterness of latent space induced by neural network architectures. Top-Left: Pairwise p-values for rank-sum hypothesis tests between latent spaces. Top-Right: $H_\mathcal{T}$ plotted against $W_\mathcal{T}$ for bootstrap samples. Bottom: Mean of $H_\mathcal{T}$ and $W_\mathcal{T}$ for the five latent spaces.



	Italian	English	Russian
$H_\mathcal{T}$	2976	3046	3090
$W_\mathcal{T}$	219	271	315

Figure 8: PCT Analysis of word embeddings. Top-Left: Pairwise p-values for rank-sum hypothesis tests between language embeddings. Top-Right: $H_\mathcal{T}$ plotted against $W_\mathcal{T}$ for bootstrap samples. Bottom: Mean of $H_\mathcal{T}$ and $W_\mathcal{T}$ for the three languages

UMAP dimensionality reduction for comparison. All latent spaces are 64 dimensional.

To accomplish this, for each bootstrap dataset we: 1) Train a network on that bootstrap dataset. 2) Transform the bootstrap dataset by passing it through the first 2-3 layers of the network. 3) Construct a PCT from this transformed dataset.

The results of this experiment are summarized in Figure 7. We first notice that UMAP produces the most clustered embeddings, which is expected given it's objective function and effectiveness in enhancing high-dimensional clusters. Amongst the Neural Network latent spaces, the latent space of the classification model is much more subspace clustered than the original data or the other architectures. This is ultimately unsurprising. The end goal of this model is to transform the original data into one-hot vectors which are maximally clustered under perfect accuracy. We posit that this objective leads to more subspace clustering in the intermediate layers of the model.

Looking at the standard and variational autoencoder (labeled AE & VAE), the results are consistent with the objective of these models. The subspace clusteredness of the original data is preserved in the latent space of the simple autoencoder, likely becasue it simply tried to accurately reproduce the training data. For the variational autoencoder, the latent space is less subspace clustered than the original. This is expected as the loss function of VAEs encourages the latent space to take the form of an isotropic gaussian function. For this comparison, it appears that $W_\mathcal{T}$ is a more sensitive measure than $H_\mathcal{T}$.

Multilingual word embeddings. For our final experiment, we compare the subspace clusteredness of multilingual word embeddings in three languages. We use the embeddings presented in (Ferreira, Martins, and Almeida 2016), where each language's embedding were trained on the TED 2020 dataset (Cettolo, Girardi, and Federico 2012), where the same TED talks are translated into each language. To ensure our word embeddings cover similar concepts, we examine the 1000 most common words from each language.

Under these circumstances, we would expect the embeddings to be structurally similar across languages, but that's not what we see in Figure 5. Each of the three kinds of embeddings have different degrees of subspace clusteredness, with Italian embeddings being the least and Russian embeddings being the most clustered.

8 Conclusion

In conclusion, this paper introduced Principal Component Trees (PCTs) as a framework for identifying mixtures of components that collectively represent the subspace structure within high-dimensional datasets. The proposed PCTs extend conventional low-dimensional models such as PCA and union of subspaces by incorporating a graph structure. Each node in the PCT corresponds to a principal component, with edges representing the necessary combinations of components required to form a subspace that approximates specific portions of the data.

To construct PCTs, two angle-distribution hypothesis tests are introduced to detect intersecting subspace clusters within the data. Additionally, for the analysis, comparison, and selection of optimal PCT models, we use two novel and interpretable persistent homology measures to examine their shape. The constructed PCTs exhibit two interesting properties: ancestral orthogonality and non-decreasing singular values, which in turn facilitate the computation of persistent homology measures.

Theoretical results demonstrate that learning PCTs reduces to PCA under multivariate normality, and that Principal Component Trees serve as efficient parameterizations for intersecting unions of subspaces. Finally, we showcase the practical applicability of PCTs by analyzing neural network latent spaces, word embeddings, and reference image datasets. This research advances our understanding of subspace modeling and offers a powerful tool for dissecting complex high-dimensional datasets in various domains.

References

Agrawal, R.; Gehrke, J.; Gunopulos, D.; and Raghavan, P. 1998. Automatic subspace clustering of high dimensional data for data mining applications. In *Proceedings of the 1998 ACM SIGMOD international conference on Management of data*, 94–105.

Bauer, U. 2021. Ripser: efficient computation of Vietoris–Rips persistence barcodes. *Journal of Applied and Computational Topology*, 5(3): 391–423.

Cai, T. T.; Fan, J.; and Jiang, T. 2013. Distributions of angles in random packing on spheres. *Journal of Machine Learning Research*, 14: 1837.

Cettolo, M.; Girardi, C.; and Federico, M. 2012. Wit3: Web inventory of transcribed and translated talks. In *Proceedings of the Conference of European Association for Machine Translation (EAMT)*, 261–268.

Chazal, F.; and Michel, B. 2021. An introduction to topological data analysis: fundamental and practical aspects for data scientists. *Frontiers in artificial intelligence*, 4: 667963.

Chung, M. K.; Hanson, J. L.; Ye, J.; Davidson, R. J.; and Pollak, S. D. 2015. Persistent homology in sparse regression and its application to brain morphometry. *IEEE transactions on medical imaging*, 34(9): 1928–1939.

CSöRgő, S.; and Faraway, J. J. 1996. The exact and asymptotic distributions of Cramér-von Mises statistics. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 58(1): 221–234.

Deng, L. 2012. The mnist database of handwritten digit images for machine learning research [best of the web]. *IEEE signal processing magazine*, 29(6): 141–142.

Elhamifar, E.; and Vidal, R. 2013. Sparse subspace clustering: Algorithm, theory, and applications. *IEEE transactions on pattern analysis and machine intelligence*, 35(11): 2765–2781.

Fan, J.; and Udell, M. 2019. Online high rank matrix completion. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 8690–8698.

Ferreira, D. C.; Martins, A. F.; and Almeida, M. S. 2016. Jointly learning to embed and predict with multiple languages. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2019–2028.

García-Portugués, E.; Navarro-Esteban, P.; and Cuesta-Albertos, J. A. 2023. On a projection-based class of uniformity tests on the hypersphere. *Bernoulli*, 29(1): 181–204.

Hotelling, H. 1933. Analysis of a complex of statistical variables into principal components. *Journal of educational psychology*, 24(6): 417.

Hu, P.; Li, X.; Lu, N.; Dong, K.; Bai, X.; Liang, T.; and Li, J. 2023. Prediction of New-Onset Diabetes after Pancreatectomy with Subspace Clustering based Multi-view Feature Selection. *IEEE Journal of Biomedical and Health Informatics*.

Jenatton, R.; Mairal, J.; Obozinski, G.; and Bach, F. R. 2010. Proximal methods for sparse hierarchical dictionary learning. In *ICML*, volume 1, 2. Citeseer.

Jolliffe, I. T. 2002. *Principal component analysis for special types of data*. Springer.

Kanari, L.; Dłotko, P.; Scolamiero, M.; Levi, R.; Shillcock, J.; Hess, K.; and Markram, H. 2016. Quantifying topological invariants of neuronal morphologies. *arXiv preprint arXiv:1603.08432*.

Kizaric, B. A.; and Pimentel-Alarcón, D. L. 2022. Classifying Incomplete Data with a Mixture of Subspace Experts. In *2022 58th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, 1–8. IEEE.

Lee, H.; Kang, H.; Chung, M. K.; Kim, B.-N.; and Lee, D. S. 2012. Weighted functional brain network modeling via network filtration. In *NIPS Workshop on Algebraic Topology and Machine Learning*, volume 3. Citeseer.

Li, Q.; Zhao, X.; and Zhu, H. 2023. Semi-supervised Sparse Subspace Clustering Based on Re-weighting. *Engineering Letters*, 31(1).

Li, Y.; Wang, D.; Ascoli, G. A.; Mitra, P.; and Wang, Y. 2017. Metrics for comparing neuronal tree shapes based on persistent homology. *PloS one*, 12(8): e0182184.

Lipor, J.; Hong, D.; Tan, Y. S.; and Balzano, L. 2021. Subspace clustering using ensembles of k-subspaces. *Information and Inference: A Journal of the IMA*, 10(1): 73–107.

Mahmood, U.; and Pimentel-Alarcón, D. 2022. Fusion Subspace Clustering for Incomplete Data. In *2022 International Joint Conference on Neural Networks (IJCNN)*, 1–8. IEEE.

McCartin-Lim, M.; McGregor, A.; and Wang, R. 2012. Approximate principal direction trees. *arXiv preprint arXiv:1206.4668*.

Menon, V.; Muthukrishnan, G.; and Kalyani, S. 2020. Subspace clustering without knowing the number of clusters: A parameter free approach. *IEEE Transactions on Signal Processing*, 68: 5047–5062.

Ng, A.; et al. 2011. Sparse autoencoder. *CS294A Lecture notes*, 72(2011): 1–19.

Pearson, K. 1901. LIII. On lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin philosophical magazine and journal of science*, 2(11): 559–572.

Pun, C. S.; Lee, S. X.; and Xia, K. 2022. Persistent-homology-based machine learning: a survey and a comparative study. *Artificial Intelligence Review*, 55(7): 5169–5213.

Rafiezadeh Shahi, K.; Khodadadzadeh, M.; Tusa, L.; Ghamisi, P.; Tolosana-Delgado, R.; and Gloaguen, R. 2020. Hierarchical sparse subspace clustering (HESSC): An automatic approach for hyperspectral image analysis. *Remote Sensing*, 12(15): 2421.

Szlam, A.; Gregor, K.; and LeCun, Y. 2012. Fast approximations to structured sparse coding and applications to object classification. In *Computer Vision–ECCV 2012: 12th European Conference on Computer Vision, Florence, Italy, October 7-13, 2012, Proceedings, Part V 12*, 200–213. Springer.

Thordsen, E.; and Schubert, E. 2022. ABID: Angle Based Intrinsic Dimensionality—Theory and analysis. *Information Systems*, 108: 101989.

Tovsić, I.; and Frossard, P. 2011. Dictionary learning. *IEEE Signal Processing Magazine*, 28(2): 27–38.

Vidal, R.; Tron, R.; and Hartley, R. 2008. Multiframe motion segmentation with missing data using PowerFactorization and GPCA. *International Journal of Computer Vision*, 79: 85–105.

Wichert, A. 2009. Subspace tree. In *2009 Seventh International Workshop on Content-Based Multimedia Indexing*, 38–43. IEEE.

Wu, T.; Gurram, P.; Rao, R. M.; and Bajwa, W. U. 2015. Hierarchical union-of-subspaces model for human activity summarization. In *Proceedings of the IEEE International Conference on Computer Vision Workshops*, 1–9.

Xiao, H.; Rasul, K.; and Vollgraf, R. 2017. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*.

Yuan, X.; Ren, D.; Wang, Z.; and Guo, C. 2013. Dimension projection matrix/tree: Interactive subspace visual exploration and analysis of high dimensional data. *IEEE Transactions on Visualization and Computer Graphics*, 19(12): 2625–2633.

Zhu, P.; Hui, B.; Zhang, C.; Du, D.; Wen, L.; and Hu, Q. 2019. Multi-view deep subspace clustering networks. *arXiv preprint arXiv:1908.01978*.

Principle Component Trees and their Persistent Homology: Technical Appendix

1 Appendix Outline.

In this appendix, we present material to supplement "Principle Component Trees and Their Persistent Homology". We present 3 sections of supplementary material.

- Practical Details / Considerations for the implementation of our PCT construction algorithm.
- More detailed proofs of our theoretical results listed in the paper.
- **NEW FOR CPAL!** Additional applications.

2 Practical considerations for PCT construction

In the main paper, we stated:

> For simplicity and interpretability, we leave three important implementation details about this and the following test to the technical appendix. 1) In order to reasonably compare angle distributions to those of hyper*spheres*, we first whiten $\mathbf{X}$ before taking the angle distribution. 2) This *explicit* whitening makes the expected angle distribution $\mathbb{P}_D(C)$ dependent on N, so we used monte-carlo simulation to estimate $\mathbb{P}_D(C)$ for $N < 100D$. 3) Whenever comparing the subspace clusteredness of two data matrices, we first project both to a common dimension.

We will now examine these details in order.

Data Pre-Whitening

In section 4, we compare an observed angle distribution with an expected distribution under uniformity on the hypersphere. This CDF for absolute cosine similarities was given in (Thordsen and Schubert 2022) as:

$$\mathbb{P}_D(C) = 2\,\text{Beta}_{\text{CDF}}\left(\tfrac{1+C}{2};\ \alpha = \tfrac{D-1}{2}, \beta = \tfrac{D-1}{2}\right) - 1.$$

Unfortunately, this expected distribution of angles will only properly apply *isotropic distributions*. The expexted distribution of angles will be different for anisotropic distributions. To isolate differences in the observed $\angle\mathbf{X}$ and theoretical $\mathbb{P}_D(C)$ that occur because of subspace clustering behavior and not an isotropic covariance, we whiten $\mathbf{X}$ such that it's diagonal matrix of singular values will be exactly $\mathbf{I}_D$.

Figure 1 demonstrates why this is necessary. All three distributions display roughly the same anisotropic covariance before whitening, but only two could be reasonably called subspace clustered (1b and 1c). Only after whitening the data can we make an "apples-to-apples" comparison of angle distribution.

Monte Carlo Estimation of $\mathbb{P}_D(C)$

Unfortunately, the *explicit* whitening we discuss in the previous section has a side effect. Because whitening makes the covariance **exactly** isotropic, it makes points "over-perpendicular" to what would occur randomly when points are sampled uniformly from a hypersphere. In the extreme case, when we whiten D points in $\mathbb{R}^D$, all the points become exactly perpendicular to each other. See Figure 2 for an example of this in $\mathbb{R}^2$.

Because of this over-whitening, the expected distribution of pairwise angles is no longer dependent only on the data dimension D, but also the number of points N. We denote this new, datasize-dependent expected distribution $\mathbb{P}_{D,N}(C)$. This means we can't use the existing theoretical CDF of angle distribution when dealing with pre-whitened data; it is only the *asymptotic* distribution. Deriving the exact, non-asymptotic angle distribution would be a great direction for future work, but we were unable to find this expression exactly. Instead, we turned to Monte-Carlo simulation.

More specifically, we estimate $\mathbb{P}_{D,N}(C)$ for combinations of D and N from $D = 2$ to $D = 8$, and for $N = D$ to $N = 500D$, taking only 100 values of N in this range. For each combination, we ran enough simulations to collect $100,000,000$ whitened angles, and cached empirical CDF of those angles as $\mathbb{P}_{D,N}(C)$ for later use. Thankfully, the domain of $\mathbb{P}_{D,N}(C)$ is only $[0, 1]$, so we didn't need to estimate small tails of the distribution. Figure 3 shows 4 such Monte-Carlo CDFs and the asymptotic CDF.

Whenever we needed to perform a Cramer-Von-Mises test to compare $\mathbf{X}$ to $\mathbb{P}_{D,N}(C)$, we load the closest cached $\mathbb{P}_{D,N}(C)$ and compare our $\hat{\mathbb{P}}_{D,N}(C)$ to it.

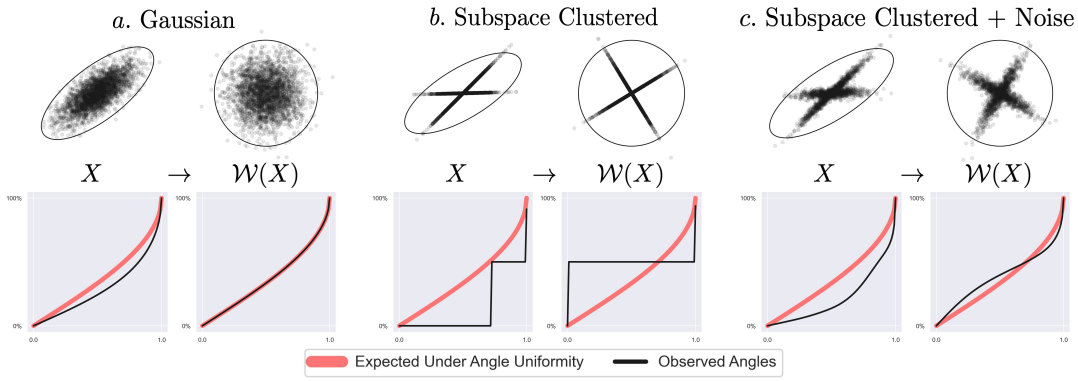


Figure 1: Effect of whitening on observed pairwise angle distribution.

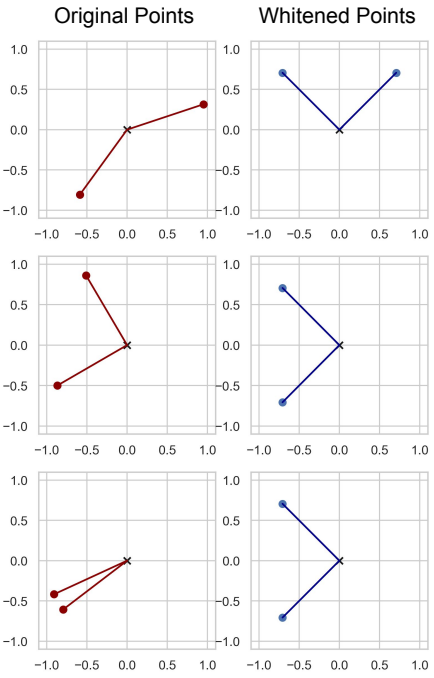


Figure 2: Data whitening makes any 2 points in $\mathbb{R}^2$ exactly perpendicular to each other.

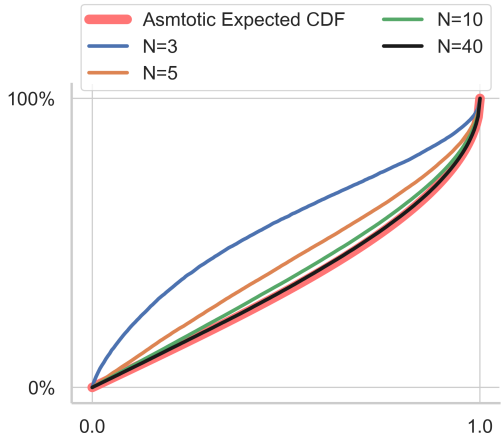


Figure 3: Expected angle CDF for different data sizes. Ambient space is $\mathbb{R}^2$

Projection to a common dimension for comparison. When discussing the test of subspace intersection, we state that our test `INTERSECT-TEST`$(\mathbf{X}, \mathbf{v})$ returns `true` if

$$\texttt{CLUSTER-TEST}(\mathbf{X} - \mathbf{v}\mathbf{v}^T\mathbf{X}) < \texttt{CLUSTER-TEST}(\mathbf{X}).$$

Where $\mathbf{v}$ is the first singular vector of $\mathbf{X}$. This leaves out two details: We need to whiten $\mathbf{X}$ and $\mathbf{X} - \mathbf{v}\mathbf{v}^T\mathbf{X}$, and we need to project $\mathbf{X}$ and $\mathbf{X} - \mathbf{v}\mathbf{v}^T\mathbf{X}$ to a common dimension. We perform this common projection for two reasons. 1) We use monte-carlo simulation to estimate $\mathbb{P}_{D,N}(C)$, and we would need to cache our $\mathbb{P}_{D,N}(C)$ for every possible D, potentially 1000s of them. By projecting to a common, smaller dimension before our test, we only need to cache $\mathbb{P}_{D,N}(C)$ for a handful of D. 2) By definition, $\mathbf{X}$ and $\mathbf{X} - \mathbf{v}\mathbf{v}^T\mathbf{X}$ are of different rank. Therefore, they would use different expected angle distributions, and we can no longer fairly compare the cramer-von-mises test results to see if $\mathbf{X} - \mathbf{v}\mathbf{v}^T\mathbf{X}$ is less clustered than $\mathbf{X}$.

The full algorithm for `INTERSECT-TEST` is in algorithm 1 $W \leq 8$ is an additional hyperparameter of the common dimensionality in which to project the data before comparison. Anecdotally, a larger W leads to wider trees.

3 Proofs of Theoretical Results

Theorem 3.1. *Any union of subspaces, or distribution lying on a union of subspaces can be equivalently represented by a PCT using the same or fewer parameters.*

Proof. We prove this result for a union / mixture of two multivatiate normal distributions which lie in a union of subspaces, but these results apply for any distribution embedded in a union of subspaces, where the data lying on each subspace can be described entirely by a covariance matrix.

Precise theorem: *In an even-prior mixture of two d-dimensional zero-mean Isotropic Gaussian's in a D-dimensional ambient space ($d < D$), where the intersec-*

Algorithm 1: INTERSECT-TEST

Input: $\mathbf{X}_{D\times K}, W \leq 8$

Output: `true` if $\mathbf{v}$, the first singular vector of $\mathbf{X}$ is a plausible intersection of data in $\mathbf{X}$.

1: Let r = RANK($\mathbf{X}$).
2: Let W' = MIN($r-2, W$)
3: Let $\mathbf{U}, s, \mathbf{V}^T$ = SVD(X). ▷ $\mathbf{V}^T$ is implicitly whitened.
4: Let α_1 = CLUSTER-TEST($\mathbf{V}^T[:, :W']$)
5: Let α_2 = CLUSTER-TEST($\mathbf{V}^T[:, 1:W'+1]$) ▷ In null space of $\mathbf{v}$
6: **RETURN** $\alpha_2 < \alpha_1$

tion of the two gaussians is a c-dimensional subspace, we can efficiently and exactly represent the mixture with $2d-c$ vector-valued parameters corresponding to a PCT structure. Formally, let $p(\mathbf{x}) \sim \frac{1}{2}N(0,\mathbf{\Sigma}_A) + \frac{1}{2}N(0,\mathbf{\Sigma}_B)$ where $\Sigma_A = \mathbf{AIA}^T, \Sigma_B = \mathbf{BIB}^T$. This representation requires $2d$ vector-valued parameters: the columns of $\mathbf{A}$ and $\mathbf{B}$. If the intersection of $\mathbf{A}$ and $\mathbf{B}$ is a rank c subspace, then we can decompose $\mathbf{\Sigma}_A$ and $\mathbf{\Sigma}_B$ into $\mathbf{\Sigma}_C + \mathbf{\Sigma}_{A\perp C}$ and $\mathbf{\Sigma}_C + \mathbf{\Sigma}_{B\perp C}$, respectively.

Body of Proof. Let us take a singular value decomposition of $\mathbf{A}^T\mathbf{B} = \mathbf{W\Lambda M}^T$, where by definition of principal angles $\mathbf{\Lambda}$ is a diagonal matrix whose first c diagonals are 1, and the others are less than 1. Let $\mathbf{\Lambda}_S$ be the matrix with only a partial diagonal of c 1s, all other values 0.

Part 1: Decomposition of $\mathbf{\Sigma}_A = \mathbf{AA}^T$ and $\mathbf{\Sigma}_B = \mathbf{BB}^T$.

Let $\mathbf{P} = \mathbf{W\Lambda}_S\mathbf{W}^T$.

$$\begin{aligned}\mathbf{AA}^T &= \mathbf{APA}^T + \mathbf{A}(\mathbf{I}-\mathbf{P})\mathbf{A}^T \\ &= \mathbf{APA}^T + (\mathbf{A}-\mathbf{AP})\mathbf{A}^T \\ &= \mathbf{APA}^T + \mathbf{AA}^T - \mathbf{APA}^T \\ &= \mathbf{AA}^T\end{aligned}$$

(Similar for $\mathbf{B}$)

Let $\mathbf{U\Sigma V}^T$ be the SVD of $(\mathbf{AW\Lambda}_S)(\mathbf{BW\Lambda}_S)^T = \mathbf{AW\Lambda}_S\mathbf{W}^T\mathbf{B}^T$. Note the following: $V\mathbf{\Sigma U}^T = (\mathbf{BW\Lambda}_S)(\mathbf{AW\Lambda}_S)^T = \mathbf{BW\Lambda}_S\mathbf{W}^T\mathbf{A}^T$. $\mathbf{A}^T\mathbf{A} = \mathbf{B}^T\mathbf{B} = \mathbf{I}$, $\mathbf{U}^T\mathbf{U} = \mathbf{V}^T\mathbf{V} = \mathbf{W}^T\mathbf{W} = \mathbf{I}$. $\mathbf{\Lambda}_S = \mathbf{\Lambda}_S^T = \mathbf{\Lambda}_S^2$. $\mathbf{\Sigma}$ also has a partial diagonal of c 1s, $\mathbf{\Sigma} = \mathbf{\Sigma}^T = \mathbf{\Sigma}^2$.

Part 2: $\mathbf{APA}^T = \mathbf{BPB}^T = \mathbf{U\Sigma U}^T$.

$$\begin{aligned}\mathbf{U\Sigma V}^T &= (\mathbf{AW\Lambda}_S)(\mathbf{BW\Lambda}_S)^T \\ \mathbf{U\Sigma V}^T\mathbf{BW\Lambda}_S &= (\mathbf{AW\Lambda}_S)(\mathbf{BW\Lambda}_S)^T\mathbf{BW\Lambda}_S \\ \mathbf{U\Sigma V}^T\mathbf{BW\Lambda}_S &= \mathbf{AW\Lambda}_S \\ \mathbf{U\Sigma V}^T\mathbf{BW\Lambda}_S\mathbf{W}^T\mathbf{A}^T &= \mathbf{AW\Lambda}_S\mathbf{W}^T\mathbf{A}^T \\ \mathbf{U\Sigma V}^T\mathbf{V\Sigma U}^T &= \mathbf{APA}^T \\ \mathbf{U\Sigma U}^T &= \mathbf{APA}^T\end{aligned}$$

(Similarly, for $\mathbf{V\Sigma V}^T = \mathbf{BPB}^T$)

Part 3: $\mathbf{APA}^T = \mathbf{BPB}^T = \mathbf{U\Sigma U}^T = \mathbf{V\Sigma V}^T$.

$$\begin{aligned}\mathbf{APA}^T &= \mathbf{BPB}^T \\ \mathbf{AW\Lambda}_S\mathbf{W}^T\mathbf{A}^T &= \mathbf{BW\Lambda}_S\mathbf{W}^T\mathbf{B}^T \\ \mathbf{U\Sigma V}^T\mathbf{BW\Lambda}_S\mathbf{W}^T\mathbf{A}^T &= \mathbf{V\Sigma U}^T\mathbf{AW\Lambda W}^T\mathbf{B}^T \\ \mathbf{AW\Lambda}_S\mathbf{W}^T\mathbf{B}^T\mathbf{V\Sigma U}^T &= \mathbf{BW\Lambda}_S\mathbf{W}^T\mathbf{A}^T\mathbf{U\Sigma V}^T \\ \mathbf{U\Sigma V}^T\mathbf{V\Sigma U}^T &= \mathbf{V\Sigma U}^T\mathbf{U\Sigma V}^T \\ \mathbf{APA}^T = \mathbf{U\Sigma U}^T &= \mathbf{V\Sigma V}^T = \mathbf{BPB}^T\end{aligned}$$

Going back to the original decomposition of the gaussian covariance matrices $\mathbf{\Sigma}_A$ and $\mathbf{\Sigma}_B$ into $\mathbf{\Sigma}_C + \mathbf{\Sigma}_{A\perp C}$ and $\mathbf{\Sigma}_C + \mathbf{\Sigma}_{B\perp C}$, we have $\mathbf{\Sigma}_C = \mathbf{U\Sigma U}^T$. Because $\mathbf{\Sigma}$ only has c non-zero elements on it's diagonal, $\mathbf{\Sigma}_C$ can be represented by c vector-valued parameters. Furthermore $\mathbf{\Sigma}_{A\perp C} = \mathbf{A}(\mathbf{I}-\P)\mathbf{A}^T$, which needs only $d-c$ parameters. In total, the Gaussian mixture needs only $2d-c$ parameters, which is the number of nodes needed by the equivalent PCT. □

Theorem 3.2. *When presented data following a multivariate normal distribution, PCT construction will reduce to PCA, yielding a degenerate tree with probability at least $(1-\alpha_{test})^D$. Formally, If $\mathbf{X} \in \mathbb{R}^D \sim N(0,\Sigma)$, then the constructed PCT tree will have D nodes and no splits with at least probability $1-(\alpha_{test})^D$. In this case, the vector $\mathbf{v}_i$ of node $\mathcal{N}_i$ at height h_i will be equivalent to the h_ith singular vector of $\mathbf{X}$.*

Proof. If the data is truly multivariate normal, then any linear transformation of the data will also be multivariate normal, such as the linear transformation used to obtain the residual $\mathbf{R}_i = (\mathbf{I}-\mathbf{V}_i\mathbf{V}_i^T)\mathbf{X}$. Therefore, when we test for subspace clustering in such data or any of it's node residuals using CLUSTER-TEST($\mathbf{R_i}$), we will erroneously split a node with probability α_{test}. We repeat this test D times. Furthermore, if we never expand a node into two nodes, each subsequent node vector is a singular vector of $\mathbf{X}$. □

We also prove that our PCT learning algorithm leads to two simple but useful properties of PCTs. Firstly, nodes are orthogonal to their ancestors, making calculation of data approximations fast and decomposable. Secondly, node singular values decrease with height. Together these two properties are used in theorem 3, which provides a simple and efficient way to select the best approximating subtree of an existing tree by selecting its most significant nodes. This is like selecting the best principal components of PCA.

Property 3.1. *The singular vector of any node is orthogonal to those of its ancestors. Formally,*

$$\mathbf{v}_i \perp \mathbf{v}_j \;\; \forall j \in \mathcal{A}_i$$

as a consequence, all ancestral bases are orthonormal.

Proof. This property arises from the tree construction. Consider a node $\mathcal{N}_i$ and it's parent $\mathcal{P}_i$. $\mathbf{v}_i$ is either a singular vector of $\mathbf{R}_{\mathcal{P}_i}$ or a singular vector of a subset of $\mathbf{R}_{\mathcal{P}_i}$. Note that $\mathbf{R}_{\mathcal{P}_i} = \mathbf{X}_{\mathcal{P}_i} - \mathbf{V}_{\mathcal{P}_i}\mathbf{V}_{\mathcal{P}_i}^T\mathbf{X}_{\mathcal{P}_i}$. Therefore, each $\mathbf{v}_i$ is somewhere in the null space of $\mathbf{V}_{\mathcal{P}_i}$, its parent's basis, so $\mathbf{v}_i \perp \mathbf{v}_{\mathcal{P}_i}$. By induction, it is orthogonal to all its ancestors. □

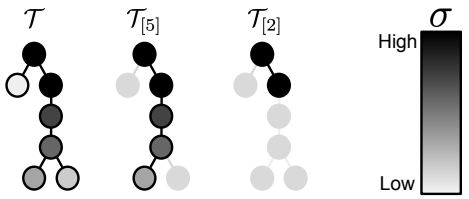


Figure 4: Illustration of optimal subtrees of a PCT.

Property 3.2. *A node's singular value is less than or equal to its ancestors. Formally,*

$$\sigma_i \leq \sigma_j \ \ \forall j \in \mathcal{A}_i.$$

Proof. We show this by contradiction. Consider a node $\mathcal{N}_i$, it's parent $\mathcal{P}_i$, and it's grandparent $\mathcal{P}_i^2$. Let $\overline{\mathbf{R}_{\mathcal{P}_i^2}}$ be the subset of $\mathbf{R}_{\mathcal{P}_i^2}$ by which the parent $\mathcal{N}_{\mathcal{P}_i}$ is constructed. Both $\mathbf{v}_{\mathcal{P}_\mathbf{i}}$ and $\mathbf{v_i}$ are derived from $\overline{\mathbf{R}_{\mathcal{P}_i^2}}$, but $\mathbf{v}_{\mathcal{P}_\mathbf{i}}$ and $\sigma_{\mathcal{P}_i}$ are exactly it's first singular vector and value. If $\sigma_i > \sigma_{\mathcal{P}_i}$, it would violate the Eckart-Young theorem. Therefore, $\sigma_i \leq \sigma_{\mathcal{P}_i}$. By induction, $\sigma_i \leq \sigma_{j \in \mathcal{A}_i}$. □

Property 3.3. *We define the subtree of $\mathcal{T}$ consisting of the n most significant nodes as:*

$$\mathcal{T}_{[n]} = \{\mathcal{N}_i \big| \sigma_i \geq \sigma_{(n)}\}, \{\mathcal{E}_{ij} \big| \sigma_i, \sigma_j \geq \sigma_{(n)}\} \qquad (1)$$

where $\sigma_{(n)}$ is the n_{th} largest singular value of nodes in $\mathcal{T}$. This is the best size n approximating subtree of $\mathcal{T}$. Formally,

$$\underset{\mathcal{T}^* \subset \mathcal{T}}{\arg\min} ||\mathbf{X} - \hat{\mathbf{X}}_{\mathcal{T}^*}||_F := \mathcal{T}_{[n]}$$

Note that we use a slightly non-traditional definition of subtree. In the literature, subtrees of $\mathcal{T}$ are usually defined as one of $\mathcal{T}$'s nodes and all it's descendants. We instead define a subtree of $\mathcal{T}$ as any subset of the nodes and edges of $\mathcal{T}$ which are connected. See Figure 4 for subtree examples.

Proof. We prove this in three parts. 1) We show that property 3.1 means that the approximation error is inversely proportional to the sum of singular values in the tree. 2) We can therefore minimize the approximation error by selecting the nodes with largest singular values. 3) Property 3.2 implies that $\mathcal{T}_{[n]}$ will share the root of $\mathcal{T}$, and will have no *gaps*, that is $\mathcal{N}_i \in \mathcal{T}_{[n]} \implies \mathcal{N}_j \in \mathcal{T}_{[n]} \ \forall \ \mathcal{N}_j \in \mathcal{A}_i$.

In order to prove this, we introduce *branches*, an additional organizing structure of Principal Component Trees. A branch $\mathcal{B}^b, b \in \{1...|\mathcal{B}|\}$ is the set of nodes that includes a single *leaf* node $\mathcal{N}_b$ and all its ancestors $\mathcal{A}_b$. $\mathbf{V}^b$ and $\mathbf{X}^b$ are simply the ancestral basis and assigned data for some leaf node $\mathcal{N}_b$. A single node can belong to one or more branches, and we denote the branches node $\mathcal{N}_i$ is a part of $\mathcal{B}_i$. Notationally, superscript = branch, subscript = node.

It is convenient for us to re-define our node approximations in terms of branches as follows:

$$\hat{\mathbf{X}} := \bigcup_{b=1}^{|\mathcal{B}|} \hat{\mathbf{X}}^b \qquad \hat{\mathbf{X}}^b := \mathbf{V}^b \mathbf{V}^{bT} \mathbf{X}^b$$

Furthermore, because all nodes in a branch are orthogonal, we can further decompose $\hat{\mathbf{X}}^b$ into the rank-1 approximations given by each node in the branch:

$$\hat{\mathbf{X}}^b := \mathbf{V}^b \mathbf{V}^{bT} \mathbf{X}^b := \sum_{i \in \mathcal{B}^b}^{|\mathcal{B}^b|} \mathbf{v}_i \mathbf{v}_i^T \mathbf{X}^b$$

Body of Proof

Part 1: Show that the quality of the tree approximation is given by the sum of the singular values of the nodes. We seek to minimize $||\mathbf{X} - \hat{\mathbf{X}}||_F$. Each element $\hat{\mathbf{x}}_k$ of $\hat{\mathbf{X}}$ is an orthonormal projection of the corresponding point $\mathbf{x}_k$. Via the Cauchy-Schwarz Inequality, we know that $||\hat{\mathbf{x}}_k|| < ||\mathbf{x}_k||$. As such, minimizing $||\mathbf{X} - \hat{\mathbf{X}}||_F$ is equivalent to maximizing $||\hat{\hat{\mathbf{X}}}||_F$. Furthermore, the sum of singular values $\sum_{i=1}^{|\mathcal{T}|} \sigma_i$ is equal to the norm of the approximation $||\hat{\mathbf{X}}||_F$.

$$||\hat{\mathbf{X}}||_F = \sum_{i=1}^{|\mathcal{T}|} \sigma_i \qquad (1a)$$

$$||\bigcup_{b \in \mathcal{B}} \hat{\mathbf{X}}^b|| = \sum_{i=1}^{|\mathcal{T}|} ||\mathbf{v}_i \mathbf{v}_i^T \mathbf{X}_i||_F \qquad (1b)$$

$$||\bigcup_{b \in \mathcal{B}} \sum_{i=1}^{|\mathcal{B}^b|} \mathbf{v}_i \mathbf{v}_i^T \mathbf{X}^b|| = \sum_{i=1}^{|\mathcal{T}|} \sum_{b=1}^{|\mathcal{B}_i|} ||\mathbf{v}_i \mathbf{v}_i^T \mathbf{X}^b||_F \qquad (1c)$$

$$\sum_{b=1}^{|\mathcal{B}|} \sum_{i=1}^{|\mathcal{B}^b|} ||\mathbf{v}_i \mathbf{v}_i^T \mathbf{X}^b||_F = \sum_{i=1}^{|\mathcal{T}|} \sum_{b=1}^{|\mathcal{B}_i|} ||\mathbf{v}_i \mathbf{v}_i^T \mathbf{X}^b||_F \qquad (1d)$$

$$= \sum_{b=1}^{|\mathcal{B}|} \sum_{i=1}^{|\mathcal{T}|} 1_{\{\mathcal{N}_i \in \mathcal{B}^b\}} ||\mathbf{v}_i \mathbf{v}_i^T \mathbf{X}^b||_F \qquad (1e)$$

- 1b) Left: Decompose our full approximation in terms of single branch approximations. Right: Rephrase singular values as norm of approximations given by individual nodes onto their data.
- 1c) Left: Decompose each branch approximation into rank-1 approximations given by each node in the branch. Right: Decompose a node's data assignments as a union of the data assignments for each branch the node is in.
- 1d) Left: Norm of union of branch approximations $\rightarrow$ sum of norm of branch approximations. Recall that each branch's data approximations is disjoint.
- 1e) Reduce both sides to sum over all branches and all nodes, where the node $\mathcal{N}_i$ is in $\mathcal{B}^b$.

Part 2: To select the best size n approximating subtree of $\mathcal{T}$, select the top n nodes from $\mathcal{T}$ in terms of singular values. From part 1, we show that minimizing $||\mathbf{X} - \hat{\mathbf{X}}||_F$ is equivalent to maximizing $\sum_{i=1}^{|\mathcal{T}|} \sigma_i$. To do this maximization, we just select the largest singular value nodes from $\mathcal{T}$ n.

Part 3: These selected nodes form a contiguous subtree of $\mathcal{T}$. Property 3.2 proves that the singular values of the ancestors of a node are larger than that node's singular value. This implies that if a node $\mathcal{N}_i$ is chosen for the optimal approximating subtree, then all of it's ancestors $\mathcal{A}_i$ are also chosen

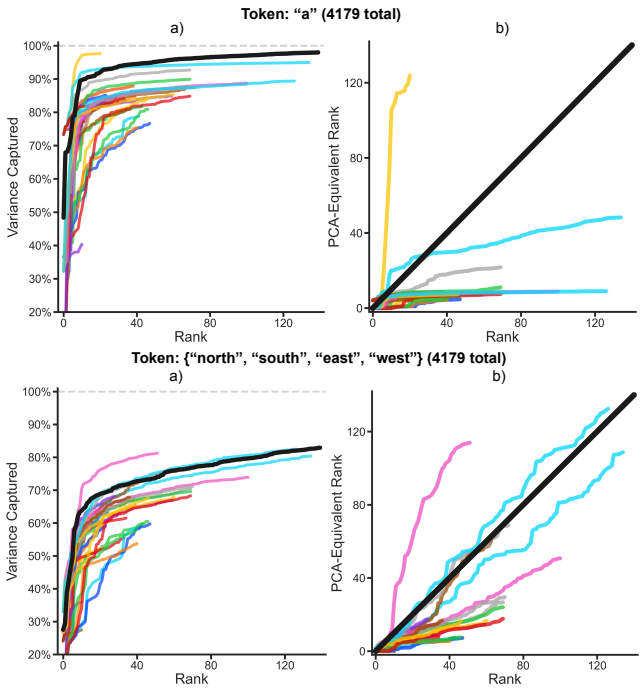


Figure 5: Compression curves for LLM token embeddings. Each curve / line shows the cumulative variance captured by each branch for the token in question. The higher the curve, the more variance captured (measured by (2)) The variance captured by Principal Component Analysis is also shown in black.

for the subtree. It follows that the root of the original tree $\mathcal{T}$ is also the root of the subtree $\mathcal{T}_{[n]}$. Additionally, there are no “gaps” in the subtree; it remains connected, and the edges of the subtree $\mathcal{E}_{[n]}$ is a subset of the edges of original tree. The fact that this optimal subtree / subset of the PCT is connected and rooted at the most important node is analogous to the fact that the optimal subset of principal components in PCA is contiguous and includes the most important components by singular value. □

4 Additional Applications

.

Lossy Data Compression.

Principal Component Analysis / the Singular Value Decomposition has been used as a lossy data compression technique for some time ((Wang et al. 2024), (Swathi et al. 2017), (de Souza, Assis, and Pal 2015)), especially for images. In this section, we show how we can also use Principal Component Trees for lossy data compression, and achieve superior compression rates, with faster decoding.

To encode vector-valued data with a Principal Component Tree, one must encode two things for each point. 1) The branch assignment b for a point $\mathbf{x}$: an integer denoting which branch basis $\mathbf{V}^b$ best represents the point. (*See Property 3.3, proof for more on branches*). 2) The $|\mathcal{B}^b|$ coefficients of that data projected onto it’s best branch: $\mathbf{V}^{bT}\mathbf{x}$. To decode, $\hat{\mathbf{x}}^b := \mathbf{V}^b\mathbf{V}^{bT}\mathbf{x}^b$.

We evaluate the compression performance of PCT vs PCA, we will use a custom dataset of 300k LLM token embeddings. These tokens were extracted from the intermediate MLP layer of LLAMA 3.2 1-Billion (Dubey et al. 2024), on a random sample of the Simple-Wikipedia Dataset (Araikatte 2023). Large Language Models serve as foundational tools in many domains, yet require a vast amount of compute to run, primarily because each token is mapped to vectors in several thousand dimensions. This analysis is not model compression per se, but we hope it inspires further work in model compression, or in using PCTs accelerate LLM inference.

Figure 5 shows compression curves for each branch of a 30-Branch PCT with 1500 nodes for a two classes tokens in the LLM token embeddings dataset: the particle “a” and the cardinal directions “north”, “south”, “east” and “west”. The compression curves show how much variance is captured when using the rank r approximation given by each PCT branch. For a single point $\mathbf{x}$ and branch b, the curve is given by:

$$f_b(r) := \frac{\sum_{i=1}^{r}(\mathbf{V}^{bT}\mathbf{x})_i^2}{||\mathbf{x}||_2^2} \qquad (2)$$

These figures demonstrate two key advantages of using PCT to compress data over PCA. 1) Each branch subspace “specializes” in a subset of the data, capturing more variance in a lower rank than you could using PCA, which finds the optimal approximating basis for all data, regardless of whether a subset of data lies on a very low-dimensional subspace. 2) Some subsets of the data are more low-dimensional than others. For example, “a” tokens seem to lie in a lower-dimensional subspace, than the cardinal directions, and can be represented economically.

Figure 6 Demonstrates the advantages of using Principal Component Trees over PCA to encode data. Figure 6a shows this in absolute terms, but it is even clearer in 6b. Using a 90-branch PCT to encode this data requires on average 101 coefficients per point, but can reproduce the token embeddings as well as rank 472 PCA model. Reducing the number of branches for the PCT reduces this advantage. Of course, the downside is that branched PCT models will use more

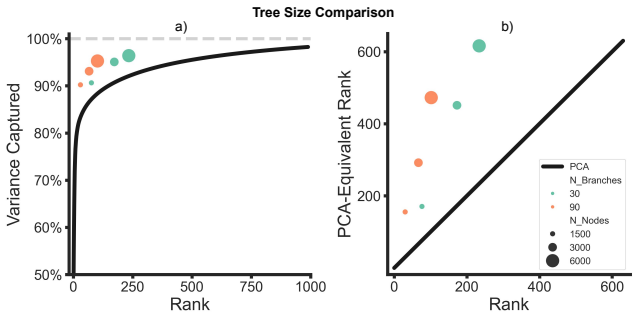


Figure 6: Compression Performance of differently-sized PCTs vs PCA.

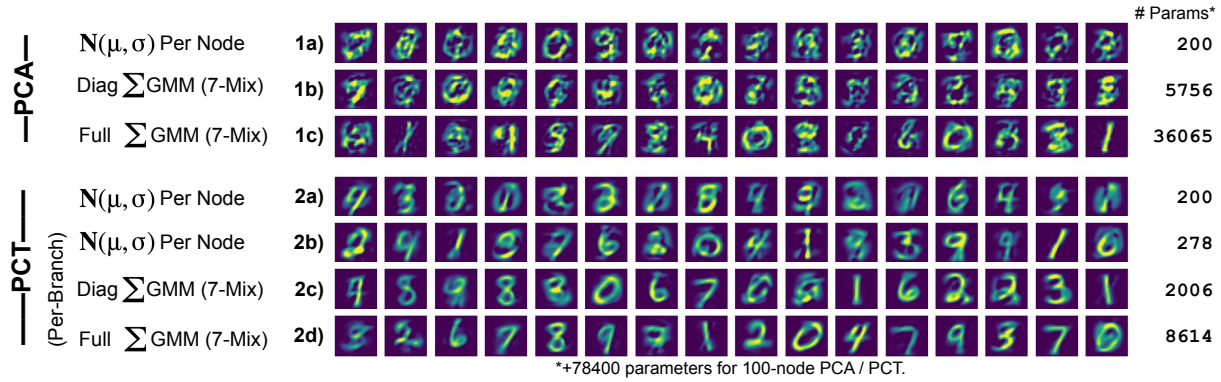


Figure 7: Examples of MNIST digits generated using models built on top of PCA & PCT. Both backbones use 100 nodes, the PCT backbone has 10 branches.

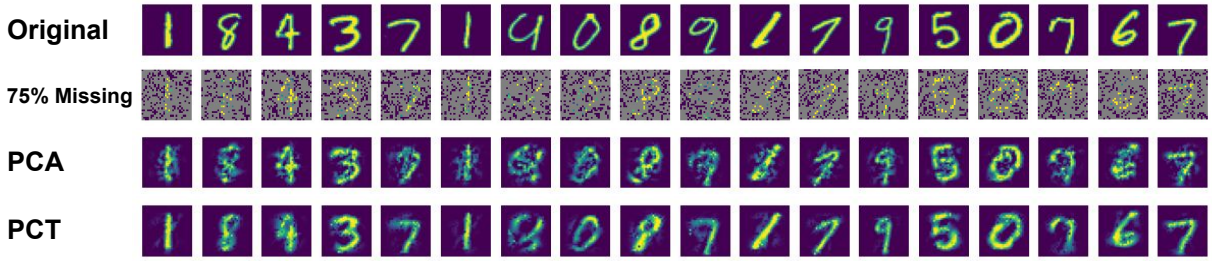


Figure 8: Examples of MNIST (Deng 2012) digits which have been imputed with ISVT (for PCA), or an analogous technique for PCT. ISVT uses a rank-100 decomposition, and PCT imputation uses a PCT with 100 nodes and 10 branches. 75% of values were removed at random.

parameters than the single-branch PCA.

Data Generation.

In this subsection, we will qualitatively demonstrate the ability to use Principal Component Trees to generate new data points. We present two data-generating processes for creating new data points, and show that both perform better, and require fewer additional parameters when using a PCT backbone vs a PCA backbone (recall that PCA is a special case of PCT).

For both processes, we first select a PCT branch b at random, with probability proportional to the number of data points best represented with that branch. For *Node-centric generation* we then sample a value c_i from a separate 1-D probability distribution for each node in the branch (1a, 2a). We then combine these values into a single vector $\mathbf{c}$ (Akin to $\mathbf{V}^{bT}\mathbf{x}$), and obtain the final point via $\mathbf{V}^{b}\mathbf{c}$. This approach is akin to using traditional PCA to decompose a multivariate normal distribution. As a slight modification, we can have a separate 1D distribution per node, per branch passing through that node (2b).

For *Branch-centric generation* we use $\mathbf{V}^{bT}\mathbf{X}^{b}$ to learn a separate multivariate distribution for each branch, and sample $\mathbf{c}$ directly from this distribution. In figure 7, we use gaussian mixture models with 7 components with either diagonal (1b, 2c) or full (1c, 2d) covariance. This produces superior results at the cost of more parameters.

Looking closer at figure 7, we see that generative models with a PCT backbone create more plausible / legible digits than those using the PCA backbone. We offer no theoretical explanation for this, but we posit that the subspace clustering used in PCT construction 'unfolds' the data manifold into regions which better meet the assumptions of multivariate gaussians / GMM models.

Missing Data Imputation.

We can also use Principal Component Trees to perform missing data imputation. In this task, a number of values are missing from each data point, and we wish to estimate the missing values from the existing ones. To this end, we follow a similar approach to Iterative Singular Value Thresholding (Cai, Candès, and Shen 2010), where after initializing our estimate of the missing values with the mean value of the column, we alternate between learning a PCT on the current estimate, and updating our estimate to be the low-rank approximation yielded by the best PCT branch for each point. With a single branch, this reduces to ISVT.

Using subspace clustering for this kind of "High-Rank Matrix Completion" (Eriksson, Balzano, and Nowak 2012) is not entirely new ((Johnson, Li, and Pimentel-Alarcón 2024) (Liu et al. 2021)), but we present qualitative results in figure 8 which show superior results.

References

Aralikatte, R. 2023. Simlple Wikipedia. https://huggingface.co/datasets/rahular/simple-wikipedia. Accessed: 2025-12-10.

Cai, J.-F.; Candès, E. J.; and Shen, Z. 2010. A singular value thresholding algorithm for matrix completion. *SIAM Journal on optimization*, 20(4): 1956–1982.

de Souza, J. C. S.; Assis, T. M. L.; and Pal, B. C. 2015. Data compression in smart distribution systems via singular value decomposition. *IEEE transactions on smart grid*, 8(1): 275–284.

Deng, L. 2012. The mnist database of handwritten digit images for machine learning research [best of the web]. *IEEE signal processing magazine*, 29(6): 141–142.

Dubey, A.; Jauhri, A.; Pandey, A.; Kadian, A.; Al-Dahle, A.; Letman, A.; Mathur, A.; Schelten, A.; Yang, A.; Fan, A.; et al. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Eriksson, B.; Balzano, L.; and Nowak, R. 2012. High-rank matrix completion. In *Artificial Intelligence and Statistics*, 373–381. PMLR.

Johnson, J. S.; Li, H.; and Pimentel-Alarcón, D. 2024. Fusion over the Grassmannian for High-Rank Matrix Completion. In *2024 IEEE International Symposium on Information Theory (ISIT)*, 1550–1555. IEEE.

Liu, J.; Liu, X.; Zhang, Y.; Zhang, P.; Tu, W.; Wang, S.; Zhou, S.; Liang, W.; Wang, S.; and Yang, Y. 2021. Self-representation subspace clustering for incomplete multi-view data. In *Proceedings of the 29th ACM international conference on multimedia*, 2726–2734.

Swathi, H.; Sohini, S.; Gopichand, G.; et al. 2017. Image compression using singular value decomposition. In *IOP Conference Series: Materials Science and Engineering*, volume 263, 042082. IOP Publishing.

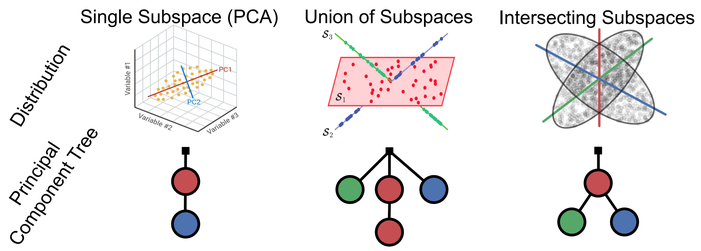
Thordsen, E.; and Schubert, E. 2022. ABID: Angle Based Intrinsic Dimensionality—Theory and analysis. *Information Systems*, 108: 101989.

Wang, X.; Zheng, Y.; Wan, Z.; and Zhang, M. 2024. Svd-llm: Truncation-aware singular value decomposition for large language model compression. *arXiv preprint arXiv:2403.07378*.

Principle Component Trees and their Persistent Homology

Ben Kizaric & Daniel Pimentel-Alarcón
University of Wisconsin - Madison

Summary

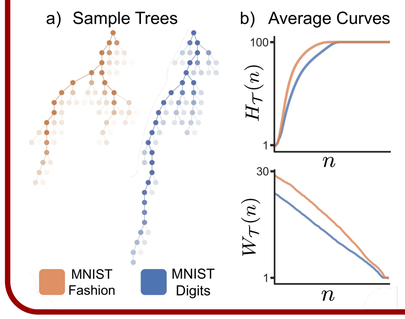


- We **Define** principal component trees as a new approximating structure for high-dimensional data. We posit that PCA and Union of Subspaces models are special cases of PCTs.
- We **Build** principal component trees top-down, using subspace clustering and an angle-based hypothesis test.
- We **Prove** key properties of PCTs and their construction.
- We **Analyze** PCT structure with new Persistent Homology Measures.
- We **Apply** PCTs by using them to investigate Neural Network latent space and multilingual embeddings.

Experimental Results

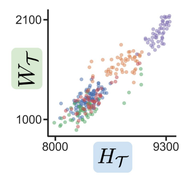
Goal: Compare the subspace structure of different distributions.
Method: Build trees from bootstrap samples of the distributions, perform two sample hypothesis tests on persistent homology measures.

MNIST Fashion vs Digits



Neural Network Latent Space

	Original	AE	VAE	Classifier	UMAP
Original		0.170	0.015	~0.00	~0.00
AE	0.206		0.003	~0.00	~0.00
VAE	~0.00	~0.00		~0.00	~0.00
Classifier	~0.00	~0.00	~0.00		~0.00
UMAP	~0.00	~0.00	~0.00	~0.00	



	Original	AE	VAE	Classifier	UMAP
$H_\mathcal{T}$	8383	8430	8363	8821	9216
$W_\mathcal{T}$	1227	1216	1107	1677	1954

(Not pictured: Multilingual Word Embeddings)

Definition and Properties of Principal Component Trees

A node of a PCT $\mathcal{N}_i$:

Is described by:

- $\mathbf{v}_i$ A basis vector, representing a direction of data variation.
- σ_i A singular value, representing amount of variation.

Has relations with other nodes:

- $\mathcal{A}_i$ & $\mathcal{D}_i$ Ancestors & Descendants
- $\mathcal{P}_i$ & $\mathcal{C}_i$ A Parent & Children

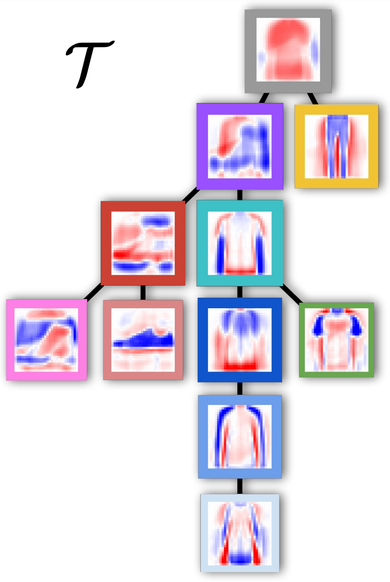
Has a rank-$|\mathcal{A}_i|$ "Ancestral Basis".

$$\mathbf{V}_i := \{\mathbf{v}_j \mid \mathcal{N}_j \in \mathcal{A}_i\}$$

Thus

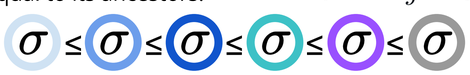
$$\mathcal{N}_a \in \mathcal{A}_b \implies \text{SPAN}(\mathbf{V}_a) \subset \text{SPAN}(\mathbf{V}_b)$$

$\mathcal{T}$



Property 1: The basis vector of any node is orthogonal to those of its ancestors. So, all ancestral bases are orthonormal. $\mathbf{v}_i \perp \mathbf{v}_j \quad \forall j \in \mathcal{A}_i$

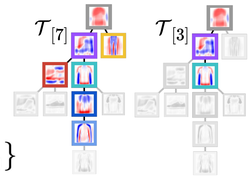
Property 2: A node's singular value is less than or equal to its ancestors. $\sigma_i \leq \sigma_j \quad \forall j \in \mathcal{A}_i$



Property 3: The best size n approximating subtree of $\mathcal{T}$ is given by $\mathcal{T}_{[n]}$ i.e. selecting the n nodes with largest singular values.

$$\arg\min_{\mathcal{T}^* \subset \mathcal{T}} ||\mathbf{X} - \hat{\mathbf{X}}_{\mathcal{T}^*}||_F := \mathcal{T}_{[n]}$$

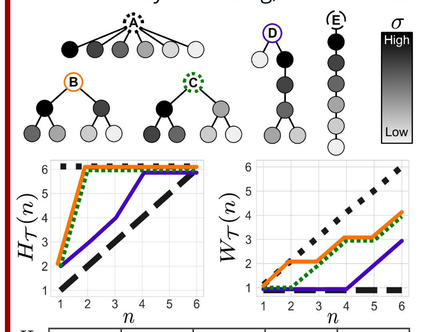
$$\mathcal{T}_{[n]} = \{\mathcal{N}_i | \sigma_i \geq \sigma_{(n)}\}, \{\mathcal{E}_{ij} | \sigma_i, \sigma_j \geq \sigma_{(n)}\}$$



Persistent Homology Measures

The more subspace clustered a distribution is, the more nodes will be split into multiple children, and the tree will be *wider*; it will have more leaves. Conversely, the tree will be *taller* the less subspace clustered a distribution is.

Our two topological measures quantify this intuition by defining curves that describe the height and width of the tree. Property 2 & 3 ensures the curves are monotonically increasing, so we can use the AUC as a reasonable summary.



	A	B	C	D	E
$H_\mathcal{T}$	36	32	32	27	21
$W_\mathcal{T}$	21	15	14	9	6

$H_\mathcal{T}(n)$ **Height Curve**

How many nodes are less than a certain height?

$$H_\mathcal{T}(n) = \sum_{i=1}^{|\mathcal{T}|} \mathbb{1}\{h_i \leq n\}.$$

$W_\mathcal{T}(n)$ **Width Curve**

How wide is the optimal size n subtree?

$$W_\mathcal{T}(n) = \sum_{i=1}^{|\mathcal{T}|} \mathbb{1}\{i \in \mathcal{T}_{[n]} \wedge \mathcal{C}_i \cap \mathcal{T}_{[n]} = \emptyset\}.$$

	(E) Minimum: PCA Tree	(A) Maximum N 1-Dim Clusters
$H_\mathcal{T}(n)$	n	N
$W_\mathcal{T}(n)$	1	n

Approximating Data with Principal Component Trees

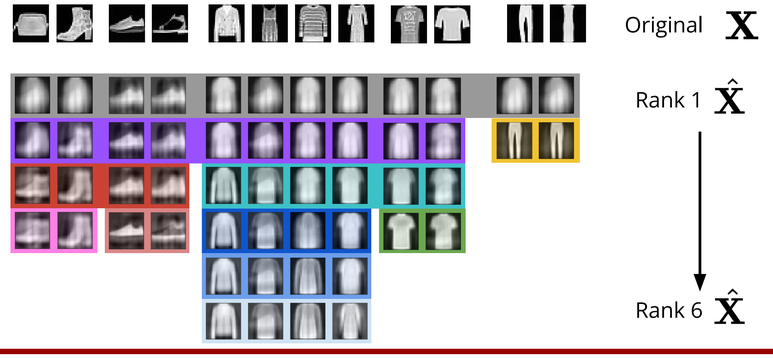
For a PCT node $\mathcal{N}_i$ and data matrix $\mathbf{X}$

$\mathcal{N}_i$ is assigned a subset of the data $\mathbf{X}_i$

$$\mathbf{X}_i := \left\{\mathbf{x} \in \mathbf{X}_{\mathcal{P}_i} \;\middle|\; \arg\max_{j \in \mathcal{S}_i} |\langle \mathbf{v}_j, \mathbf{x} \rangle| = i\right\}$$

$\mathcal{N}_i$'s low-rank approximation $\hat{\mathbf{X}}_i$ & residual $\mathbf{R}_i$

$$\hat{\mathbf{X}}_i = \mathbf{V}_i \mathbf{V}_i^T \mathbf{X}_i \quad \text{and} \quad \mathbf{R}_i := \mathbf{X}_i - \hat{\mathbf{X}}_i$$



More Theoretical Results

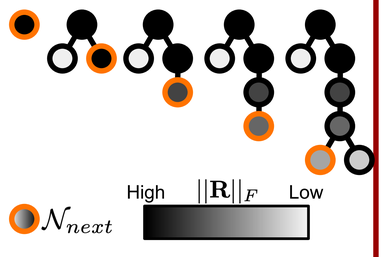
Theorem 6.1 *Any union of subspaces, or distribution lying on a union of subspaces can be equivalently represented by a PCT using the same or fewer parameters.*

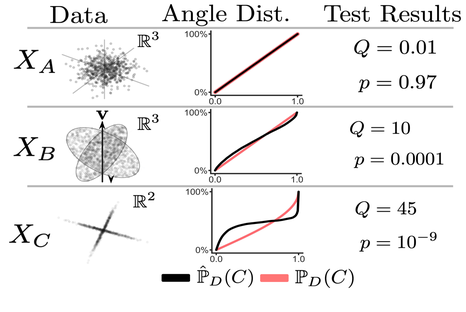
Theorem 6.2 *When presented data following a multivariate normal distribution, PCT construction will reduce to PCA, yielding a degenerate tree with probability at least* $(1 - \alpha_{test})^D$.

Tree Construction Algorithm

Core Algorithm:

while We have not reached our target tree size. **do**
  Pick the leaf node with greatest residuals $\mathcal{N}_{next}$.
  if Subspace clustered structure is detected. **then**
    if $\mathbf{v}_{next}$ is a subspace intersection **then**
      Give $\mathcal{N}_{next}$ one child: The first principle component of $\mathbf{R}_i$.
    else
      Give $\mathcal{N}_{next}$ two children by performing subspace clustering on $\mathbf{R}_i$.
  else
    Give $\mathcal{N}_{next}$ one child: The first principle component of $\mathbf{R}_i$.





To test for subspace clustered structure, we use a non-parametric hypothesis test based on angle distribution. We test our observed angle distribution $\hat{\mathbb{P}}_D(C)$ against the expected distribution:

$$\mathbb{P}_D(C) = 2\,\text{Beta}_{\text{CDF}}\left(\tfrac{1+C}{2};\ \alpha = \tfrac{D-1}{2}, \beta = \tfrac{D-1}{2}\right) - 1.$$

To test if $\mathbf{v}_{next}$ is an intersection of subspaces, we compare the clusteredness of $\mathbf{R}_{next}$ vs the clusteredness of $\mathbf{R}_{next}$ projected onto the null space of $\mathbf{R}_{next}$.