EECE-6822 Machine Learning

Dimensionality reduction, PCA

Cris Ababei
Dept. of Electrical and Computer Engineering
MARQUETTE UNIVERSITY
BE THE DIFFERENCE.

1

1

PART 1

Principal Component Analysis (PCA)

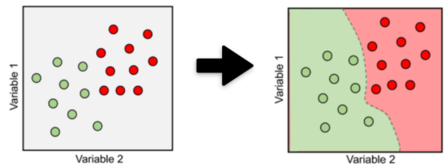
2

2

Unsupervised vs. Supervised learning

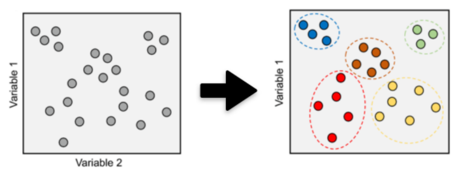
Previously: supervised learning

- Each data point x_i has a corresponding label y_i;$\{x_i, y_i\}_{i=1}^{n}$ with $x_i \in \mathbb{R}^d$, $y_i \in \mathbb{R}$. Try to predict the label y for a new test point x

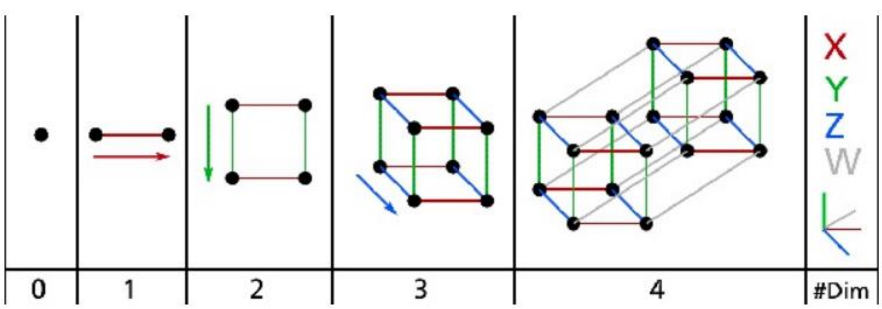


Now: Unsupervised learning

- No labels: data $\{x_i\}_{i=1}^{n}$ with $x_i \in \mathbb{R}^d$. Try to model the data distribution $P(X)$, potentially by finding patterns/clusters, or a low-dimensional representation



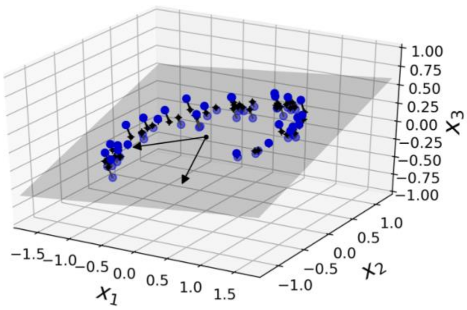
The Curse of Dimensionality

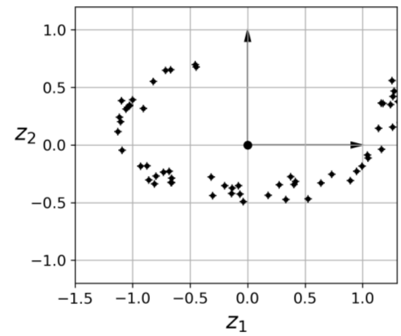


Here is a more troublesome difference: if you pick two points randomly in a unit square, the distance between these two points will be, on average, roughly 0.52. If you pick two random points in a unit 3D cube, the average distance will be roughly 0.66. But what about two points picked randomly in a 1,000,000-dimensional hypercube? The average distance, believe it or not, will be about 408.25 (roughly $\sqrt{1,000,000/6}$)!

[***B3-Geron**] Aurelien Geron, Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow, O'Reilly, 2022.

Approaches for Dimensionality Reduction: **Projection**



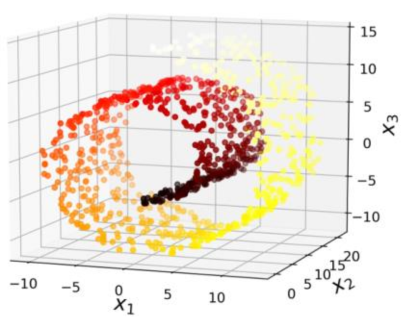


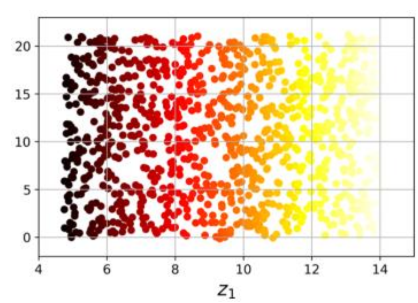
Notice that all training instances lie close to a plane: this is a lower-dimensional (2D) subspace of the high-dimensional (3D) space. If we project every training instance perpendicularly onto this subspace (as represented by the short lines connecting the instances to the plane), we get the new 2D dataset shown in Figure 8-3. Ta-da! We have just reduced the dataset's dimensionality from 3D to 2D. Note that the axes correspond to new features z_1 and z_2 (the coordinates of the projections on the plane).

5

5

Approaches for Dimensionality Reduction: **Manifold Learning**





The Swiss roll is an example of a 2D manifold. Put simply, a 2D manifold is a 2D shape that can be bent and twisted in a higher-dimensional space. More generally, a q-dimensional manifold is a part of an d-dimensional space (where q < d) that locally resembles a q-dimensional hyperplane. In the case of the Swiss roll, q = 2 and d = 3: it locally resembles a 2D plane, but it is rolled in the third dimension.

6

6

Principal Component Analysis (PCA)

- PCA is by far the most popular dimensionality reduction algorithm.
  - First, it identifies the hyperplane that lies closest to the data
  - Then, it projects the data onto it.
- Choosing the hyperplane: **preserve the variance**

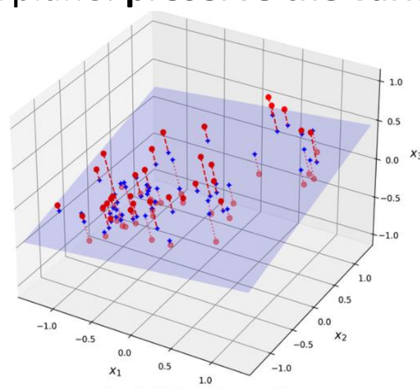
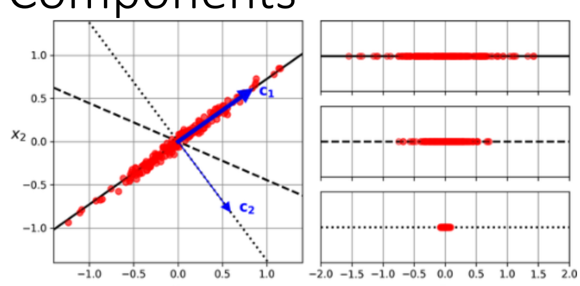


Figure 8-2. A 3D dataset lying close to a 2D subspace

7

7

Principal Components



PCA identifies the axis that accounts for the largest amount of variance in the training set. In Figure 8-7, it is the solid line. It also finds a second axis, orthogonal to the first one, that accounts for the largest amount of remaining variance. In this 2D example there is no choice: it is the dotted line. If it were a higher-dimensional dataset, PCA would also find a third axis, orthogonal to both previous axes, and a fourth, a fifth, and so on—as many axes as the number of dimensions in the dataset.

The i^{th} axis is called the i^{th} *principal component* (PC) of the data. In Figure 8-7, the first PC is the axis on which vector $\mathbf{c}_1$ lies, and the second PC is the axis on which vector $\mathbf{c}_2$ lies. In Figure 8-2 the first two PCs are the orthogonal axes on which the two arrows lie, on the plane, and the third PC is the axis orthogonal to that plane.

8

8

How to Find Principal Components?

- So, how can you find the principal components of a training set? See APPENDIX A
- Standard matrix factorization technique called **Singular Value Decomposition (SVD)**
  - Can decompose the training set matrix **X** into the matrix multiplication of three matrices $\mathbf{U}\ \mathbf{\Sigma}\ \mathbf{V}^T$, where **V** contains the unit vectors that define all the principal components:

$$\mathbf{V} = \begin{pmatrix} | & | & & | \\ \mathbf{c}_1 & \mathbf{c}_2 & \cdots & \mathbf{c}_d \\ | & | & & | \end{pmatrix}$$

- **Projecting down to *q* dimensions**
  - Once you have identified all the principal components, you can reduce the dimensionality of the dataset down to *q* dimensions by projecting it onto the hyperplane defined by the first *q* principal components.
  - Compute the matrix multiplication of the training set matrix **X** by the matrix $\mathbf{V}_q$, defined as the matrix containing the first *q* columns of **V**:

$$\mathbf{X}_{proj} = \mathbf{X}\mathbf{V}_q$$

9

9

NumPy

The following Python code uses NumPy's `svd()` function to obtain all the principal components of the training set, then extracts the two unit vectors that define the first two PCs:

```
X_centered = X - X.mean(axis=0)
U, s, Vt = np.linalg.svd(X_centered)
c1 = Vt.T[:, 0]
c2 = Vt.T[:, 1]
```

The following Python code projects the training set onto the plane defined by the first two principal components:

```
W2 = Vt.T[:, :2]
X2D = X_centered.dot(W2)
```

SciKit-Learn

Scikit-Learn's `PCA` class uses SVD decomposition to implement PCA, just like we did earlier in this chapter. The following code applies PCA to reduce the dimensionality of the dataset down to two dimensions (note that it automatically takes care of centering the data):

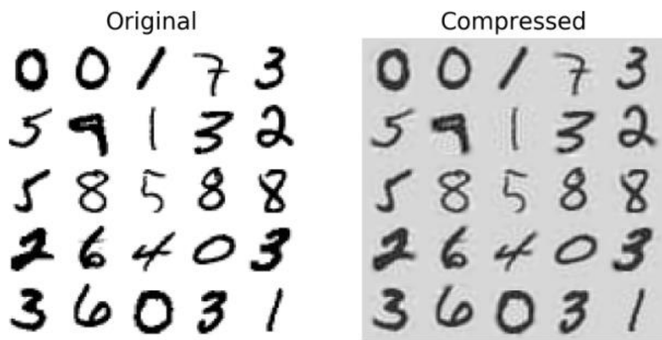
```
from sklearn.decomposition import PCA

pca = PCA(n_components = 2)
X2D = pca.fit_transform(X)
```

10

10

PCA for **compression**

After dimensionality reduction, the training set takes up much less space. As an example, try applying PCA to the MNIST dataset while preserving 95% of its variance. You should find that each instance will have just over 150 features, instead of the original 784 features. So, while most of the variance is preserved, the dataset is now less than 20% of its original size! This is a reasonable compression ratio, and you can see how this size reduction can speed up a classification algorithm (such as an SVM classifier) tremendously.



11

11

- It takes $n \times d$ memory to store data $\{x_i\}_{i=1}^{n}$ with $x_i \in \mathbb{R}^d$
- But many real data have patterns that repeat over samples. Can we find some patterns and use them?

d=32x32pixels per image

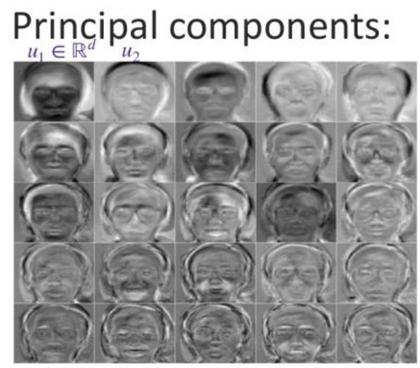
n images

$d \times n$ real values to store the data

12

12

PCA finds a compact **linear representation**



https://en.wikipedia.org/wiki/Eigenface

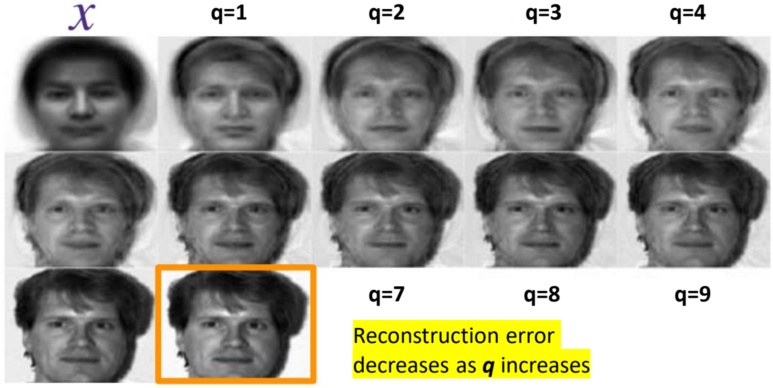
Principal components:
- Patterns that capture the distinct features of the samples
- We can represent each sample as a **weighted linear combination** of, say, ***q=25*** principal components, and just store the **weights**, *z[1..25]*

$$\approx z[1]u_1 + z[2]u_2 + \cdots + z[25]u_{25}$$

- With ***q=25***, to store ***n*** images, it requires memory of only: $d \times q + q \times n \ll d \times n$

10 principal components give a pretty good reconstruction of a face

average face $\bar{x}$ | $\bar{x} + a[1]u_1$ **q=1** | $\bar{x} + a[1]u_1 + a[2]u_2$ **q=2** | **q=3** | **q=4**



q=7 **q=8** **q=9**

q=10

Reconstruction error decreases as ***q*** increases

Ground truths real face

PCA: a high-fidelity linear projection

Basis vectors (dxq). Each column is a **basis face**

Given $x_1, \ldots, x_n \in \mathbb{R}^d$, find a compressed representation $z_1, \ldots, z_n \in \mathbb{R}^q$ with $q \ll d$ such that $x_i \approx \bar{x} + \mathbf{V}_q z_i$ and $\mathbf{V}_q^\top \mathbf{V}_q = \mathbf{I}$.

Coefficients/Weights

$$\min_{\mathbf{V}_q, \{z_i\}} \sum_{i=1}^n ||x_i - \bar{x} - \mathbf{V}_q z_i||_2^2$$

Fix $\mathbf{V}_q$ and solve for $\{z_i\}$: $\quad z_i = \mathbf{V}_q^\top (x_i - \bar{x})$

$$\widehat{x}_i := \bar{x} + \mathbf{V}_q \mathbf{V}_q^\top (x_i - \bar{x}) = \bar{x} + \sum_{j=1}^q v_j v_j^\top (x_i - \bar{x})$$

Projection matrix: projects onto q-dimensional linear subspace embedded into original high d-dimensional space

15

15

PCA: a high-fidelity linear projection

Given $x_1, \ldots, x_n \in \mathbb{R}^d$, find a compressed representation $z_1, \ldots, z_n \in \mathbb{R}^q$ with $q \ll d$ such that $x_i \approx \bar{x} + \mathbf{V}_q z_i$ and $\mathbf{V}_q^\top \mathbf{V}_q = \mathbf{I}$.

$$\min_{\mathbf{V}_q, \{z_i\}} \sum_{i=1}^n ||x_i - \bar{x} - \mathbf{V}_q z_i||_2^2$$

Fix $\mathbf{V}_q$ and solve for $\{z_i\}$: $\quad z_i = \mathbf{V}_q^\top (x_i - \bar{x})$

$$\widehat{x}_i := \bar{x} + \mathbf{V}_q \mathbf{V}_q^\top (x_i - \bar{x}) = \bar{x} + \sum_{j=1}^q v_j v_j^\top (x_i - \bar{x})$$

$$\min_{\mathbf{V}_q} \sum_{i=1}^N ||(x_i - \bar{x}) - \mathbf{V}_q \mathbf{V}_q^\top (x_i - \bar{x})||_2^2$$

$\mathbf{V}_q \mathbf{V}_q^T$ is a *projection matrix* that minimizes error in basis of size q

16

16

PCA: a high-fidelity linear projection

$$\min_{\mathbf{V}_q} \sum_{i=1}^{N} ||(x_i - \bar{x}) - \mathbf{V}_q \mathbf{V}_q^\top (x_i - \bar{x})||_2^2$$

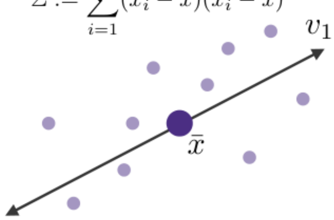
$\mathbf{V}_q \mathbf{V}_q^T$ is a *projection matrix* that minimizes error in basis of size q

$$\widehat{x}_i := \bar{x} + \mathbf{V}_q \mathbf{V}_q^\top (x_i - \bar{x}) = \bar{x} + \sum_{j=1}^{q} v_j \langle v_j, x_i - \bar{x} \rangle$$

Case when $q = 1$

$$\begin{aligned} v_1 &= \arg \min_{v:\|v\|_2=1} \sum_{i=1}^{N} \|(x_i - \bar{x}) - vv^\top (x_i - \bar{x})\|_2^2 \\ &= \arg \min_{v:\|v\|_2=1} \sum_{i=1}^{N} \|x_i - \bar{x}\|_2^2 - 2(x_i - \bar{x})^\top vv^\top (x_i - \bar{x}) \\ &\qquad + (x_i - \bar{x})^\top vv^\top vv^\top (x_i - \bar{x}) \\ &= \arg \min_{v:\|v\|_2=1} \sum_{i=1}^{N} \|x_i - \bar{x}\|_2^2 - \sum_{i=1}^{N} (x_i - \bar{x})^\top vv^\top (x_i - \bar{x}) \\ &= \arg \max_{v:\|v\|_2=1} \sum_{i=1}^{N} (x_i - \bar{x})^\top vv^\top (x_i - \bar{x}) \\ &= \arg \max_{v:\|v\|_2=1} v^\top \Sigma v \end{aligned}$$

$$\Sigma := \sum_{i=1}^{N} (x_i - \bar{x})(x_i - \bar{x})^T$$



17

17

PCA: a high-fidelity linear projection

$$\min_{\mathbf{V}_q} \sum_{i=1}^{N} ||(x_i - \bar{x}) - \mathbf{V}_q \mathbf{V}_q^\top (x_i - \bar{x})||_2^2$$

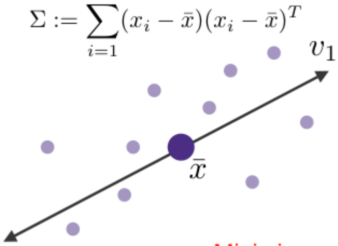
$\mathbf{V}_q \mathbf{V}_q^T$ is a *projection matrix* that minimizes error in basis of size q

$$\widehat{x}_i := \bar{x} + \mathbf{V}_q \mathbf{V}_q^\top (x_i - \bar{x}) = \bar{x} + \sum_{j=1}^{q} v_j \langle v_j, x_i - \bar{x} \rangle$$

General $q \geq 1$

$$\min_{\mathbf{V}_q} \sum_{i=1}^{N} ||(x_i - \bar{x}) - \mathbf{V}_q \mathbf{V}_q^T (x_i - \bar{x})||_2^2 = \min_{\mathbf{V}_q} Tr(\Sigma) - Tr(\mathbf{V}_q^T \Sigma \mathbf{V}_q)$$

$$\Sigma := \sum_{i=1}^{N} (x_i - \bar{x})(x_i - \bar{x})^T$$



See APPENDIX B

$\mathbf{V}_q$ are the first q eigenvectors of Σ

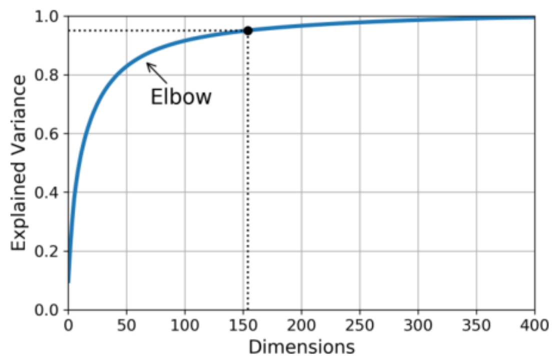
Minimize reconstruction error = capture the most variance in your data.

18

18

Choosing the right number of dimensions *q*

- Instead of arbitrarily choosing the number of dimensions to reduce down to, it is simpler to choose the number of dimensions that add up to a sufficiently large portion of the variance (e.g., 95%)
- Plot the explained variance as a function of the number of dimensions



19

19

Summary table

Quantity	Symbol	Shape	Meaning
Principal components	V_q	$d \times q$	Basis for subspace
Projection matrix	$P_q = V_q V_q^\top$	$d \times d$	Projects vectors into span(V_q)
Projected data (in subspace)	$Z = XV_q$	$n \times q$	Low-dimensional representation
Reconstructed data	$\hat{X} = XV_q V_q^\top$	$n \times d$	Projection of X back into original space

20

20

PART 2

Code Time!

21

21

Code Time

- See demonstration and discussion in class.
- See also links in the “Code Examples” for this lecture assignment.

22

22

Conclusion
Takeaways

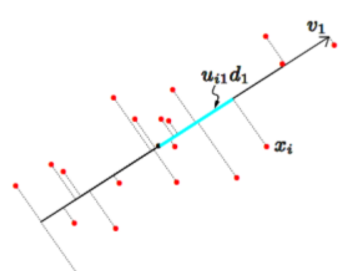
PCA: a high-fidelity linear projection

Given $x_i \in \mathbb{R}^d$ and some $q < d$ consider

$$\min_{\mathbf{V}_q} \sum_{i=1}^{N} ||(x_i - \bar{x}) - \mathbf{V}_q \mathbf{V}_q^T (x_i - \bar{x})||^2.$$

where $\mathbf{V}_q = [v_1, v_2, \ldots, v_q]$ is orthonormal:

$$\mathbf{V}_q^T \mathbf{V}_q = I_q$$



$\mathbf{V}_q$ are the first q eigenvectors of Σ

$\mathbf{V}_q$ are the first q *principal components*

$$\Sigma := \sum_{i=1}^{N} (x_i - \bar{x})(x_i - \bar{x})^T$$

Principal Component Analysis (PCA) projects $(\mathbf{X} - \mathbf{1}\bar{x}^T)$ down onto $\mathbf{V}_q$

$$(\mathbf{X} - \mathbf{1}\bar{x}^T)\mathbf{V}_q = \mathbf{U}_q \text{diag}(d_1, \ldots, d_q) \qquad \mathbf{U}_q^T \mathbf{U}_q = I_q$$

References and Credits

Many of the teaching materials for this course have been adapted from various sources. We are very grateful and thank the following professors, researchers, and practitioners for sharing their teaching materials (in no particular order):

- Yaser S. Abu-Mostafa, Malik Magdon-Ismail and Hsuan-Tien Lin. https://amlbook.com/slides.html
- Ethem Alpaydin. https://www.cmpe.boun.edu.tr/~ethem/i2ml3e/
- Natasha Jaques. https://courses.cs.washington.edu/courses/cse446/25sp/
- Lyle Ungar. https://alliance.seas.upenn.edu/~cis520/dynamic/2022/wiki/index.php?n=Lectures.Lectures
- Aurelien Geron. https://github.com/ageron/handson-ml3
- Sebastian Raschka. https://github.com/rasbt/machine-learning-book
- Trevor Hastie. https://www.statlearning.com/resources-python
- Andrew Ng. https://www.youtube.com/playlist?list=PLoROMvodv4rMiGQp3WXShtMGgzqpfVfbU
- Richard Povineli. https://www.richard.povinelli.org/teaching
- … and many others.

25

25

Appendix A: Derivation of the connection between PCA and the Singular Value Decomposition (SVD)

$$\text{PCA of } X \iff \text{SVD of centered } X, \quad \text{with } V = \text{eigenvectors of } X^\top X,\ \lambda_i = \sigma_i^2/(n-1).$$

1) Start with centered data

Let

$$X \in \mathbb{R}^{n\times d}, \quad \text{with each column mean-centered.}$$

2) PCA formulation

PCA finds orthonormal directions $V = [v_1, v_2, \ldots, v_d]$ that diagonalize the **sample covariance matrix**:

$$S = \frac{1}{n-1} X^\top X = V \Lambda V^\top,$$

where $\Lambda = \mathrm{diag}(\lambda_1, \ldots, \lambda_d)$ contains eigenvalues (variances).

Each **principal component** direction v_i satisfies

$$S v_i = \lambda_i v_i.$$

26

26

3) Now introduce the SVD

Take the **Singular Value Decomposition** of the centered data matrix:

$$\boxed{X = U\Sigma V^\top}$$

where:

- $U \in \mathbb{R}^{n\times r}$: left singular vectors (orthonormal),
- $V \in \mathbb{R}^{d\times r}$: right singular vectors (orthonormal),
- $\Sigma = \mathrm{diag}(\sigma_1, \dots, \sigma_r)$: singular values,
  with $r = \mathrm{rank}(X)$.

4) Relate SVD to covariance

Compute the covariance matrix:

$$S = \frac{1}{n-1} X^\top X = \frac{1}{n-1}(V\Sigma^\top U^\top)(U\Sigma V^\top) = \frac{1}{n-1} V\Sigma^2 V^\top.$$

So the eigenvectors of S are exactly the **right singular vectors** V of X,

and the eigenvalues of S are related to the singular values by

$$\boxed{\lambda_i = \frac{\sigma_i^2}{n-1}}.$$

27

27

5) Truncated (q-dimensional) PCA via SVD

To reduce dimensionality to q principal components, keep only the top q singular triplets:

$$X \approx U_q \Sigma_q V_q^\top.$$

Then:

- Projected data: $Z_q = XV_q = U_q\Sigma_q$,
- Reconstruction: $\hat{X}_q = U_q\Sigma_q V_q^\top = XV_qV_q^\top$.

This is the **best rank-q** approximation of X (Eckart–Young–Mirsky theorem).

Summary

Concept	PCA expression	SVD expression	Relationship
Covariance	$S = \frac{1}{n-1}X^\top X$	$S = V\frac{\Sigma^2}{n-1}V^\top$	v_i are right singular vectors
Eigenvalues / variances	λ_i	$\lambda_i = \sigma_i^2/(n-1)$	singular ↔ variance
Principal directions	V_q	right singular vectors	identical
Principal components	$Z = XV_q$	$U_q\Sigma_q$	identical coordinates
Reconstruction	$XV_qV_q^\top$	$U_q\Sigma_qV_q^\top$	same projection
Rank-q optimum	—	best rank-q SVD approximation	Eckart–Young theorem

28

28

Appendix B: Minimizing reconstruction error is equivalent to maximizing the variance of the projected datapoints

- $X \in \mathbb{R}^{n \times d}$ is **column-centered**.
- Sample covariance $S = \frac{1}{n-1} X^\top X$.
- Let $V = [v_1, \dots, v_d]$ be the orthonormal eigenvectors of S with eigenvalues $\lambda_1 \geq \cdots \geq \lambda_d \geq 0$.
- Let $V_q = [v_1, \dots, v_q] \in \mathbb{R}^{d \times q}$ (orthonormal: $V_q^\top V_q = I_q$).
- Projection/reconstruction in the original space: $\widehat{X}_q = X V_q V_q^\top$.

29

29

1) Reconstruction error with $V_q V_q^\top$

$$
\begin{aligned}
\|X - X V_q V_q^\top\|_F^2 &= \|X(I - V_q V_q^\top)\|_F^2 \\
&= \operatorname{tr}\Big((I - V_q V_q^\top)^\top X^\top X (I - V_q V_q^\top)\Big) \\
&= \operatorname{tr}(X^\top X) - \operatorname{tr}\big(V_q^\top X^\top X V_q\big),
\end{aligned}
$$

because $V_q V_q^\top$ is an orthogonal projector $(V_q V_q^\top)^2 = V_q V_q^\top$ and $\operatorname{tr}(AB) = \operatorname{tr}(BA)$.

Thus minimizing reconstruction error over all q-dimensional orthonormal bases V_q is equivalent to

$$
\boxed{\min_{V_q^\top V_q = I_q} \|X - X V_q V_q^\top\|_F^2 \iff \max_{V_q^\top V_q = I_q} \operatorname{tr}\big(V_q^\top X^\top X V_q\big).}
$$

30

30

2) Projected variance equals the trace term

Let the projected data be $Y = XV_q \in \mathbb{R}^{n \times q}$. Since X is centered,

$$\mathrm{Var}(Y) \;=\; \sum_{j=1}^{q} \mathrm{Var}(Y_{\cdot j}) \;=\; \frac{1}{n-1}\|Y\|_F^2 \;=\; \frac{1}{n-1}\,\mathrm{tr}(Y^\top Y) \;=\; \frac{1}{n-1}\,\mathrm{tr}\big(V_q^\top X^\top X V_q\big).$$

Hence

$$\boxed{\max_{V_q^\top V_q = I_q} \mathrm{tr}\big(V_q^\top X^\top X V_q\big) \iff \max_{V_q^\top V_q = I_q} \mathrm{Var}(XV_q).}$$

31

3) Equivalence and PCA solution

Combining 1) and 2):

$$\min_{V_q^\top V_q = I_q} \|X - XV_qV_q^\top\|_F^2 \quad \iff \quad \max_{V_q^\top V_q = I_q} \mathrm{Var}(XV_q).$$

By the **Ky Fan maximum principle**, the maximizer is $V_q = [v_1, \ldots, v_q]$, the **top q eigenvectors** of S. Consequently,

$$\|X - XV_qV_q^\top\|_F^2 = (n-1)\sum_{j=q+1}^{d} \lambda_j, \qquad \mathrm{Var}(XV_q) = \sum_{j=1}^{q} \lambda_j.$$

Takeaway

Using V_q explicitly:

- $P_q = V_qV_q^\top$ is the orthogonal projector onto the q-dimensional PCA subspace.
- **Minimizing** reconstruction error $\|X - XV_qV_q^\top\|_F^2$ is **equivalent** to **maximizing** the total variance of the projected data XV_q, attained by choosing V_q as the top q eigenvectors of the covariance matrix.

32