Statistical learning - Report 4 (Knockoff, PCA, PPCA)

Julia Kiczka

June 22, 2025

1) Knockoff Variable Selection at FDR Level $q = 0.4$

The knockoff procedure provides a way to control the false discovery rate (FDR) in variable selection. Given a vector of knockoff statistics $W = (W_1, \ldots, W_p)$, we select variables based on the adaptive threshold

$$T = \min\left\{t > 0 : \frac{1 + \#\{j : W_j \leq -t\}}{\#\{j : W_j \geq t\}} \leq q\right\},$$

where q is the desired FDR level. In this case, we are given:

$$W = (8,\ -4,\ -2,\ 2,\ -1.2,\ -0.6,\ 10,\ 12,\ 1,\ 5,\ 6,\ 7), \quad q = 0.4.$$

In the lecture slides, it was not explained where the thresholds t come from; however, in the original paper *Controlling the False Discovery Rate via Knockoffs* by Rina Foygel Barber and Emmanuel J. Candès, these thresholds are chosen as the absolute values of the statistics W_j. Therefore, we test increasing thresholds $t \in \{0.6, 1, 1.2, 2, 4, 5, 6, 7, 8, 10, 12\}$ to find the smallest one satisfying the knockoff condition. For $t = 4$, we obtain:

$$\#\{j : W_j \leq -4\} = 1, \quad \#\{j : W_j \geq 4\} = 6,$$

so the fraction becomes

$$\frac{1+1}{6} = \frac{2}{6} \approx 0.333 < 0.4.$$

This is the smallest t satisfying the threshold rule, so we set $T = 4$. Thus, the selected variables are those for which $W_j \geq 4$, which correspond to indices:

$$\boxed{0,\ 6,\ 7,\ 9,\ 10,\ 11}.$$

2) PCA – Eigenvalue Decomposition and Projection

Given the centered data matrix:

$$X = \begin{bmatrix} 2 & 0 \\ 0 & 2 \\ -2 & 0 \\ 0 & -2 \end{bmatrix}$$

(a) Sample Covariance Matrix

The sample covariance matrix is computed as:

$$\Sigma = \frac{1}{n-1} X^T X = \frac{1}{3} \begin{bmatrix} 2 & 0 & -2 & 0 \\ 0 & 2 & 0 & -2 \end{bmatrix} \begin{bmatrix} 2 & 0 \\ 0 & 2 \\ -2 & 0 \\ 0 & -2 \end{bmatrix} = \frac{1}{3} \begin{bmatrix} 8 & 0 \\ 0 & 8 \end{bmatrix} = \begin{bmatrix} \frac{8}{3} & 0 \\ 0 & \frac{8}{3} \end{bmatrix}$$

(b) Eigenvalues and Eigenvectors

The covariance matrix is:

$$\Sigma = \begin{bmatrix} \frac{8}{3} & 0 \\ 0 & \frac{8}{3} \end{bmatrix}$$

Eigenvalues: $\lambda_1 = \lambda_2 = \frac{8}{3}$. Eigenvectors (standard basis since matrix is diagonal):

$$v_1 = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \quad v_2 = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

(c) Projection onto the First Principal Component

Assuming we take $v_1 = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$ as the first principal component (arbitrary due to equal eigenvalues), the projection is:

$$Z = Xv_1 = \begin{bmatrix} 2 & 0 \\ 0 & 2 \\ -2 & 0 \\ 0 & -2 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \end{bmatrix} = \begin{bmatrix} 2 \\ 0 \\ -2 \\ 0 \end{bmatrix}$$

(d) Variance Explained by the First Component

Total variance is the sum of eigenvalues:

$$\text{Total variance} = \lambda_1 + \lambda_2 = \frac{16}{3}$$

$$\text{Variance explained by the first component} = \frac{\lambda_1}{\lambda_1 + \lambda_2} = \frac{\frac{8}{3}}{\frac{16}{3}} = 0.5$$

(e) Approximate Reconstruction Using First Principal Component

Reconstruct using:

$$\hat{X} = Zv_1^T = \begin{bmatrix} 2 \\ 0 \\ -2 \\ 0 \end{bmatrix} \begin{bmatrix} 1 & 0 \end{bmatrix} = \begin{bmatrix} 2 & 0 \\ 0 & 0 \\ -2 & 0 \\ 0 & 0 \end{bmatrix}$$

(f) Reconstruction Error

Error is the sum of squared distances between original and reconstructed data:

$$\text{Error} = \sum_{i=1}^{4} \|x_i - \hat{x}_i\|^2 = \|(0,2) - (0,0)\|^2 + \|(0,-2) - (0,0)\|^2 = 2^2 + (-2)^2 = 4 + 4 = 8$$

Problem 3: PPCA – Log-Likelihood and Parameter Estimation

Suppose a single data point $\mathbf{x} \in R^2$ is generated by the PPCA model:

$$\mathbf{x} = Wz + \boldsymbol{\mu} + \boldsymbol{\epsilon}$$

where:

$$W = \begin{bmatrix} 1 \\ 1 \end{bmatrix}, \quad \boldsymbol{\mu} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}, \quad \sigma^2 = 1$$

and the latent variable $z \sim \mathcal{N}(0,1)$ and noise $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \sigma^2 I)$.

(a) Marginal Distribution of x

The marginal distribution of $\mathbf{x}$ in probabilistic PCA is Gaussian:

$$\mathbf{x} \sim \mathcal{N}(\boldsymbol{\mu}, C)$$

where:

$$C = WW^\top + \sigma^2 I$$

Compute:

$$WW^\top = \begin{bmatrix}1\\1\end{bmatrix}\begin{bmatrix}1 & 1\end{bmatrix} = \begin{bmatrix}1 & 1\\1 & 1\end{bmatrix}$$

$$C = WW^\top + \sigma^2 I = \begin{bmatrix}1 & 1\\1 & 1\end{bmatrix} + \begin{bmatrix}1 & 0\\0 & 1\end{bmatrix} = \begin{bmatrix}2 & 1\\1 & 2\end{bmatrix}$$

So the marginal distribution is:

$$\mathbf{x} \sim \mathcal{N}\left(\begin{bmatrix}0\\0\end{bmatrix}, \begin{bmatrix}2 & 1\\1 & 2\end{bmatrix}\right)$$

(b) Log-Likelihood of Observing $\mathbf{x} = \begin{bmatrix}2\\2\end{bmatrix}$

We use the log-density of a multivariate Gaussian:

$$\log p(\mathbf{x}) = -\frac{d}{2}\log(2\pi) - \frac{1}{2}\log|C| - \frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^\top C^{-1}(\mathbf{x}-\boldsymbol{\mu})$$

Here, $d = 2$, $\mathbf{x} = \begin{bmatrix}2\\2\end{bmatrix}$, and $\boldsymbol{\mu} = \begin{bmatrix}0\\0\end{bmatrix}$.

First compute the determinant:

$$|C| = \det\begin{bmatrix}2 & 1\\1 & 2\end{bmatrix} = 4 - 1 = 3$$

Next, the inverse:

$$C^{-1} = \frac{1}{3}\begin{bmatrix}2 & -1\\-1 & 2\end{bmatrix}$$

Compute the quadratic form:

$$(\mathbf{x}-\boldsymbol{\mu})^\top C^{-1}(\mathbf{x}-\boldsymbol{\mu}) = \begin{bmatrix}2 & 2\end{bmatrix}\cdot\frac{1}{3}\begin{bmatrix}2 & -1\\-1 & 2\end{bmatrix}\cdot\begin{bmatrix}2\\2\end{bmatrix}$$

$$= \frac{1}{3}\begin{bmatrix}2 & 2\end{bmatrix}\begin{bmatrix}2\\2\end{bmatrix} = \frac{1}{3}(4+4) = \frac{8}{3}$$

Now compute the log-likelihood:

$$\log p(\mathbf{x}) = -\log(2\pi) - \frac{1}{2}\log 3 - \frac{1}{2}\cdot\frac{8}{3}$$

$$= -\log(2\pi) - \frac{1}{2}\log 3 - \frac{4}{3}$$

(c) Maximum Likelihood Estimation of W and σ^2

Given a dataset $\{\mathbf{x}_n\}_{n=1}^N$, where each $\mathbf{x}_n \in R^D$, the PPCA model assumes the generative process:

$$\mathbf{x}_n = W\mathbf{z}_n + \boldsymbol{\mu} + \boldsymbol{\epsilon}_n$$

where:

- $\mathbf{z}_n \sim \mathcal{N}(0, I_q)$ is a q-dimensional latent variable.

- $\boldsymbol{\epsilon}_n \sim \mathcal{N}(0, \sigma^2 I_D)$ is isotropic Gaussian noise.
- W is a $D \times q$ loading matrix.

The marginal distribution of $\mathbf{x}_n$ is:

$$\mathbf{x}_n \sim \mathcal{N}(\boldsymbol{\mu}, C), \quad \text{where} \quad C = WW^\top + \sigma^2 I$$

The log-likelihood of the full dataset is:

$$\mathcal{L}(W, \sigma^2) = \sum_{n=1}^{N} \log \mathcal{N}(\mathbf{x}_n \mid \boldsymbol{\mu}, WW^\top + \sigma^2 I)$$

We center the data: let $\bar{\mathbf{x}} = \frac{1}{N}\sum_{n=1}^{N} \mathbf{x}_n$ and define the centered data matrix $X_c = [\mathbf{x}_1 - \bar{\mathbf{x}}, \ldots, \mathbf{x}_N - \bar{\mathbf{x}}]$. The sample covariance matrix is:

$$S = \frac{1}{N} X_c X_c^\top$$

According to the MLE solution for PPCA and using the eigendecomposition of S:

$$S = U \Lambda U^\top$$

where:

- $U = [\mathbf{u}_1, \ldots, \mathbf{u}_D]$ is the matrix of eigenvectors.
- $\Lambda = \text{diag}(\lambda_1, \ldots, \lambda_D)$ is the diagonal matrix of eigenvalues, ordered $\lambda_1 \geq \cdots \geq \lambda_D$.

Let U_q be the first q eigenvectors, and $\Lambda_q = \text{diag}(\lambda_1, \ldots, \lambda_q)$. Then the MLE solutions are:

$$W_{\text{ML}} = U_q (\Lambda_q - \sigma^2 I_q)^{1/2} R$$

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

where R is an arbitrary orthogonal rotation matrix in $R^{q \times q}$.

Interpretation:

- The optimal σ^2 is the average variance in the directions orthogonal to the principal subspace.
- The matrix W captures the principal axes scaled by the variance explained by the latent variable.
- This solution is similar to PCA but accounts for isotropic Gaussian noise in the data.

Project 1: Knockoffs

Data Generation: We generate the design matrix $X \in R^{500 \times 450}$ such that its elements are independent and identically distributed (iid) random variables from a normal distribution $\mathcal{N}(0, \frac{1}{\sqrt{n}})$. The response vector is generated according to

$$y = X\beta + \varepsilon,$$

where $\varepsilon \sim 2 \cdot \mathcal{N}(0, I_n)$, the true coefficients satisfy $\beta_i = 10$ for $i \in \{1, \ldots, k\}$ and $\beta_i = 0$ for $i \in \{k+1, \ldots, 450\}$, with $k \in \{5, 20, 50\}$. We investigate the performance of knockoff-based variable selection in high-dimensional linear regression ($n = 500$, $p = 450$), across three levels of sparsity: $k = 5$, 20, and 50 true nonzero coefficients.

Procedure Overview

We use Gaussian knockoffs constructed via the `create.gaussian` function from the `knockoff` package. These knockoffs are generated assuming a multivariate normal distribution with zero mean and a covariance matrix matching the empirical covariance of the design matrix X. Variable importance is evaluated using the `stat.glmnet_coefdiff` statistic, which compares LASSO coefficients for original and knockoff features. Ridge regression is also tested with a similar statistic for comparison. The goal is to estimate the regression coefficients β, predict the response $\mu = X\beta$, and identify the support of β while controlling the False Discovery Rate (FDR) at a target level of $q = 0.2$.

Results

Table 1: β MSE for 100 iterations, $k \in \{5, 20, 50\}$ (mean $\pm$ std)

	$k = 5$	$k = 20$	$k = 50$
OLS	41.2105 ± 7.6181	41.6771 ± 8.2654	39.7854 ± 7.2050
Ridge	0.9212 ± 0.0444	2.6407 ± 0.1559	4.7365 ± 0.2947
LASSO	0.2993 ± 0.1023	0.9116 ± 0.1970	2.0385 ± 0.3112

Table 2: $\mu = X\beta$ MSE for 100 iterations, $k \in \{5, 20, 50\}$ (mean $\pm$ std)

	$k = 5$	$k = 20$	$k = 50$
OLS	3.6127 ± 0.2533	3.6124 ± 0.2229	3.5724 ± 0.2335
Ridge	0.6873 ± 0.0583	1.5062 ± 0.1146	2.0818 ± 0.1451
LASSO	0.2531 ± 0.0833	0.6832 ± 0.1349	1.2756 ± 0.1645

Table 3: False Discovery Proportion (FDP) for 100 iterations, $k \in \{5, 20, 50\}$ (mean $\pm$ std)

	$k = 5$	$k = 20$	$k = 50$
LASSO	0.7785 ± 0.1212	0.7404 ± 0.0577	0.6725 ± 0.0419
Knockoff LASSO	0.1598 ± 0.1881	0.1821 ± 0.1183	0.1930 ± 0.0698
Knockoff Ridge	0.0985 ± 0.1827	0.1628 ± 0.1499	0.1826 ± 0.1000

Table 4: Power for 100 iterations, $k \in \{5, 20, 50\}$ (mean $\pm$ std)

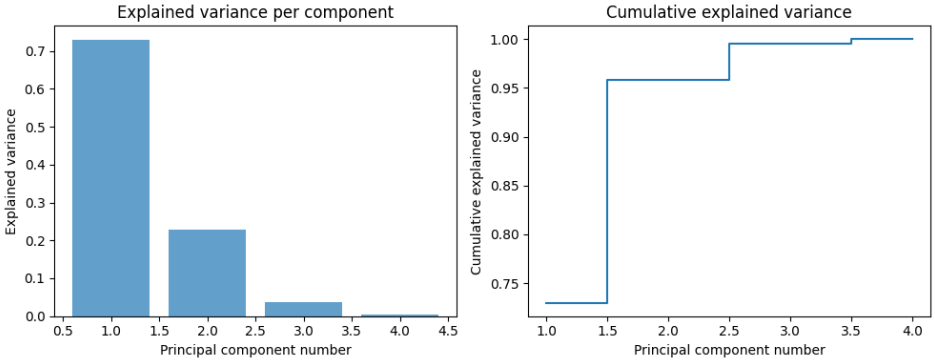
	$k = 5$	$k = 20$	$k = 50$
LASSO	0.9960 ± 0.0281	0.9990 ± 0.0070	0.9988 ± 0.0056
Knockoff LASSO	0.7860 ± 0.3982	0.9665 ± 0.0438	0.9402 ± 0.0455
Knockoff Ridge	0.2500 ± 0.3839	0.6240 ± 0.1894	0.6174 ± 0.1150

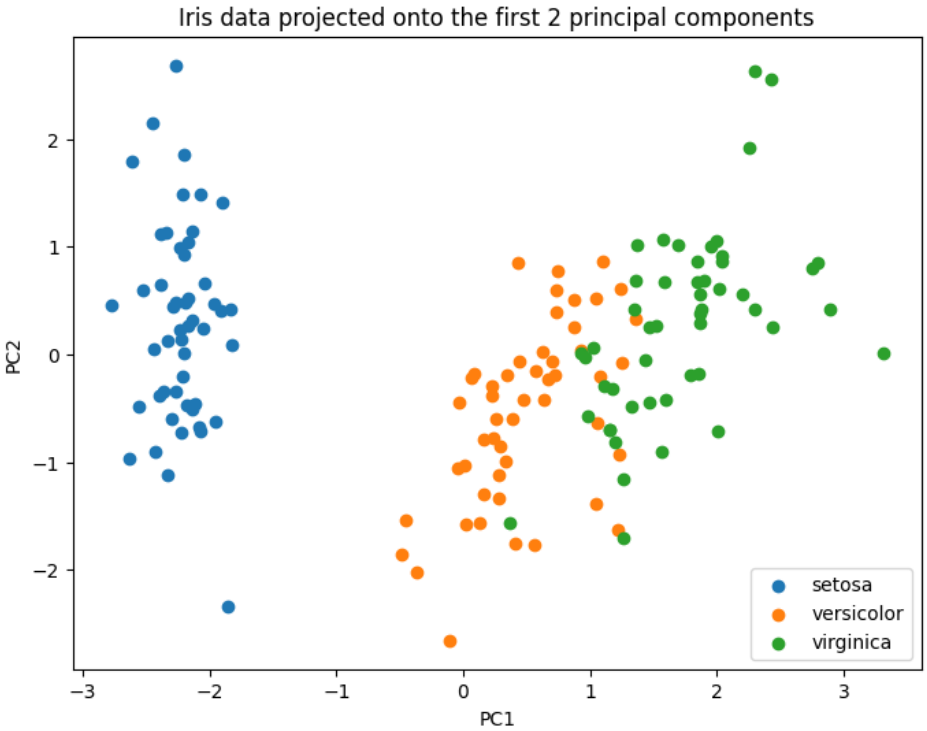
- **Estimation accuracy:** LASSO achieves the lowest MSE for estimating both β and the mean response μ, especially when sparsity is high (low k). Ridge regression shows increasing MSE as k increases but remains better than OLS, which performs poorly throughout.
- **Variable selection:** Without knockoffs, LASSO exhibits extremely high power (nearly 1) but at the cost of very high false discovery proportion (FDP), indicating frequent inclusion of noise variables.
- **Knockoff filtering:** When using knockoffs with LASSO, the FDP is substantially reduced (to 0.16–0.19), while power remains high, especially for $k = 20$ and 50. Knockoff filtering with Ridge statistics also reduces FDP, but with noticeably lower power.
- **Effect of increasing k:** As signal density increases (larger k), the power of knockoff procedures increases and becomes more stable, while FDP remains relatively controlled. This suggests that the knockoff filter adapts well to denser signal settings.

Conclusion: The Gaussian knockoff procedure, paired with LASSO and the coefficient-difference statistic, achieves an effective balance between power and FDR control, particularly as the signal becomes less sparse. It significantly improves variable selection quality over LASSO alone.

Project 2 - Exploratory Data Analysis with PCA

The Iris dataset was standardized prior to applying Principal Component Analysis (PCA) to ensure all variables contributed equally by being on comparable scales. Based on the explained variance plots, the first two principal components capture the majority of the data's variability.





In general, we observe that the first principal component accounts for approximately 70% of the variance in the Iris dataset. When projected into a two-dimensional PCA space, this representation enables clear and effective clustering of the classes, indicating a good separation between them.

Loadings:

$$\text{PC}_1 = 0.521 \cdot \text{sepal length} - 0.269 \cdot \text{sepal width} + 0.580 \cdot \text{petal length} + 0.565 \cdot \text{petal width}$$

$$\text{PC}_2 = 0.377 \cdot \text{sepal length} + 0.923 \cdot \text{sepal width} + 0.024 \cdot \text{petal length} + 0.067 \cdot \text{petal width}$$

The first principal component (PC_1) captures a combination of petal length and petal width with strong positive contributions, reflecting their dominant role in explaining variance. In contrast, the second principal component (PC_2) is primarily influenced by sepal width, indicating it represents variability orthogonal to PC_1, largely independent of petal characteristics.

Project 3: Probabilistic PCA vs Classical PCA – A Simulation Study

Data Generation Procedure

Data were synthetically generated from the PPCA model defined by:

$$\mathbf{x} = W\mathbf{z} + \boldsymbol{\mu} + \boldsymbol{\epsilon},$$

where the latent variable $\mathbf{z} \sim \mathcal{N}(\mathbf{0}, I_k)$, noise $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \sigma^2 I_p)$, $W \in R^{p\times k}$ is the loading matrix, and $\boldsymbol{\mu} \in R^p$ is the mean vector. The parameters were set as follows:

- Number of samples $n = 200$,
- Data dimensionality $p = 20$,
- Latent dimension $k = 3$,
- Noise variance σ^2 systematically varied to analyze noise effects.

The latent variables $\mathbf{z}_i \in R^k$ for each sample $i = 1, \ldots, n$ were independently drawn from the standard multivariate normal distribution:

$$\mathbf{z}_i \sim \mathcal{N}(\mathbf{0}, I_k).$$

The loading matrix $W \in R^{p\times k}$ was generated by sampling each element independently from a univariate standard normal distribution:

$$W_{jk} \sim \mathcal{N}(0, 1), \quad \text{for } j = 1, \ldots, p, \quad k = 1, \ldots, k.$$

Similarly, the mean vector $\boldsymbol{\mu} \in R^p$ was generated by sampling each component independently from a univariate standard normal distribution:

$$\mu_j \sim \mathcal{N}(0, 1), \quad \text{for } j = 1, \ldots, p.$$

Data Reconstruction and Reconstruction Error

Both PCA and PPCA reduce data dimensionality by projecting the original high-dimensional data onto a lower-dimensional latent space. Given the original data matrix $\mathbf{X} \in R^{n\times p}$, where n is the number of samples and p the number of features, the goal is to approximate $\mathbf{X}$ using fewer latent dimensions $k < p$.

Reconstruction with PCA: PCA identifies an orthogonal projection matrix $\mathbf{U} \in R^{p\times k}$ corresponding to the top k eigenvectors of the sample covariance matrix. The low-dimensional representation is

$$\mathbf{Y} = (\mathbf{X} - \bar{\mathbf{X}})\mathbf{U},$$

where $\bar{\mathbf{X}}$ is the mean vector of $\mathbf{X}$. The reconstructed data in the original space is then

$$\hat{\mathbf{X}} = \mathbf{Y}\mathbf{U}^\top + \bar{\mathbf{X}}.$$

Reconstruction with PPCA: PPCA models the data probabilistically using a latent variable model with Gaussian noise. After estimating the loading matrix $\mathbf{W} \in R^{p\times k}$, noise variance σ^2, and mean $\boldsymbol{\mu}$, the latent representation $\mathbf{Z}$ is inferred, and the reconstructed data is given by

$$\hat{\mathbf{X}} = \mathbf{Z}\mathbf{W}^\top + \boldsymbol{\mu}.$$

Reconstruction Error: To quantify how well the reduced representations capture the original data, the reconstruction error is computed as the mean squared error (MSE) between the original data $\mathbf{X}$ and its reconstruction $\hat{\mathbf{X}}$:

$$\text{Reconstruction Error} = \frac{1}{n}\sum_{i=1}^{n}\|\mathbf{x}_i - \hat{\mathbf{x}}_i\|_2^2 = \frac{1}{n}\sum_{i=1}^{n}\sum_{j=1}^{p}(x_{ij} - \hat{x}_{ij})^2.$$

Lower reconstruction error indicates better fidelity in capturing the original data structure by the dimensionality reduction method.

Covariance Matrix Error Metric

To assess how well PCA and PPCA preserve the second-order structure of the data, we computed the discrepancy between the covariance matrix of the original data and that of the reconstructed data. Let $\mathbf{X} \in R^{n\times p}$ denote the original data matrix and $\hat{\mathbf{X}} \in R^{n\times p}$ the reconstructed data obtained via either PCA or PPCA. Denote the sample covariance matrices as:

$$\mathbf{S} = \frac{1}{n-1}(\mathbf{X} - \bar{\mathbf{X}})^\top(\mathbf{X} - \bar{\mathbf{X}}), \quad \hat{\mathbf{S}} = \frac{1}{n-1}(\hat{\mathbf{X}} - \bar{\hat{\mathbf{X}}})^\top(\hat{\mathbf{X}} - \bar{\hat{\mathbf{X}}}),$$

where $\bar{\mathbf{X}}$ and $\bar{\hat{\mathbf{X}}}$ are row-wise means of $\mathbf{X}$ and $\hat{\mathbf{X}}$, respectively. The covariance matrix error was then computed as the mean squared element-wise difference between these two covariance matrices:

$$\text{Covariance Error} = \frac{1}{p^2}\sum_{i=1}^{p}\sum_{j=1}^{p}\left(S_{ij} - \hat{S}_{ij}\right)^2 = \frac{1}{p^2}\|\mathbf{S} - \hat{\mathbf{S}}\|_F^2,$$

PPCA Covariance Formula:

For PPCA, we also compute covariance error based on the model-implied covariance formula:

$$\hat{\mathbf{C}} = \hat{\mathbf{W}}\hat{\mathbf{W}}^\top + \hat{\sigma}^2\mathbf{I}$$

We estimate $\hat{\sigma}^2$ as the average of the residual eigenvalues:

$$\hat{\sigma}^2 = \frac{1}{p-k}\sum_{j=k+1}^{p}\lambda_j$$

and the loading matrix $\hat{\mathbf{W}}$ as:

$$\hat{\mathbf{W}} = \mathbf{U}_k\,\text{diag}\left(\sqrt{\lambda_1 - \hat{\sigma}^2}, \ldots, \sqrt{\lambda_k - \hat{\sigma}^2}\right)$$

where λ_j are the eigenvalues and $\mathbf{U}_k$ the top k eigenvectors of the sample covariance matrix.

Minka's BIC for Probabilistic PCA (Eigenvalue-Based)

Given the sample covariance eigenvalues $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_p$, the noise variance is estimated as:

$$\sigma^2 = \frac{1}{p-k}\sum_{j=k+1}^{p}\lambda_j$$

The approximate log-likelihood (up to constants) is:

$$\log p(\mathbf{X} \mid k) = -\frac{n}{2}\left(\sum_{i=1}^{k}\log\lambda_i + (p-k)\log\sigma^2 + p\right)$$

The number of free parameters in the PPCA model is:

$$\gamma = pk - \frac{k(k-1)}{2} + 1 + p$$

Hence, the BIC is:

$$\boxed{\mathrm{BIC}(k) = -2 \log p(\mathbf{X} \mid k) + \left(pk - \frac{k(k-1)}{2} + 1 + p \right) \log n}$$

where:

- n is the number of observations,
- p is the number of dimensions,
- k is the number of latent dimensions,
- λ_i are the top k eigenvalues of the sample covariance matrix,
- σ^2 is the estimated noise variance.

Results Summary on Synthetic Data

Noise	Reconstruction Error PCA	Reconstruction Error PPCA	Covariance Error PCA	Covaraince Error PPCA	Covariance Error PPCA Formula	Dimension (BIC)
0.001	0.0166	0.0166	4.54e-08	4.54e-08	4.55e-09	3
0.005	0.0845	0.0845	1.16e-06	1.16e-06	9.77e-08	3
0.010	0.1671	0.1671	4.45e-06	4.45e-06	3.06e-07	3
0.050	0.8484	0.8484	1.16e-04	1.16e-04	9.35e-06	3
0.100	1.6944	1.6944	4.74e-04	4.74e-04	4.80e-05	3
0.500	8.3109	8.3109	1.12e-02	1.12e-02	9.20e-04	3
1.000	16.1768	16.1768	4.23e-02	4.23e-02	3.45e-03	3
2.000	33.9530	33.9530	1.84e-01	1.84e-01	1.28e-02	3

Table 5: Reconstruction and covariance errors for PCA and PPCA under varying noise levels and best dimension chosen by Minka's BIC for PPCA.

PCA_Iris	PPCA_Iris
0.1014	0.1014

Table 6: Reconstruction error on the centered Iris dataset (2D projection).

Interpretation and Conclusion

Classical PCA performs dimensionality reduction by identifying orthogonal directions that maximize the variance of the projected data. It can be understood as solving an eigenvalue problem that seeks a low-rank approximation minimizing the squared reconstruction error. PPCA generalizes PCA through a probabilistic latent variable model, explicitly incorporating Gaussian noise with variance σ^2. This probabilistic framework enables parameter estimation via maximum likelihood and facilitates principled noise modeling and uncertainty quantification. Our simulation results show that both methods generally achieve similar reconstruction accuracy. Reconstruction errors increase systematically with noise. Theory states that in the limit $\sigma^2 \to 0$, PPCA converges to classical PCA. Interestingly, empirical covariance matrix reconstruction errors are nearly identical for both PCA and PPCA, indicating that despite differences in reconstruction fidelity, both models approximate the underlying data covariance similarly. Formula-based covariance error for PPCA is smaller than empirical one, showing accuracy of model-based estimation. All covariance errors rise when more noise is added. Minka's BIC criterion consistently identifies the optimal latent dimension as 3. Given that the true latent dimension $k = 3$ was used to generate the data, this shows that BIC can effectively detect the correct dimension under various noise levels. When applied to the real-world Iris dataset, PCA and PPCA produce very similar reconstruction errors. Projected into a two-dimensional latent space, both methods enable clear and effective clustering of classes, indicating good class separation.

Figure 1: PCA - Iris data projected onto the first 2 principal components.

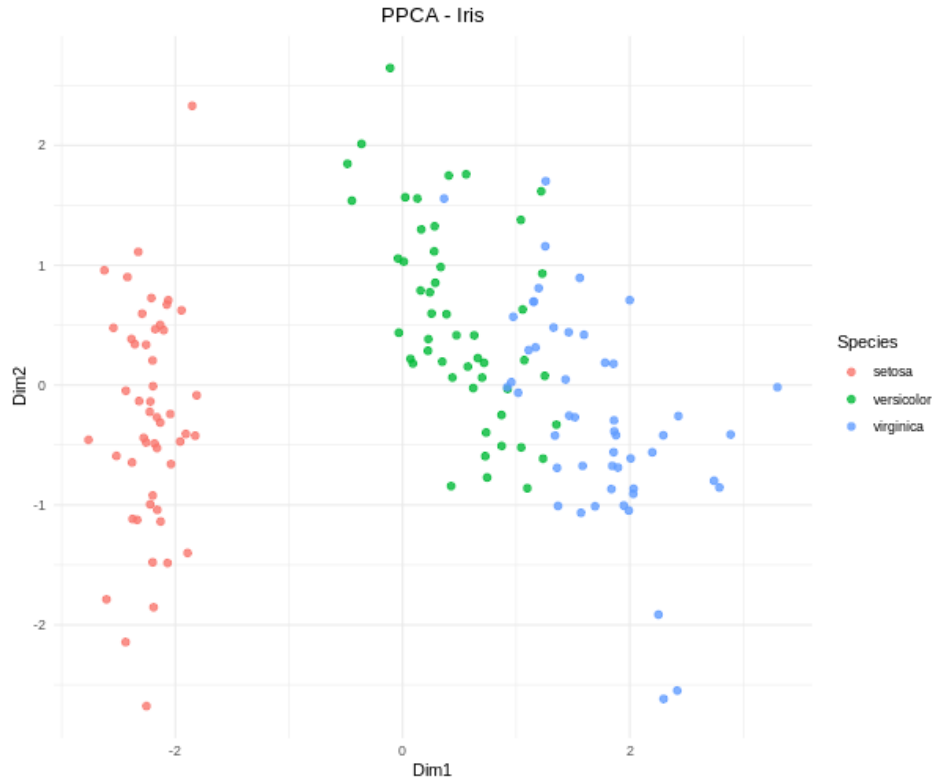


Figure 2: PPCA - Iris data projected onto the first 2 principal components.

Code for Project 3

```
r('''
library(Rdimtools)
library(ggplot2)

set.seed(123)

n <- 200
p <- 20
k_true <- 3
noise_levels <- c(0.001, 0.005, 0.01, 0.05, 0.1, 0.5, 1.0, 2.0)
max_dim <- 19

results <- data.frame()

for (sigma2 in noise_levels) {
  W <- matrix(rnorm(p * k_true), p, k_true)
  mu <- rnorm(p)
  Z <- matrix(rnorm(n * k_true), n, k_true)
  epsilon <- matrix(rnorm(n * p, sd = sqrt(sigma2)), n, p)
  X <- Z %*% t(W) + matrix(mu, n, p, byrow = TRUE) + epsilon
  X_centered <- X - matrix(colMeans(X), nrow(X), ncol(X), byrow = TRUE)
  res_pca <- do.pca(X, ndim = k_true) #centers by default
  res_ppca <- do.ppca(X_centered, ndim = k_true)
  Xhat_pca <- res_pca$Y %*% t(res_pca$projection)
  Xhat_ppca <- res_ppca$Y %*% t(res_ppca$projection)
  rec_error_pca <- mean(rowSums((X_centered - Xhat_pca)^2))
  rec_error_ppca <- mean(rowSums((X_centered - Xhat_ppca)^2))
  S_true <- cov(X_centered)
  S_pca <- cov(Xhat_pca)
  S_ppca <- cov(Xhat_ppca)
  cov_diff_pca <- mean((S_true - S_pca)^2)
  cov_diff_ppca <- mean((S_true - S_ppca)^2)
  eig <- eigen(S_true, symmetric = TRUE)
  lambda <- eig$values
  U <- eig$vectors
  sigma2_est_ppca <- mean(lambda[(k_true + 1):p])
  W_est_ppca <- U[, 1:k_true] %*% diag(sqrt(pmax(lambda[1:k_true] - sigma2_est_ppca, 0)), k_true, k_t
  C_ppca_formula <- W_est_ppca %*% t(W_est_ppca) + sigma2_est_ppca * diag(p)
  cov_diff_ppca_formula <- mean((S_true - C_ppca_formula)^2)
  dims <- 1:max_dim
  bics <- rep(Inf, length(dims))
  for (i in seq_along(dims)) {
    ki <- dims[i]
    sigma2_est <- mean(lambda[(ki + 1):p])
    loglik <- - (n / 2) * (sum(log(lambda[1:ki])) + (p - ki) * log(sigma2_est) + p)
    penalty <- p * ki - ki * (ki - 1) / 2 + 1 + p
    bics[i] <- -2 * loglik + penalty * log(n)
  }
  best_dim <- dims[which.min(bics)]
  results <- rbind(results, data.frame(
    Noise = sigma2,
    RecError_PCA = rec_error_pca,
    RecError_PPCA = rec_error_ppca,
    CovDiff_PCA = cov_diff_pca,
```

```
    CovDiff_PPCA = cov_diff_ppca,
    CovDiff_PPCA_Form = cov_diff_ppca_formula,
    BestDim = best_dim
  ))
}

data(iris)
X_real <- scale(iris[, 1:4], center = TRUE, scale = FALSE)
res_pca_real <- do.pca(X_real, ndim = 2)
res_ppca_real <- do.ppca(X_real, ndim = 2)
Xhat_pca_real <- res_pca_real$Y %*% t(res_pca_real$projection)
Xhat_ppca_real <- res_ppca_real$Y %*% t(res_ppca_real$projection)
rec_error_pca_real <- mean(rowSums((X_real - Xhat_pca_real)^2))
rec_error_ppca_real <- mean(rowSums((X_real - Xhat_ppca_real)^2))
print(results)
print(c(PCA_Iris = rec_error_pca_real, PPCA_Iris = rec_error_ppca_real))


''')
```