Probabilistic PCA and Extensions

Michael Montemurri [1] **Ahmed Mhedhbi** [2]

Abstract

Principal component analysis (PCA) is a widely used technique for dimensionality reduction, but its lack of flexibility limits its applicability in complex settings. Tipping and Bishop addressed this limitation by developing a probabilistic interpretation, known as probabilistic PCA (PPCA), which enables extensions such as mixtures of PPCA and probabilistic kernel PCA. In this paper, we provide a unified overview of the development and key findings of PPCA and its extensions. We develop an original Bayesian-optimized approach to hyperparameter selection of kernel PCA. Finally, we implement the derived algorithms and apply them to various datasets, highlighting their advantages and confirming theoretical computational results outlined in the literature. The implementations and experiments can be found here

1. Introduction

Principal Component Analysis (PCA) has long served as a cornerstone in data analysis and dimensionality reduction, with applications spanning image processing, bioinformatics, finance, and natural language processing. By projecting high-dimensional data onto a lower-dimensional subspace while maximizing retained variance, PCA efficiently captures the most significant features of the data while filtering out noise. However, for nearly 100 years, classical PCA lacked a formal probabilistic interpretation.

This limitation was addressed in 1997 with the introduction of Probabilistic PCA (PPCA) (Tipping & Bishop, 1999b), by framing PCA within a probabilistic model. PPCA models observed data as a linear transformation of lower-dimensional latent variables, with Gaussian noise accounting for variations not captured by the latent structure. This approach not only quantifies uncertainty in the data but also derives posterior distributions over the latent variables, enabling their estimation given observed data. Moreover, PPCA's probabilistic foundation facilitates parameter estimation through Bayesian techniques, such as the Expectation-Maximization (EM) algorithm.

Although PPCA does not inherently improve PCA's performance in dimensionality reduction, its probabilistic nature enables powerful model extensions. PPCA was extended to mixture of PPCA Models (Tipping & Bishop, 1999a), allowing for data generated from multiple sources or clusters. Moreover, nonlinearity in PPCA was introduced through kernel methods, resulting in probabilistic principal component analysis (PKPCA) (Zhang et al., 2004).

In this report, we explore the theoretical foundations and practical applications of PPCA and its extensions. We begin with a review of classical PCA and PPCA before investigating key extensions, including mixture models, and probabilistic kernel PCA. We introduce original contributions by employing Bayesian optimization to identify optimal hyperparameters for kernel PCA. Experimental results are presented to compare PCA to PPCA in settings with missing data, to compare PCA and MPPCA on heterogeneous data, and our implementation of Bayesian-optimized kernel PCA for image reconstruction on the MNIST dataset. Through this work, we aim to highlight PPCA's versatility in tackling complex data challenges while offering original contributions.

2. PCA and the Probabilistic PCA model

2.1. PCA and Its Limitations

Principal Component Analysis (PCA) is a widely used technique for dimensionality reduction and feature extraction. It identifies a lower-dimensional subspace by projecting data onto orthonormal axes, called principal components, such that the retained variance under projection is maximized. Formally, let $\mathbf{X} \in \mathbb{R}^{N \times d}$ represent a dataset of N observations, where each observation $\mathbf{x}_n \in \mathbb{R}^d$ is a d-dimensional vector. The sample covariance matrix is defined as

$$\mathbf{S} = \frac{1}{N} \sum_{n=1}^{N} (\mathbf{x}_n - \bar{\mathbf{x}})(\mathbf{x}_n - \bar{\mathbf{x}})^\top,$$

[1]Department of Mathematics and Statistics, McGill University, Montréal, Canada [2]Departement d'informatique et de recherche opérationnelle, Université de Montréal, Montréal, Canada. Correspondence to: Michael Montemurri <michael.montemurri@mail.mcgill.ca>, Ahmed Mhedhbi <ahmed.mhedhbi@umontreal.ca>.

IFT 6269 project report - Université de Montréal

where $\bar{\mathbf{x}}$ is the sample mean. The principal components are given by the dominant eigenvectors $\mathbf{w}_j$ (those associated with the largest eigenvalues λ_j) of $\mathbf{S}$, satisfying $\mathbf{S}\mathbf{w}_j = \lambda_j \mathbf{w}_j$. By projecting the data onto the subspace spanned by the first q principal components, PCA provides a q-dimensional representation (typically, $q < d$) in the form $\mathbf{t}_n = \mathbf{W}^\top(\mathbf{x}_n - \bar{\mathbf{x}})$, where the columns of $\mathbf{W}$ correspond to the q eigenvectors associated with the q largest eigenvalues.

While PCA provides an optimal linear representation in terms of variance maximization, it suffers from several limitations:

1. **Linear Assumption**: PCA assumes that the data lies in or near a linear subspace, making it unsuitable for datasets with complex nonlinear structures.
2. **No Probabilistic Interpretation**: Classical PCA lacks a probabilistic framework, preventing it from quantifying uncertainty or modeling latent variable distributions.
3. **Global Representation**: PCA provides a single global linear subspace, which may not adequately capture multimodal or heterogeneous data distributions.

These limitations motivated the development of Probabilistic PCA (PPCA), which introduces a probabilistic interpretation to address some of these challenges. In the following sections, we explore the PPCA model and its extensions, which leverage the strengths of probabilistic models to overcome these limitations.

2.2. The PPCA Model

Probabilistic Principal Component Analysis (PPCA) extends classical PCA by introducing a probabilistic framework, which provides a generative interpretation of the observed data.

In PPCA, the observed d-dimensional data, $\mathbf{x}_n \in \mathbb{R}^d$, is modeled as a linear transformation of q-dimensional latent variables $\mathbf{z}_n \in \mathbb{R}^q$, with additive Gaussian noise. The generative model is given by:

$$\mathbf{x}_n = \mathbf{W}\mathbf{z}_n + \boldsymbol{\mu} + \boldsymbol{\epsilon}_n, \quad \boldsymbol{\epsilon}_n \sim \mathcal{N}(\mathbf{0}, \sigma^2\mathbf{I}),$$

where:

- $\mathbf{W} \in \mathbb{R}^{d\times q}$ is the weight matrix that maps the latent variables to the observed space,
- $\boldsymbol{\mu} \in \mathbb{R}^d$ is the mean vector of the observed data,
- $\boldsymbol{\epsilon}_n \sim \mathcal{N}(\mathbf{0}, \sigma^2\mathbf{I})$ represents isotropic Gaussian noise with variance σ^2.

The latent variables $\mathbf{z}_n$ are assigned a standard normal prior:

$$p(\mathbf{z}_n) = \mathcal{N}(\mathbf{0}, \mathbf{I}),$$

where $\mathbf{I}$ is the identity matrix.

The marginal likelihood of the observed data $\mathbf{x}_n$ can then be derived by integrating over the latent variables:

$$p(\mathbf{x}_n) = \int p(\mathbf{x}_n|\mathbf{z}_n)p(\mathbf{z}_n)\, d\mathbf{z}_n.$$

By substituting the Gaussian form of $p(\mathbf{x}_n|\mathbf{z}_n)$ and $p(\mathbf{z}_n)$, the observed data $\mathbf{x}_n$ follows a multivariate Gaussian distribution:

$$p(\mathbf{x}_n) = \mathcal{N}(\boldsymbol{\mu}, \mathbf{C}), \quad \mathbf{C} = \mathbf{W}\mathbf{W}^\top + \sigma^2\mathbf{I},$$

where $\mathbf{C}$ is the covariance matrix capturing the contributions of both the latent variables and noise.

The posterior distribution of the latent variables $\mathbf{z}_n$ given the observed data $\mathbf{x}_n$ is also Gaussian:

$$p(\mathbf{z}_n|\mathbf{x}_n) = \mathcal{N}(\mathbf{M}^{-1}\mathbf{W}^\top(\mathbf{x}_n - \boldsymbol{\mu}), \sigma^2\mathbf{M}^{-1}),$$

where:

$$\mathbf{M} = \mathbf{W}^\top\mathbf{W} + \sigma^2\mathbf{I}.$$

The posterior provides a probabilistic estimate of the latent variables, incorporating uncertainty into the low-dimensional representation of the data.

Finally, the log-likelihood of the entire dataset is expressed as:

$$\mathcal{L}(\mathbf{W}, \boldsymbol{\mu}, \sigma^2) = -\frac{N}{2}\left[d\ln(2\pi) + \ln|\mathbf{C}| + \mathrm{Tr}(\mathbf{C}^{-1}\mathbf{S})\right],$$

where

$$\mathbf{S} = \frac{1}{N}\sum_{n=1}^{N}(\mathbf{x}_n - \boldsymbol{\mu})(\mathbf{x}_n - \boldsymbol{\mu})^\top,$$

is the sample covariance matrix of the observed data.

2.3. Maximum Likelihood Estimation of PPCA Parameters

The maximum likelihood estimate of the mean vector is simply the empirical mean of the observed data:

$$\boldsymbol{\mu}_{\mathrm{ML}} = \frac{1}{N}\sum_{n=1}^{N}\mathbf{x}_n,$$

For the weight matrix $\mathbf{W}$, the log-likelihood is maximized when its columns are aligned with the eigenvectors of the sample covariance matrix $\mathbf{S}$ that correspond to the largest q eigenvalues. The sample covariance matrix $\mathbf{S}$ is defined as above with $\boldsymbol{\mu} = \boldsymbol{\mu}_{\mathrm{ML}}$

Differentiating the log-likelihood with respect to $\mathbf{W}$ gives us the following:

$$\frac{\partial \mathcal{L}}{\partial \mathbf{W}} = N(\mathbf{C}^{-1}\mathbf{S}\mathbf{C}^{-1}\mathbf{W} - \mathbf{C}^{-1}\mathbf{W}) = 0.$$

Under the modeling assumption that the covariance matrix of the observed data can be written as $\mathbf{C} = \mathbf{W}\mathbf{W}^\top + \sigma^2\mathbf{I}$, it can be shown that the global maximum of the log-likelihood occurs when

$$\mathbf{W}_{\text{ML}} = \mathbf{U}_q(\mathbf{\Lambda}_q - \sigma^2\mathbf{I})^{1/2}\mathbf{R},$$

where:

- $\mathbf{U}_q \in \mathbb{R}^{d\times q}$ is the matrix of eigenvectors of $\mathbf{S}$ corresponding to the q largest eigenvalues,
- $\mathbf{\Lambda}_q \in \mathbb{R}^{q\times q}$ is the diagonal matrix of the q largest eigenvalues,
- $\mathbf{R} \in \mathbb{R}^{q\times q}$ is an arbitrary orthogonal rotation matrix.

The maximum likelihood estimate of the noise variance σ^2 is given by:

$$\sigma^2_{\text{ML}} = \frac{1}{d-q}\sum_{j=q+1}^{d}\lambda_j,$$

where $\lambda_{q+1}, \ldots, \lambda_d$ are the smallest $d-q$ eigenvalues of the sample covariance matrix $\mathbf{S}$. This ensures that the variance not explained by the first q principal components is captured by the noise term. This can be interpreted as the average variance lost per dimension discarded.

2.4. EM Algorithm for PPCA

Although the maximum likelihood estimates for PPCA parameters can be computed directly using the eigendecomposition of the sample covariance matrix, the probabilistic framework of PPCA enables the use of the EM algorithm for parameter estimation of $\mathbf{W}$, $\boldsymbol{\mu}$, and σ^2. This method can offer computational advantages for high-dimensional data (large d) or when the data is incomplete, as it avoids explicitly computing and performing the eigendecomposition of $\mathbf{S}$.

2.4.1. E-Step

In the E-step, we compute the expected value of the latent variables $\mathbf{z}_n$ and their second moments, conditioned on the observed data:

$$\mathbb{E}[\mathbf{z}_n|\mathbf{x}_n] = \mathbf{M}^{-1}\mathbf{W}^\top(\mathbf{x}_n - \boldsymbol{\mu}),$$

$$\mathbb{E}[\mathbf{z}_n\mathbf{z}_n^\top|\mathbf{x}_n] = \sigma^2\mathbf{M}^{-1} + \mathbb{E}[\mathbf{z}_n|\mathbf{x}_n]\mathbb{E}[\mathbf{z}_n|\mathbf{x}_n]^\top,$$

2.4.2. M-Step

In the M-step, we update the parameters $\mathbf{W}$, $\boldsymbol{\mu}$, and σ^2 by maximizing the expected complete-data log-likelihood.

The weight matrix is updated as:

$$\mathbf{W}_{\text{new}} = \mathbf{S}\mathbf{W}(\sigma^2\mathbf{I} + \mathbf{M}^{-1}\mathbf{W}^\top\mathbf{S}\mathbf{W})^{-1},$$

The noise variance is updated as:

$$\sigma^2_{\text{new}} = \frac{1}{d}\left[\text{Tr}(\mathbf{S}) - \text{Tr}(\mathbf{S}\mathbf{W}\mathbf{M}^{-1}\mathbf{W}^\top_{\text{new}})\right].$$

To avoid explicit computation of the covariance matrix, we can compute $\mathbf{S}\mathbf{W}$ as $\sum_n(\mathbf{x_n} - \boldsymbol{\mu})\{(\mathbf{x_n} - \boldsymbol{\mu})^\top\mathbf{W}\}$ rather than $\{\sum_n(\mathbf{x_n} - \boldsymbol{\mu})(\mathbf{x_n} - \boldsymbol{\mu})^\top\}\mathbf{W}$ to go from $O(Nd^2)$ to $O(Ndq)$ complexity.

3. Mixture of PPCA Models

The Mixture of PPCA models (MPPCA) extends the probabilistic framework of PPCA to capture multimodal data distributions. By combining multiple PPCA components, each of which models a local linear subspace, MPPCA is well-suited for datasets with heterogeneous or clustered structures.

3.0.1. Generative Model

MPPCA assumes that the observed data $\mathbf{x}_n \in \mathbb{R}^d$ is generated from a mixture of K local PPCA components. Each component corresponds to a Gaussian distribution parameterized by a local mean $\boldsymbol{\mu}_k$ and covariance matrix $\mathbf{C}_k$.

The generative process for the data can be described as follows:

1. A latent variable $z_n \in \{1, \ldots, K\}$ is drawn from a categorical prior distribution:

   $$p(z_n = k) = \pi_k, \quad \text{where } \sum_{k=1}^{K}\pi_k = 1.$$

   Here, π_k represents the mixing coefficients (the prior probabilities of each component).

2. Conditioned on $z_n = k$, the observed data $\mathbf{x}_n$ is generated from a Gaussian distribution with parameters $(\boldsymbol{\mu}_k, \mathbf{C}_k)$:

   $$p(\mathbf{x}_n|z_n = k) = \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_k, \mathbf{C}_k),$$

   where the covariance matrix $\mathbf{C}_k$ is defined as:

   $$\mathbf{C}_k = \mathbf{W}_k\mathbf{W}_k^\top + \sigma_k^2\mathbf{I}.$$

- $\boldsymbol{\mu}_k \in \mathbb{R}^d$ is the mean of the k-th PPCA component.
- $\mathbf{W}_k \in \mathbb{R}^{d\times q}$ maps the q-dimensional latent subspace to the observed d-dimensional space.
- $\sigma_k^2\mathbf{I}$ accounts for the isotropic Gaussian noise.

The overall marginal distribution of the observed data $\mathbf{x}_n$ is obtained by summing over all K components:

$$p(\mathbf{x}_n) = \sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_k, \mathbf{C}_k).$$

This mixture model captures the multimodal nature of complex datasets, where each Gaussian component corresponds to a distinct region or subspace of the data distribution.

3.0.2. Posterior of the Latent Variables

Given the observed data $\mathbf{x}_n$, the posterior distribution of the latent variable z_n, also known as the *responsibility* is obtained using Bayes' theorem:

$$r_{nk} := p(z_n = k|\mathbf{x}_n) = \frac{\pi_k \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_k, \mathbf{C}_k)}{\sum_{j=1}^{K} \pi_j \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_j, \mathbf{C}_j)}.$$

Here:

- The numerator $\pi_k \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_k, \mathbf{C}_k)$ represents the joint probability of $\mathbf{x}_n$ and $z_n = k$.
- The denominator $\sum_{j=1}^{K} \pi_j \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_j, \mathbf{C}_j)$ is the marginal likelihood of the data.

This reflects the responsibility that the k-th PPCA component has for generating the observation $\mathbf{x}_n$.

3.0.3. Log-Likelihood of the Model

The log-likelihood of the observed data under the MPPCA model is expressed as:

$$\mathcal{L}(\Theta) = \sum_{n=1}^{N} \ln \left(\sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_k, \mathbf{C}_k) \right),$$

where $\Theta = \{\pi_k, \boldsymbol{\mu}_k, \mathbf{W}_k, \sigma_k^2\}_{k=1}^K$ represents the model parameters.

3.0.4. EM Algorithm for MPPCA

Tipping and Bishop develop the following two-staged EM procedure for the estimation of model parameters. As in the development of PPCA, we can use the eigendecomposition of $\mathbf{S}$ to compute the M-step updates or we can use iterative EM procedures. In the following algorithm we outline the eigendecomposition version.

Algorithm 1 Two-Stage EM Algorithm for MPPCA

Input: Observed data $\mathbf{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\} \in \mathbb{R}^d$, number of mixture components K, latent dimension $q < d$.
Initialization: Initialize the parameters for each component $k \in \{1, \ldots, K\}$:
$\boldsymbol{\mu}_k, \mathbf{W}_k \in \mathbb{R}^{d\times q}, \sigma_k^2 > 0, \pi_k$ where, $\sum_{k=1}^K \pi_k = 1$.
repeat
E-step:

1. Compute: $\mathbf{C}_k \;=\; \mathbf{W}_k\,\mathbf{W}_k^\top \;+\; \sigma_k^2\,\mathbf{I}_d$.
2. Compute the responsibilities for each observation n and component k:

$$r_{nk} \;=\; \frac{\pi_k\,\mathcal{N}\big(\mathbf{x}_n \mid \boldsymbol{\mu}_k, \mathbf{C}_k\big)}{\sum_{j=1}^K \pi_j\,\mathcal{N}\big(\mathbf{x}_n \mid \boldsymbol{\mu}_j, \mathbf{C}_j\big)}.$$

M-step (Stage 1): Update mixing coefficients π_k and means $\boldsymbol{\mu}_k$:

$$N_k = \sum_{n=1}^{N} r_{nk}, \quad \pi_{k,\text{new}} = \frac{N_k}{N}, \quad \boldsymbol{\mu}_{k,\text{new}} = \frac{\sum_{n=1}^N r_{nk}\,\mathbf{x}_n}{N_k}$$

M-step (Stage 2): Update $\mathbf{W}_k$ and σ_k^2 for each component.

1. Center data for component k: $\widetilde{\mathbf{x}}_{n,k} \;=\; \mathbf{x}_n \;-\; \boldsymbol{\mu}_{k,\text{new}}$.
2. Compute the responsibility-weighted covariance:

$$\mathbf{S}_k \;=\; \frac{1}{\pi_{k,\text{new}} N_k} \sum_{n=1}^{N} r_{nk}\,\big(\widetilde{\mathbf{x}}_{n,k}\big)\big(\widetilde{\mathbf{x}}_{n,k}\big)^\top.$$

3. (a) Compute the eigendecomposition of $\mathbf{S}_k$. Let $\mathbf{U}_{k,q}$ be the top-q eigenvectors and $\boldsymbol{\Lambda}_{k,q}$ the corresponding top-q eigenvalues.

   (b) Update the loading matrix:

$$\mathbf{W}_{k,\text{new}} \;=\; \mathbf{U}_{k,q}\,\Big(\boldsymbol{\Lambda}_{k,q} - \sigma_k^2\,\mathbf{I}_q\Big)^{1/2}.$$

   (c) Update the noise variance:

$$\sigma^2_{k,\text{new}} \;=\; \frac{1}{d-q} \sum_{j=q+1}^{d} \lambda_{k,j},$$

   where $\lambda_{k,j}$ are the remaining eigenvalues.

until The log-likelihood converges or a max iteration count is reached.
Output: Final estimates $\{\pi_k, \boldsymbol{\mu}_k, \mathbf{W}_k, \sigma_k^2\}_{k=1}^K$. =0

4. Kernel PCA and Probabilistic Extensions

4.1. Kernel PCA

In many applications, nonlinear relationships play a crucial role in describing how data are generated and organized. While standard Principal Component Analysis (PCA) and Probabilistic PCA (PPCA) capture only linear structures, Kernel PCA (KPCA) addresses nonlinearity by implicitly mapping data into a high-dimensional (sometimes infinite-dimensional) feature space $\mathcal{F}$ via a kernel function $k(\mathbf{x}, \mathbf{x}')$. The algorithm for KPCA is outlined as follows:

Algorithm 2 Kernel PCA

Require: Kernel matrix K (computed from data points $X_1, \ldots, X_n$), number of components ℓ

Ensure: Low-dimensional representation points $y_1, \ldots, y_n \in \mathbb{R}^\ell$

1: **Center the kernel matrix:**

$$\tilde{K} = K - \mathbf{1}_n K - K\mathbf{1}_n + \mathbf{1}_n K \mathbf{1}_n$$

where $\mathbf{1}_n$ is an $n \times n$ matrix with all entries equal to $\frac{1}{n}$.

2: **Compute the eigendecomposition of the centered kernel matrix:**

$$\tilde{K} = A \Lambda A^T$$

where A contains the eigenvectors, and Λ is the diagonal matrix of eigenvalues.

3: **Select the top ℓ eigenvectors and eigenvalues:** Let A_k denote the k-th column of A, and λ_k the corresponding eigenvalue, for $k = 1, \ldots, \ell$.

4: **Normalize the selected eigenvectors:** Define the matrix V_ℓ whose columns are:

$$V_\ell = \left[\frac{A_1}{\sqrt{\lambda_1}}, \frac{A_2}{\sqrt{\lambda_2}}, \ldots, \frac{A_\ell}{\sqrt{\lambda_\ell}} \right].$$

5: **for** $i = 1$ to n **do**

6: **Compute the low-dimensional representation for data point X_i:**

$$y_i = \left(v_1^T \tilde{K}_{:,i}, v_2^T \tilde{K}_{:,i}, \ldots, v_\ell^T \tilde{K}_{:,i} \right)^T$$

where v_k is the k-th column of V_ℓ, and $\tilde{K}_{:,i}$ is the i-th column of the centered kernel matrix.

7: **end for**

$y_1, \ldots, y_n \in \mathbb{R}^\ell$ =0

4.1.1. Hyperparameter Selection in KPCA

The performance of Kernel PCA (KPCA) is highly dependent on the selection of its hyperparameters. Key hyperparameters include:

- The **kernel function** $k(\mathbf{x}, \mathbf{x}')$: Common choices include the radial basis function (RBF) kernel, polynomial kernel, and sigmoid kernel, each of which introduces its own set of parameters, such as the bandwidth parameter γ in the RBF kernel or the degree in the polynomial kernel.
- The **number of principal components** ℓ: Determines the dimensionality of the reduced space, balancing the trade-off between dimensionality reduction and preservation of data variance.
- Any **kernel-specific parameters**, such as the regularization constant in certain kernels, which can influence the smoothness and flexibility of the feature mapping.

4.1.2. Challenges in Hyperparameter Selection.

Selecting hyperparameters manually can be challenging, as the optimal configuration often depends on the dataset and task-specific objectives. Grid search or random search techniques are commonly used but can be computationally expensive and suboptimal in high-dimensional hyperparameter spaces. Additionally, evaluating the quality of a given set of parameters often involves metrics such as reconstruction error, variance explained, or task-specific measures like classification accuracy. In Section **??**, we discuss an original contribution for the application of Bayesian optimization techniques to the selection of these hyperparameters, followed by an implementation and comparison of these methods in Section 6.

4.2. Probabilistic Kernel PCA

Probabilistic Kernel PCA (PKPCA) (Zhang et al., 2004), adds a probabilistic generative viewpoint to KPCA, enabling not only nonlinear dimensionality reduction but also uncertainty quantification and Bayesian-style extensions.

4.2.1. Motivation and Representation

Classical KPCA performs PCA in the feature space $\mathcal{F}$ but does not offer a clear generative interpretation there. Probabilistic Kernel PCA (PKPCA) remedies this by proposing that each dimension (or coordinate) in $\mathcal{F}$ is generated by a low-dimensional latent variable plus Gaussian noise.

Concretely, let

$$\mathcal{F}: \quad \mathbf{x}_n \;\mapsto\; \boldsymbol{\phi}(\mathbf{x}_n) \;\in\; \mathbb{R}^r,$$

where r could be large or infinite in principle, but PKPCA's kernel-based formulation ensures we never need to compute $\boldsymbol{\phi}(\mathbf{x})$ directly. If we have n data points, one typically constructs an $(n \times n)$ kernel matrix $\mathbf{K}$, with entries $\mathbf{K}_{ij} = k(\mathbf{x}_i, \mathbf{x}_j)$. PKPCA then treats the feature-vector $\mathbf{g}_n = \boldsymbol{\phi}(\mathbf{x}_n)$ as an implicitly defined vector (often labeled $\in \mathbb{R}^n$ in a sample-based setting), but the final results will not depend explicitly on n or r.

4.2.2. Generative Model in Feature Space

Zhang et al. propose modeling each observed $\mathbf{g}$ (the image of $\mathbf{x}$ under $\boldsymbol{\phi}$) with a latent vector $\mathbf{w} \in \mathbb{R}^m$ (where $m \ll r$ or $m \ll n$). Formally:

$$\mathbf{g} \;=\; \mathbf{B}\,\mathbf{w} \;+\; u\,\mathbf{1} \;+\; \boldsymbol{\epsilon},$$

where:

- $\mathbf{g} \in \mathbb{R}^r$ (or viewed as $\mathbb{R}^n$ in a sample-based approach) is the feature-space representation,
- $\mathbf{w} \in \mathbb{R}^m$ is a latent variable drawn from a Gaussian prior,
- $\mathbf{B} \in \mathbb{R}^{r\times m}$ maps $\mathbf{w}$ into $\mathcal{F}$,
- $u\,\mathbf{1}$ is a global bias (with $\mathbf{1} \in \mathbb{R}^r$),
- $\boldsymbol{\epsilon} \sim \mathcal{N}\big(\mathbf{0},\, (\frac{r}{n})\,\sigma^2\mathbf{I}_r\big)$ or a similar scaling that arises from normalization in Zhang's derivation.

Avoiding Explicit r or n. Initially, one sees factors of r (dimension of $\mathcal{F}$) and n (number of samples). Zhang's main result is that after applying the kernel trick, the final covariance and updates depend solely on $\mathbf{K}$, removing explicit references to r or n. That is, although $\mathbf{g} \in \mathbb{R}^r$ and $\mathbf{B}\mathbf{B}^\top$ is an $(r \times r)$ matrix, we never form them directly; we operate via $\mathbf{K} \in \mathbb{R}^{n\times n}$.

4.2.3. Posterior Inference and Kernel Trick

From the model, $\mathbf{B}\mathbf{B}^\top$ is conceptually an $(r \times r)$ matrix, but Zhang et al. demonstrate:

$$\mathbf{B}\,\mathbf{B}^\top \;\to\; \mathbf{K}_{\text{low-rank}} \;\subseteq\; \mathbf{K} = \boldsymbol{\Phi}^\top\boldsymbol{\Phi}.$$

Hence, $\mathbf{C} = \mathbf{B}\mathbf{B}^\top + \frac{r}{n}\sigma^2\mathbf{I}_r$ transforms into:

$$\mathbf{C} \;=\; \mathbf{K} + \sigma^2\,\mathbf{I}_n$$

in the sample-based coordinate system, with no explicit appearance of r. The dimension r (feature-space dimension) effectively becomes hidden behind $\mathbf{K}$, which is always $(n \times n)$. This is the essence of the kernel trick: the generative approach in $\mathcal{F}$ is mirrored by operations solely on $\mathbf{K}$.

4.2.4. ML Estimates for PKPCA

Similar to PPCA, maximum likelihood solutions in PKPCA can be expressed via the eigendecomposition of a centered kernel matrix. Let $\mathbf{K} \in \mathbb{R}^{n\times n}$ be centered ($\mathbf{HKH}$), and let $\lambda_1 \geq \cdots \geq \lambda_m \geq \cdots \geq \lambda_n \geq 0$ be the eigenvalues of $\mathbf{HKH}$. Then:

$$\sigma^2_{\text{ML}} = \frac{1}{n-m}\sum_{j=m+1}^{n}\lambda_j,$$

$$\mathbf{B}_{\text{ML}} = \mathbf{U}_m\left(\boldsymbol{\Lambda}_m - \sigma^2_{\text{ML}}\mathbf{I}_m\right)^{\frac{1}{2}}\mathbf{R},$$

where:

- $\mathbf{U}_m$ is the matrix of top m eigenvectors of $\frac{1}{n}\mathbf{HKH}$,
- $\boldsymbol{\Lambda}_m$ is the diagonal of the corresponding top m eigenvalues,
- $\mathbf{R}$ is an arbitrary orthogonal rotation.

In kernel form, we never explicitly form $\mathbf{B} \in \mathbb{R}^{r\times m}$; rather, we can express the rank-m subspace in terms of the top eigenvectors of $\mathbf{K}$. Hence, $\mathbf{C}$ emerges as

$$\mathbf{C} \;=\; \mathbf{K} + \sigma^2_{\text{ML}}\,\mathbf{I}_n$$

and no explicit factor of r or n remains in the final usage beyond the dimension of $\mathbf{K}$ itself (which is $(n \times n)$). This aligns with the core principle of PKPCA: the kernel trick eliminates explicit computations in $\mathcal{F}$, making PKPCA feasible for large or even infinite r, while also revealing the ML solutions from the top eigenvalues of the centered kernel matrix.

5. Original Contributions

5.1. Bayesian-Optimized Hyperparameter Tuning of KPCA

We propose a novel approach for selecting optimal hyperparameters for kernel principal component analysis (KPCA) using Bayesian optimization with Gaussian priors. Unlike grid search or random search, Bayesian optimization more efficiently explores the hyperparameter space by leveraging a probabilistic surrogate model to approximate the objective function and guide the search.

Bayesian Optimization Framework. We modeled the reconstruction error as the objective function:

$$f(\theta) = \frac{1}{n}\sum_{i=1}^{n}\|x_i - \hat{x}_i(\theta)\|^2,$$

where $\hat{x}_i(\theta)$ is the reconstruction of x_i under the KPCA model with hyperparameters θ.

Using Gaussian priors for the hyperparameters, we employed a Gaussian Process (GP) as the surrogate model:

$$f(\theta) \sim \mathcal{GP}(m(\theta), k(\theta, \theta')),$$

where $m(\theta)$ is the mean function, and $k(\theta, \theta')$ is the kernel function capturing correlations between hyperparameter configurations.

The acquisition function $a(\theta)$ determines the next set of hyperparameters to evaluate, balancing exploration and exploitation. In this implementation, we used Expected Improvement (EI):

$$a(\theta) = \mathbb{E}\big[\max(0, f_{\text{best}} - f(\theta))\big],$$

where f_{best} is the best observed value of the objective function.

6. Implementation and Experimental Results

6.1. Comparison of PPCA to PCA for Reconstruction Error with Missing Data

To evaluate the reconstruction performance of classical PCA, PPCA (Eigen), and PPCA (EM), we conducted experiments on a synthetic 3D dataset with varying levels of missing data. Missing entries were randomly introduced at ratios of 10%, 25%, 50%, and 75%. The mean squared error (MSE) of reconstruction was computed for the missing entries to assess the effectiveness of each method.

6.1.1. Setup

The experiment involved the following steps:

1. **Data Generation:** A synthetic 3D dataset was generated where z is approximately a linear combination of x and y plus Gaussian noise.

2. **Missing Data Introduction:** Randomly selected entries were set to missing (i.e., `NaN`) at predefined ratios.

3. **Reconstruction Methods:**

   - **PCA:** Missing entries were filled with column means before performing PCA.
   - **PPCA (Eigen):** A probabilistic PCA method leveraging eigen-decomposition for parameter estimation.
   - **PPCA (EM):** A probabilistic PCA method employing the Expectation-Maximization algorithm to iteratively refine parameters.

6.1.2. Results

Table 1 summarizes the reconstruction MSE for each method under different missing ratios. The results demonstrate that PPCA consistently outperforms classical PCA, with PPCA (Eigen) achieving the lowest reconstruction error in most cases.

Table 1. Reconstruction MSE on Missing Entries

Missing Ratio	PCA	PPCA (Eigen)	PPCA (EM)
10%	77.18	73.82	76.85
25%	69.43	64.04	68.59
50%	78.05	73.63	77.26
75%	78.70	75.16	77.21

6.1.3. Discussion

The results highlight the superiority of probabilistic approaches, particularly PPCA (Eigen), in handling missing data.

Figure 1 provides a visual comparison of the reconstructions for a dataset with 50% missing entries. Both PPCA methods produce reconstructions closer to the true data, with PPCA (Eigen) showing slightly better performance.

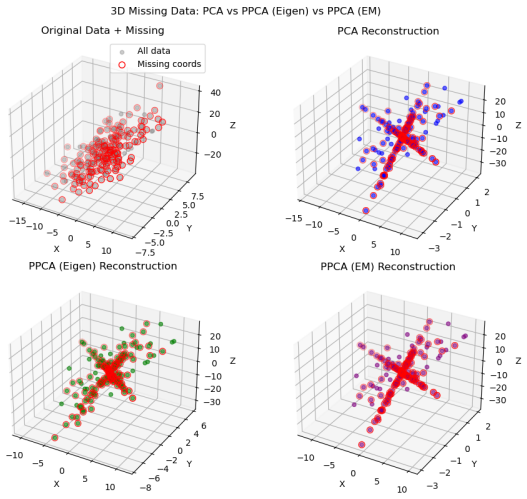

Figure 1. Reconstruction comparison for 3D data with 50% missing entries. Missing points are highlighted in red.

6.2. Implementation of Bayesian-Optimizated KPCA on MNIST

6.2.1. Setup

We apply our implementation of Bayesian-optimized Kernel PCA to the MNIST Dataset for the reconstruction of images and compare the results to the original images and a classical PCA reconstruction. Our models were fit on 100 images.

6.2.2. Results

Figure 2 portrays the comparison of reconstructed images using our approach with the ARD kernel while retaining 100 principal components with the original images and reconstruction with the classic PCA method. The reconstructed images were similar to the classical PCA, indicating the Bayesian-optimization framework for hyperparameter selection for KPCA offers a similar performance while being more flexible to non-linear data.

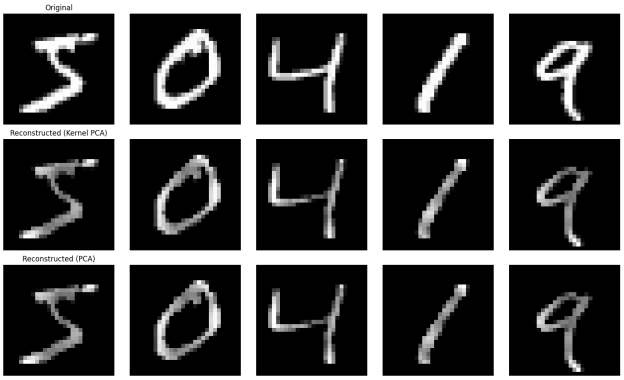

Figure 2. Reconstruction comparison of Bayesian-optimized hyperparameter tuning and PCA for MNIST data.

6.3. Comparing PCA and Mixtures of PPCA

6.3.1. Setup

We compared a global Principal Component Analysis (PCA) and a Mixture of Probabilistic PCA (MPPCA) on a synthetic 3D dataset with five ellipsoidal clusters, designed to reflect complex covariance structures.

- **Dataset**: 700 samples, 3 features, and 5 clusters, each with distinct means and covariances.
- **Metric**: Reconstruction Mean Squared Error (MSE) was used to assess performance.

6.3.2. Results

The reconstruction MSE values for both methods are shown below:

Table 2. Reconstruction MSE for PCA and MPPCA on the synthetic 3D dataset.

Method	Reconstruction MSE
PCA (Single Global Subspace)	0.7713
MPPCA (Local Subspaces)	0.0121

6.3.3. Discussion

The results demonstrate that MPPCA significantly outperforms PCA in terms of reconstruction error. The MPPCA model's ability to utilize local subspaces for each cluster allows it to capture the cluster-specific variations effectively, leading to a much lower reconstruction MSE. In contrast, the single global subspace in PCA fails to model the diverse covariance structures present in the data, resulting in higher reconstruction error. These findings highlight the advantages of using probabilistic mixture models in scenarios with heterogeneous data distributions.

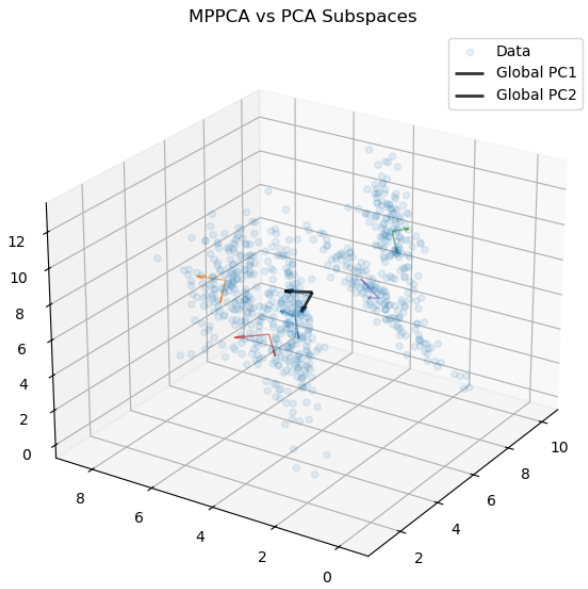

Figure 3. Comparison of global PCA solution versus Local MPPCA Solutions

7. Conclusion

In this project, we provided an overview of probabilistic principal component analysis (PPCA) and its extensions, including mixture models, kernel PCA (KPCA), probabilistic kernel PCA (PKPCA), and a discussion of future directions. We implemented these methods, and tested them on a variety of synthetic and real datasets to showcase their strengths. This work underscores the potential of probabilistic and kernel-based PCA techniques in modern data analysis.

References

Tipping, M. E. and Bishop, C. M. Mixtures of probabilistic principal component analyzers. *Neural computation*, 11 (2):443–482, 1999a.

Tipping, M. E. and Bishop, C. M. Probabilistic principal component analysis. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 61(3):611–622, 1999b.

Zhang, Z., Wang, G., Yeung, D.-Y., and Kwok, J. T. Probabilistic kernel principal component analysis. *Department of Compputer Science, The Hong Kong University of Science and Technology, Tech. Rep*, 2004.