Dimensionality reduction

Introduction and PCA tutorial

Amit Moscovich Eiger
moscovich@gmail.com

Dimensionality reduction framework

Given inputs $\mathbf{x}_1, \ldots, \mathbf{x}_n \in \mathbb{R}^p$

1. *Somehow* transform to lower-dimensional vectors

$$\mathbf{x}_i \in \mathbb{R}^p \longrightarrow \mathbf{x}'_i \in \mathbb{R}^q$$

2. Forget $\mathbf{x}_1, \ldots, \mathbf{x}_n$.
3. Work with $\mathbf{x}'_1, \ldots, \mathbf{x}'_n$ instead.

Motivation 1: Data visualization

C	A	MA	Ash	$AshA$	Mag	Phe	Fl	NFP	Pro	Col	Hue	OD	$Prol$
1	14.23	1.71	2.43	15.6	127	2.8	3.06	.28	2.29	5.64	1.04	3.92	1065
1	13.2	1.78	2.14	11.2	100	2.65	2.76	.26	1.28	4.38	1.05	3.4	1050
1	13.16	2.36	2.67	18.6	101	2.8	3.24	.3	2.81	5.68	1.03	3.17	1185
1	14.37	1.95	2.5	16.8	113	3.85	3.49	.24	2.18	7.8	.86	3.45	1480
1	13.24	2.59	2.87	21	118	2.8	2.69	.39	1.82	4.32	1.04	2.93	735
2	12.37	.94	1.36	10.6	88	1.98	.57	.28	.42	1.95	1.05	1.82	520
2	12.33	1.1	2.28	16	101	2.05	1.09	.63	.41	3.27	1.25	1.67	680
2	12.64	1.36	2.02	16.8	100	2.02	1.41	.53	.62	5.75	.98	1.59	450
2	13.67	1.25	1.92	18	94	2.1	1.79	.32	.73	3.8	1.23	2.46	630
2	12.37	1.13	2.16	19	87	3.5	3.1	.19	1.87	4.45	1.22	2.87	420
3	13.73	4.36	2.26	22.5	88	1.28	.47	.52	1.15	6.62	.78	1.75	520
3	13.45	3.7	2.6	23	111	1.7	.92	.43	1.46	10.68	.85	1.56	695
3	12.82	3.37	2.3	19.5	88	1.48	.66	.4	.97	10.26	.72	1.75	685
3	13.58	2.58	2.69	24.5	105	1.55	.84	.39	1.54	8.66	.74	1.8	750
3	13.4	4.6	2.86	25	112	1.98	.96	.27	1.11	8.5	.67	1.92	630

Figure: Wine recognition data set (sample)

Motivation 1: Data visualization

IsraelTech challenge

C	A	MA	Ash	$AshA$	Mag	Phe	Fl	NFP	Pro	Col	Hue	OD	$Prol$
1	14.23	1.71	2.43	15.6	127	2.8	3.06	.28	2.29	5.64	1.04	3.92	1065
1	13.2	1.78	2.14	11.2	100	2.65	2.76	.26	1.28	4.38	1.05	3.4	1050
1	13.16	2.36	2.67	18.6	101	2.8	3.24	.3	2.81	5.68	1.03	3.17	1185
1	14.37	1.95	2.5	16.8	113	3.85	3.49	.24	2.18	7.8	.86	3.45	1480
1	13.24	2.59	2.87	21	118	2.8	2.69	.39	1.82	4.32	1.04	2.93	735
2	12.37	.94	1.36	10.6	88	1.98	.57	.28	.42	1.95	1.05	1.82	520
2	12.33	1.1	2.28	16	101	2.05	1.09	.63	.41	3.27	1.25	1.67	680
2	12.64	1.36	2.02	16.8	100	2.02	1.41	.53	.62	5.75	.98	1.59	450
2	13.67	1.25	1.92	18	94	2.1	1.79	.32	.73	3.8	1.23	2.46	630
2	12.37	1.13	2.16	19	87	3.5	3.1	.19	1.87	4.45	1.22	2.87	420
3	13.73	4.36	2.26	22.5	88	1.28	.47	.52	1.15	6.62	.78	1.75	520
3	13.45	3.7	2.6	23	111	1.7	.92	.43	1.46	10.68	.85	1.56	695
3	12.82	3.37	2.3	19.5	88	1.48	.66	.4	.97	10.26	.72	1.75	685
3	13.58	2.58	2.69	24.5	105	1.55	.84	.39	1.54	8.66	.74	1.8	750
3	13.4	4.6	2.86	25	112	1.98	.96	.27	1.11	8.5	.67	1.92	630

Figure: Wine recognition data set (sample)

High dimensional sets of vectors are difficult to understand!

Motivation 1: Data visualization

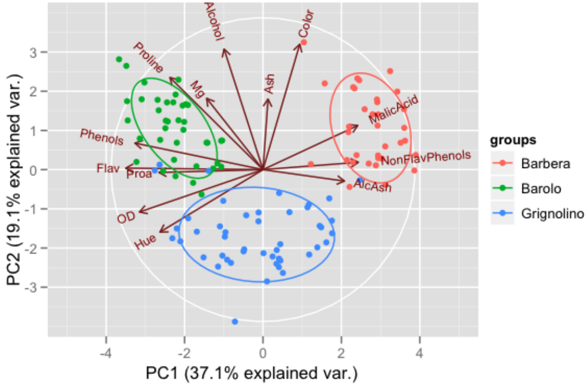
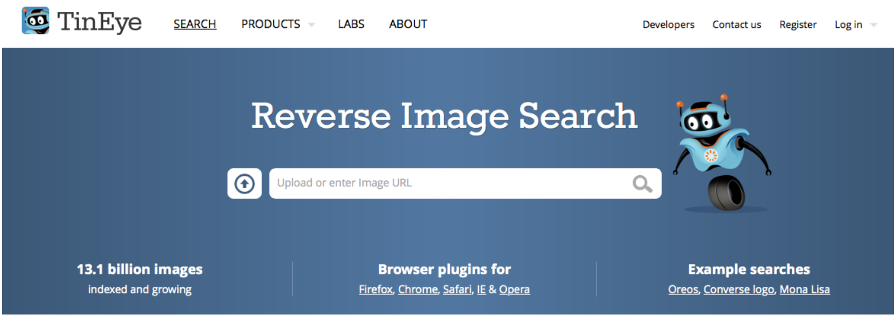


Figure: Biplot of wine data set

Motivation 2: Reducing computation/storage



Motivation 3: Compression

Many lossy compressions involve dimensionality reduction. e.g.

- JPEG
- MPEG

Motivation 4: Improving statistical performance

Recall the curse of dimensionality in nonparametric statistical methods (e.g. knn)

Motivation 4: Improving statistical performance

Recall the curse of dimensionality in nonparametric statistical methods (e.g. knn)

So, why not:

- Reduce the dimension of the samples
- Learn using the lower-dimensional samples

Motivation 4: Improving statistical performance

Recall the curse of dimensionality in nonparametric statistical methods (e.g. knn)

So, why not:

- Reduce the dimension of the samples
- Learn using the lower-dimensional samples

This often improves the statistical performance!

Dimensionality reduction by dropping coordinates

One way to reduce the dimension of vectors is to keep only specific coordinates, e.g.

$$(\alpha_1, \alpha_2, \alpha_3, \alpha_4) \mapsto (\alpha_2, \alpha_4)$$

This is known as *feature selection*

Dimensionality reduction by dropping coordinates

One way to reduce the dimension of vectors is to keep only specific coordinates, e.g.

$$(\alpha_1, \alpha_2, \alpha_3, \alpha_4) \mapsto (\alpha_2, \alpha_4)$$

This is known as *feature selection*

Q: Which coordinates should we keep?

Dimensionality reduction by averaging coordinates

Another idea is to average coordinates.

$$(\alpha_1, \alpha_2, \alpha_3, \alpha_4) \mapsto \left(\frac{\alpha_1 + \alpha_2}{2}, \frac{\alpha_3 + \alpha_4}{2} \right)$$

Dimensionality reduction by averaging coordinates

Another idea is to average coordinates.

$$(\alpha_1, \alpha_2, \alpha_3, \alpha_4) \mapsto \left(\frac{\alpha_1 + \alpha_2}{2}, \frac{\alpha_3 + \alpha_4}{2} \right)$$

Q: When does this make sense?

Dimensionality reduction by averaging coordinates

Figure: A squirrel (640×434 pixels)

Dimensionality reduction by averaging coordinates

Figure: 320×217 pixels, still a squirrel!

Dimensionality reduction by dropping coordinates

This transformation

$$(\alpha_1, \alpha_2, \alpha_3, \alpha_4) \mapsto (\alpha_2, \alpha_4)$$

can be written as

$$\mathbf{x} \mapsto (\mathbf{e}_2^T \mathbf{x}, \mathbf{e}_4^T \mathbf{x}).$$

Dimensionality reduction by dropping coordinates

This transformation

$$(\alpha_1, \alpha_2, \alpha_3, \alpha_4) \mapsto (\alpha_2, \alpha_4)$$

can be written as

$$\mathbf{x} \mapsto (\mathbf{e}_2^T \mathbf{x}, \mathbf{e}_4^T \mathbf{x}).$$

The resulting (α_2, α_4) are coordinates of the orthogonal projection of $\mathbf{x}$ onto $H = sp\{\mathbf{e}_2, \mathbf{e}_4\}$.

Dimensionality reduction by averaging coordinates

This transformation

$$(\alpha_1, \alpha_2, \alpha_3, \alpha_4) \mapsto \left(\frac{\alpha_1 + \alpha_2}{2}, \frac{\alpha_3 + \alpha_4}{2} \right)$$

can be written as

$$\mathbf{x} \mapsto \left(\frac{\mathbf{e}_1^T + \mathbf{e}_2^T}{2} \mathbf{x}, \frac{\mathbf{e}_3^T + \mathbf{e}_4^T}{2} \mathbf{x} \right).$$

Dimensionality reduction by averaging coordinates

This transformation

$$(\alpha_1, \alpha_2, \alpha_3, \alpha_4) \mapsto \left(\frac{\alpha_1 + \alpha_2}{2}, \frac{\alpha_3 + \alpha_4}{2} \right)$$

can be written as

$$\mathbf{x} \mapsto \left(\frac{\mathbf{e}_1^T + \mathbf{e}_2^T}{2}\mathbf{x}, \frac{\mathbf{e}_3^T + \mathbf{e}_4^T}{2}\mathbf{x} \right).$$

Up to a constant, these are just the coordinates of the orthogonal projection of $\mathbf{x}$ onto $H = sp\{\mathbf{e}_1 + \mathbf{e}_2, \mathbf{e}_3 + \mathbf{e}_4\}$.

Dimensionality reduction using linear projections

We just saw 2 examples of dimensionality reduction by linear projections.

Dimensionality reduction using linear projections

We just saw 2 examples of dimensionality reduction by linear projections.

Linear projections are great!

- Fast to compute.
- Easy to understand.
- Easy to analyze mathematically.

Dimensionality reduction using linear projections

We just saw 2 examples of dimensionality reduction by linear projections.

Linear projections are great!

- Fast to compute.
- Easy to understand.
- Easy to analyze mathematically.

But onto which linear subspace should we project?

Dimensionality reduction using linear projections

We just saw 2 examples of dimensionality reduction by linear projections.

Linear projections are great!

- Fast to compute.
- Easy to understand.
- Easy to analyze mathematically.

But onto which linear subspace should we project?

PCA gives one answer. But first let us recall the mathematics of orthogonal projections.

Orthogonal projections

Let $H \subset \mathbb{R}^p$ be a linear subspace.

Orthogonal projections

Let $H \subset \mathbb{R}^p$ be a linear subspace.
We wish to compute orthogonal projections on H.

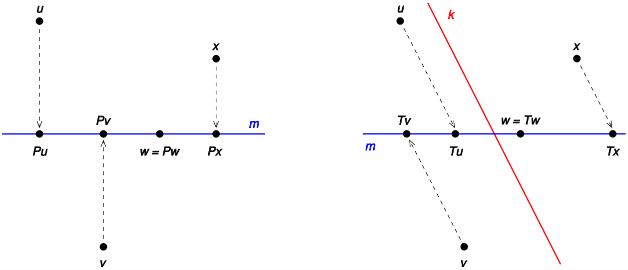


Figure: Orthogonal vs. non-orthogonal projection of 2D points

Orthogonal projections

Given any *orthonormal* basis of H

$$\{\mathbf{b}_1, \dots, \mathbf{b}_q\}$$

The orthogonal projection operator is:

$$\boxed{P_H = \mathbf{b}_1\mathbf{b}_1^T + \dots \mathbf{b}_q\mathbf{b}_q^T}$$

Orthogonal projections

Given any *orthonormal* basis of H

$$\{\mathbf{b}_1, \ldots, \mathbf{b}_q\}$$

The orthogonal projection operator is:

$$\boxed{P_H = \mathbf{b}_1 \mathbf{b}_1^T + \ldots \mathbf{b}_q \mathbf{b}_q^T}$$

Why does this work?

Orthogonal projections

First, complete $\{\mathbf{b}_1, \ldots, \mathbf{b}_q\}$ to an orthonormal basis of $\mathbb{R}^p$

$$\{\mathbf{b}_1, \ldots, \mathbf{b}_q, \ldots, \mathbf{b}_p\}$$

Orthogonal projections

First, complete $\{\mathbf{b}_1, \dots, \mathbf{b}_q\}$ to an orthonormal basis of $\mathbb{R}^p$

$$\{\mathbf{b}_1, \dots, \mathbf{b}_q, \dots, \mathbf{b}_p\}$$

Let $\mathbf{x} \in \mathbb{R}^p$ be some vector, express $\mathbf{x}$ in this basis

$$\mathbf{x} = \alpha_1 \mathbf{b}_1 + \dots \alpha_p \mathbf{b}_p$$

Orthogonal projections

$$
\begin{aligned}
P_H\mathbf{x} &= P_H\left(\alpha_1\mathbf{b}_1 + \ldots \alpha_p\mathbf{b}_p\right) \\
&= \left(\sum_{i=1}^{q}\mathbf{b}_i\mathbf{b}_i^T\right)\left(\sum_{j=1}^{p}\alpha_j\mathbf{b}_j\right) \\
&= \sum_{i=1}^{q}\sum_{j=1}^{p}\alpha_j\mathbf{b}_i\underbrace{\mathbf{b}_i^T\mathbf{b}_j}_{=\delta_{ij}} \\
&= \sum_{i=1}^{q}\alpha_i\mathbf{b}_i
\end{aligned}
$$

Orthogonal projections

We obtain the orthogonal decomposition of $\mathbf{x}$

$$\mathbf{x} = \underbrace{\alpha_1 \mathbf{b}_1 + \ldots \alpha_q \mathbf{b}_q}_{P_H \mathbf{x} \in H} + \underbrace{\alpha_{q+1} \mathbf{b}_{q+1} + \ldots + \alpha_p \mathbf{b}_p}_{\mathbf{x} - P_H \mathbf{x} \,\in\, H^{\perp}}$$

Orthogonal projections

Israeltech challenge

We obtain the orthogonal decomposition of $\mathbf{x}$

$$\mathbf{x} = \underbrace{\alpha_1 \mathbf{b}_1 + \ldots \alpha_q \mathbf{b}_q}_{P_H \mathbf{x} \in H} + \underbrace{\alpha_{q+1} \mathbf{b}_{q+1} + \ldots + \alpha_p \mathbf{b}_p}_{\mathbf{x} - P_H \mathbf{x} \ \in \ H^{\perp}}$$

The coordinates $P_H \mathbf{x}$ in the basis $\{\mathbf{b}_1, \ldots, \mathbf{b}_q\}$ of H are

$$(\alpha_1, \ldots, \alpha_q) = (\mathbf{b}_1^T \mathbf{x}, \ldots, \mathbf{b}_q^T \mathbf{x})$$

Orthogonal projections

Q: What if we want to project $\mathbf{x}$ on an affine subspace?

$$F = \mathsf{Sp}\{\mathbf{b}_1, \ldots, \mathbf{b}_q\} + \ \mathbf{b}$$

Orthogonal projections

Q: What if we want to project $\mathbf{x}$ on an affine subspace?

$$F = \mathsf{Sp}\{\mathbf{b}_1, \ldots, \mathbf{b}_q\} + \mathbf{b}$$

A: No problem! Subtract $\mathbf{b}$ and add it later.

$$P_F(\mathbf{x}) = \mathbf{b}_1\mathbf{b}_1^T(\mathbf{x} - \mathbf{b}) + \ldots \mathbf{b}_q\mathbf{b}_q^T(\mathbf{x} - \mathbf{b}) + \mathbf{b}$$

Principal Component Analysis

PCA is the most popular method of dim. reduction.

- It is well understood mathematically
- Simple to implement
- Fast

PCA: definition

Input:

- Points $\mathbf{x}_1, \ldots, \mathbf{x}_n \in \mathbb{R}^p$ *zero centered* vectors (i.e. $\bar{\mathbf{x}} = \frac{1}{n} \sum \mathbf{x}_i = 0$)
- Desired output dimension q.

PCA: definition

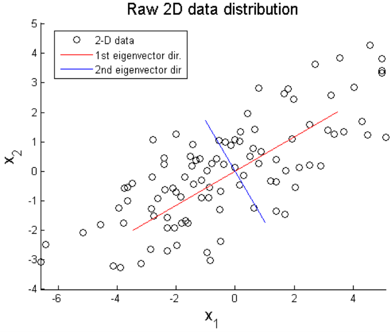
Input:

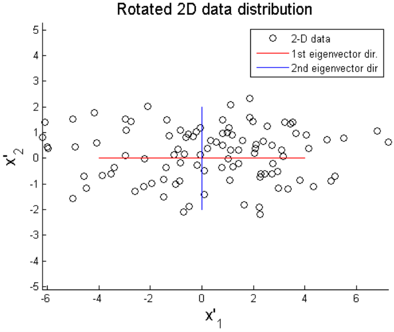
- Points $\mathbf{x}_1, \ldots, \mathbf{x}_n \in \mathbb{R}^p$ *zero centered* vectors (i.e. $\bar{\mathbf{x}} = \frac{1}{n} \sum \mathbf{x}_i = 0$)
- Desired output dimension q.

Output:

- Orthonormal set of vectors $\mathbf{u}_1, \ldots, \mathbf{u}_q$ of the most important "directions" of the data.

PCA: definition





PCA: definition

Projecting $T = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$ along $\mathbf{u}_1$ yields the maximum variance.

$$\mathbf{u}_1 = \underset{\mathbf{u}:\|\mathbf{u}\|=1}{\operatorname{argmax}} \operatorname{Var}\left(\mathbf{u}^T T\right)$$

PCA: definition

Projecting $T = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$ along $\mathbf{u}_1$ yields the maximum variance.

$$\mathbf{u}_1 = \underset{\mathbf{u}:\|\mathbf{u}\|=1}{\operatorname{argmax}} \operatorname{Var}\left(\mathbf{u}^T T\right)$$

$\mathbf{u}_1$ is called the first *loadings* or *coefficients* vector of the first *principal axis*.

PCA: definition

Projecting $T = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$ along $\mathbf{u}_1$ yields the maximum variance.

$$\mathbf{u}_1 = \underset{\mathbf{u}:\|\mathbf{u}\|=1}{\operatorname{argmax}} \operatorname{Var}\left(\mathbf{u}^T T\right)$$

$\mathbf{u}_1$ is called the first *loadings* or *coefficients* vector of the first *principal axis*.

The new coordinate $\mathbf{u}_1^T \mathbf{x}_i$ is called the first *principal component* of $\mathbf{x}_i$

PCA: definition

$\mathbf{u}_2$ is the direction with highest variance that is orthogonal to $\mathbf{u}_1$:

$$\mathbf{u}_2 = \operatorname*{argmax}_{\mathbf{u}:\ \|\mathbf{u}\|=1,\ \mathbf{u}\perp\mathbf{u}_1} \mathrm{Var}\left(\mathbf{u}^T T\right).$$

PCA: definition

$\mathbf{u}_2$ is the direction with highest variance that is orthogonal to $\mathbf{u}_1$:

$$\mathbf{u}_2 = \underset{\mathbf{u}:\ \|\mathbf{u}\|=1,\ \mathbf{u}\perp\mathbf{u}_1}{\operatorname{argmax}} \operatorname{Var}\left(\mathbf{u}^T T\right).$$

And so on:

$$\mathbf{u}_3 = \underset{\mathbf{u}:\ \|\mathbf{u}\|=1,\ \mathbf{u}\perp\mathbf{u}_1,\ \mathbf{u}\perp\mathbf{u}_2}{\operatorname{argmax}} \operatorname{Var}\left(\mathbf{u}^T T\right).$$

etc.

PCA: how to find u_1

Relevant fact:

$$\mathrm{Var}\left(\mathbf{u}^T T\right) = \mathbf{u}^T \Sigma \mathbf{u}$$

where Σ is the *sample covariance matrix*.

PCA: how to find u_1

Relevant fact:

$$\mathrm{Var}\left(\mathbf{u}^T T\right) = \mathbf{u}^T \Sigma \mathbf{u}$$

where Σ is the *sample covariance matrix*. Thus,

$$\mathbf{u}_1 = \operatorname*{argmax}_{\mathbf{u}: \|\mathbf{u}\|=1} \mathbf{u}^T \Sigma \mathbf{u}$$

$$\mathbf{u}_2 = \operatorname*{argmax}_{\mathbf{u}:\ \|\mathbf{u}\|=1, \mathbf{u} \perp \mathbf{u}_1} \mathbf{u}^T \Sigma \mathbf{u}.$$

etc.

PCA: how to find u_1

Relevant fact:

$$\mathrm{Var}\left(\mathbf{u}^T T\right) = \mathbf{u}^T \Sigma \mathbf{u}$$

where Σ is the *sample covariance matrix*. Thus,

$$\mathbf{u}_1 = \operatorname*{argmax}_{\mathbf{u}:\|\mathbf{u}\|=1} \mathbf{u}^T \Sigma \mathbf{u}$$

$$\mathbf{u}_2 = \operatorname*{argmax}_{\mathbf{u}:\ \|\mathbf{u}\|=1, \mathbf{u}\perp\mathbf{u}_1} \mathbf{u}^T \Sigma \mathbf{u}.$$

etc.

Remember how the sample covariance is defined?

PCA: sample covariance reminder

Let $\mathbf{x}_1, \ldots, \mathbf{x}_n \in \mathbb{R}^p$ be a sample.

PCA: sample covariance reminder

Let $\mathbf{x}_1, \ldots, \mathbf{x}_n \in \mathbb{R}^p$ be a sample.
Denote the mean by $\mu = \frac{1}{n} \sum_{i=1}^{n} \mathbf{x}_i$

PCA: sample covariance reminder

Let $\mathbf{x}_1, \ldots, \mathbf{x}_n \in \mathbb{R}^p$ be a sample.
Denote the mean by $\mu = \frac{1}{n}\sum_{i=1}^{n} \mathbf{x}_i$
The sample covariance $\Sigma_{p\times p}$ is a matrix

$$\Sigma_{jk} = \frac{1}{n}\sum_{i=1}^{n}(x_{ij} - \mu_j)(x_{ik} - \mu_k)$$

PCA: sample covariance reminder

In matrix notation, let $\mathbf{z}_i = \mathbf{x}_i - \mu$ denote the zero-centered samples as column vectors, then

$$\Sigma = \frac{1}{n}\sum_{i=1}^{n}(\mathbf{x}_i - \mu)(\mathbf{x}_i - \mu)^T$$

PCA: sample covariance reminder

In matrix notation, let $\mathbf{z}_i = \mathbf{x}_i - \mu$ denote the zero-centered samples as column vectors, then

$$\Sigma = \frac{1}{n}\sum_{i=1}^{n}(\mathbf{x}_i - \mu)(\mathbf{x}_i - \mu)^T$$

Shorter yet,

$$\Sigma = \frac{1}{n}ZZ^T \quad \text{where} \quad Z_{p\times n} = \begin{pmatrix}\mathbf{z}_1 & \mathbf{z}_2 & \dots & \mathbf{z}_n\end{pmatrix}$$

PCA: how to find u_1

Consider a simple case where Σ is diagonal

$$\Sigma = \begin{pmatrix} 5 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

PCA: how to find u_1

Denote

$$\mathbf{u} = \begin{pmatrix} \alpha_1 & \alpha_2 & \alpha_3 \end{pmatrix}^T$$

Q: how to maximize

$$\mathbf{u}^T \Sigma \mathbf{u} = \begin{pmatrix} \alpha_1 & \alpha_2 & \alpha_3 \end{pmatrix} \begin{pmatrix} 5 & 0 & 0 \\ 0 & 3 & 0 \\ 0 & 0 & 2 \end{pmatrix} \begin{pmatrix} \alpha_1 \\ \alpha_2 \\ \alpha_3 \end{pmatrix} = 5\alpha_1^2 + 3\alpha_2^2 + 2\alpha_3^2$$

while keeping $\|\mathbf{u}\|^2 = \alpha_1^2 + \alpha_2^2 + \alpha_3^2 = 1$.

PCA: how to find u_1

Answer:

$$\alpha_1^2 = 1,\ \alpha_2^2 = 0,\ \alpha_3^3 = 0$$

Hence, the first loadings vector is

$$\mathbf{u}_1 = \begin{pmatrix} \alpha_1 \\ \alpha_2 \\ \alpha_3 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} = \mathbf{e}_1$$

PCA: how to find u_1

Answer:

$$\alpha_1^2 = 1,\ \alpha_2^2 = 0,\ \alpha_3^3 = 0$$

Hence, the first loadings vector is

$$\mathbf{u}_1 = \begin{pmatrix} \alpha_1 \\ \alpha_2 \\ \alpha_3 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} = \mathbf{e}_1$$

It is the eigenvector of Σ with largest eigenvalue.

PCA: how to find $\mathbf{u}_2$

$$\mathbf{u}_2 = \underset{\mathbf{u}:\ \|\mathbf{u}\|=1,\ \mathbf{u} \perp \mathbf{u}_1}{\operatorname{argmax}} \mathbf{u}^T \Sigma \mathbf{u}$$

PCA: how to find $\mathbf{u}_2$

$$\mathbf{u}_2 = \underset{\mathbf{u}:\ \|\mathbf{u}\|=1,\ \mathbf{u}\perp\mathbf{u}_1}{\operatorname{argmax}} \mathbf{u}^T \Sigma \mathbf{u}$$

From the constraints:

- $\mathbf{u}_2 \perp \mathbf{u}_1 \implies \mathbf{u}_2 = (0 \quad \beta_1 \quad \beta_2)$
- $\|\mathbf{u}_2\| = 1 \implies \beta_1^2 + \beta_2^2 = 1$

PCA: how to find $\mathbf{u}_2$

$$\mathbf{u}_2 = \underset{\mathbf{u}:\ \|\mathbf{u}\|=1,\ \mathbf{u}\perp\mathbf{u}_1}{\operatorname{argmax}} \mathbf{u}^T \Sigma \mathbf{u}$$

From the constraints:

- $\mathbf{u}_2 \perp \mathbf{u}_1 \implies \mathbf{u}_2 = (0 \quad \beta_1 \quad \beta_2)$
- $\|\mathbf{u}_2\| = 1 \implies \beta_1^2 + \beta_2^2 = 1$

Q: how to maximize $\mathbf{u}_2^T \Sigma \mathbf{u}_2 = 3\beta_1^2 + 2\beta_2^2$?

PCA: how to find $\mathbf{u}_2$

$$\mathbf{u}_2 = \underset{\mathbf{u}:\ \|\mathbf{u}\|=1,\ \mathbf{u}\perp\mathbf{u}_1}{\operatorname{argmax}} \quad \mathbf{u}^T \Sigma \mathbf{u}$$

From the constraints:

- $\mathbf{u}_2 \perp \mathbf{u}_1 \implies \mathbf{u}_2 = (0 \quad \beta_1 \quad \beta_2)$
- $\|\mathbf{u}_2\| = 1 \implies \beta_1^2 + \beta_2^2 = 1$

Q: how to maximize $\mathbf{u}_2^T \Sigma \mathbf{u}_2 = 3\beta_1^2 + 2\beta_2^2$?

Answer:

$$\mathbf{u}_2 = \mathbf{e}_2 = \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix}$$

Notice a pattern?

PCA: non-diagonal case

What if Σ is not diagonal?

PCA: non-diagonal case

What if Σ is not diagonal? No problem! Luckily Σ is always *orthogonally diagonalizable*:

$$\Sigma = R\Lambda R^T$$

where

- R is an orthogonal matrix
- Λ is a diagonal matrix with non-negative entries. It holds the eigenvalues of Σ in decreasing order.

PCA: non-diagonal case

Reminder:
an orthogonal matrix R is a real matrix that satisfies

$$RR^T = R^T R = I$$

PCA: non-diagonal case

Reminder:
an orthogonal matrix R is a real matrix that satisfies

$$RR^T = R^T R = I$$

Some properties:

- R is square
- Its rows (columns) are orthonormal vectors
- $R^{-1} = R^T$, it is also an orthogonal matrix.

PCA: non-diagonal case

Orthogonal matrices preserve distances between vectors

$$\|Rv - Ru\| = \|v - u\|.$$

Such transformations are known as *isometries* or rotation/reflection transformations.

PCA: non-diagonal case

... and back to PCA.

We wish to maximize $\mathbf{u}^T\Sigma\mathbf{u}$ where Σ is a general covariance matrix.

PCA: non-diagonal case

... and back to PCA.
We wish to maximize $\mathbf{u}^T \Sigma \mathbf{u}$ where Σ is a general covariance matrix.

Standard trick: Instead of maximizing over $\mathbf{u}$, maximize over rotated vectors $\mathbf{u} = R\mathbf{v}$ where $\Sigma = R \Lambda R^T$.

PCA: non-diagonal case

... and back to PCA.
We wish to maximize $\mathbf{u}^T\Sigma\mathbf{u}$ where Σ is a general covariance matrix.

Standard trick: Instead of maximizing over $\mathbf{u}$, maximize over rotated vectors $\mathbf{u} = R\mathbf{v}$ where $\Sigma = R\Lambda R^T$.

$$\mathbf{u}^T\Sigma\mathbf{u} = (R\mathbf{v})^T\Sigma R\mathbf{v} = \mathbf{v}^T \underbrace{R^T R}_{=I} \Lambda \underbrace{R^T R}_{=I} \mathbf{v} = \mathbf{v}^T\Lambda\mathbf{v}$$

PCA: non-diagonal case

... and back to PCA.
We wish to maximize $\mathbf{u}^T\Sigma\mathbf{u}$ where Σ is a general covariance matrix.

Standard trick: Instead of maximizing over $\mathbf{u}$, maximize over rotated vectors $\mathbf{u} = R\mathbf{v}$ where $\Sigma = R\Lambda R^T$.

$$\mathbf{u}^T\Sigma\mathbf{u} = (R\mathbf{v})^T\Sigma R\mathbf{v} = \mathbf{v}^T \underbrace{R^T R}_{=I} \Lambda \underbrace{R^T R}_{=I} \mathbf{v} = \mathbf{v}^T\Lambda\mathbf{v}$$

This reduces the problem to the easy diagonal case!

PCA: non-diagonal case

What are the loading vectors $\mathbf{u}_1, \ldots, \mathbf{u}_p$ of Σ ?

PCA: non-diagonal case

What are the loading vectors $\mathbf{u}_1, \ldots, \mathbf{u}_p$ of Σ ?

- In the rotated system - same as before $\mathbf{v}_i = \mathbf{e}_i$.

PCA: non-diagonal case

What are the loading vectors $\mathbf{u}_1, \ldots, \mathbf{u}_p$ of Σ ?

- In the rotated system - same as before $\mathbf{v}_i = \mathbf{e}_i$.
- Hence $\mathbf{u}_i = R\mathbf{v}_i = R\mathbf{e}_i$

PCA: non-diagonal case

What are the loading vectors $\mathbf{u}_1, \ldots, \mathbf{u}_p$ of Σ ?

- In the rotated system - same as before $\mathbf{v}_i = \mathbf{e}_i$.
- Hence $\mathbf{u}_i = R\mathbf{v}_i = R\mathbf{e}_i$

The loading vectors of the covariance matrix $\Sigma = R\Lambda R^T$ are the columns of R!

PCA: algorithm summary

- zero-center the samples

$$\mathbf{z}_i = \mathbf{x}_i - \mu \quad \text{where} \quad \mu = \frac{1}{n}\sum_i \mathbf{x}_i$$

- Compute sample covariance matrix

$$\Sigma = \frac{1}{n}\sum_{i=1}^{n} \mathbf{z}_i \mathbf{z}_i^T$$

- Diagonalize $\Sigma = R \Lambda R^T$

PCA: algorithm summary

- Diagonalize $\Sigma = R \Lambda R^T$.

Where $\lambda_1 \geq \lambda_2 \ldots \geq \lambda_p \geq 0$ and

$$R_{p\times p} = \begin{pmatrix} \vdots & & \vdots \\ \mathbf{u}_1 & \ldots & \mathbf{u}_p \\ \vdots & & \vdots \end{pmatrix} \quad \Lambda_{p\times p} = \begin{pmatrix} \lambda_1 & 0 & \ldots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \ldots & \lambda_p \end{pmatrix}$$

PCA: algorithm summary

- Diagonalize $\Sigma = R\Lambda R^T$.

Where $\lambda_1 \geq \lambda_2 \ldots \geq \lambda_p \geq 0$ and

$$R_{p\times p} = \begin{pmatrix} \vdots & & \vdots \\ \mathbf{u}_1 & \ldots & \mathbf{u}_p \\ \vdots & & \vdots \end{pmatrix} \quad \Lambda_{p\times p} = \begin{pmatrix} \lambda_1 & 0 & \ldots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \ldots & \lambda_p \end{pmatrix}$$

- The kth loading vector is $\mathbf{u}_k$

PCA: algorithm summary

- Diagonalize $\Sigma = R\Lambda R^T$.

Where $\lambda_1 \geq \lambda_2 \ldots \geq \lambda_p \geq 0$ and

$$R_{p\times p} = \begin{pmatrix} \vdots & & \vdots \\ \mathbf{u}_1 & \ldots & \mathbf{u}_p \\ \vdots & & \vdots \end{pmatrix} \quad \Lambda_{p\times p} = \begin{pmatrix} \lambda_1 & 0 & \ldots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \ldots & \lambda_p \end{pmatrix}$$

- The k$^{\text{th}}$ loading vector is $\mathbf{u}_k$
- For any vector $\mathbf{x}$, its k$^{\text{th}}$ PC is $\mathbf{u}_k^T(\mathbf{x} - \mu)$.

PCA: algorithm summary

Notes:

- All PCs of $\mathbf{x}$ are given by the rotation

$$R^T(\mathbf{x}-\mu) = (\mathbf{u}_1^T(\mathbf{x}-\mu), \ldots, \mathbf{u}_p^T(\mathbf{x}-\mu))^T.$$

PCA: algorithm summary

Notes:

- All PCs of $\mathbf{x}$ are given by the rotation

$$R^T(\mathbf{x}-\mu) = (\mathbf{u}_1^T(\mathbf{x}-\mu), \ldots, \mathbf{u}_p^T(\mathbf{x}-\mu))^T.$$

- Usually $q \ll p$ PCs are used.

PCA: algorithm summary

Notes:

- All PCs of $\mathbf{x}$ are given by the rotation

$$R^T(\mathbf{x}-\mu) = (\mathbf{u}_1^T(\mathbf{x}-\mu), \ldots, \mathbf{u}_p^T(\mathbf{x}-\mu))^T.$$

- Usually $q \ll p$ PCs are used.
- PCA needs just μ, Σ as input.

Visualizing the result of PCA

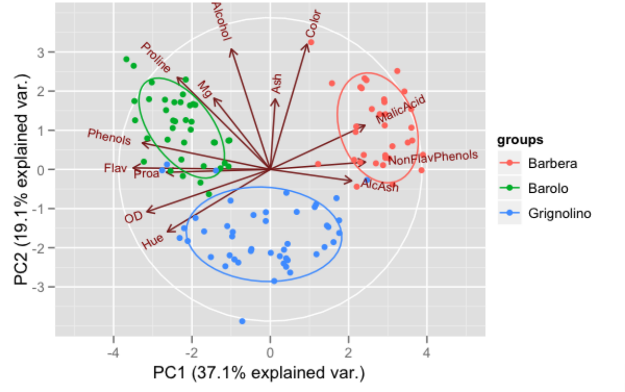


Figure: Biplot of wine data set

How many dimensions to take?

Let $\mathbf{z}_1, \ldots, \mathbf{z}_n$ be zero-centered vectors and consider the mean squared length, or *Total Variation* (TV)

$$\frac{1}{n}\sum_i \|\mathbf{z}_i\|^2 = \frac{1}{n}\sum_i\sum_j |z_{ij}|^2.$$

By Pythagoras' theorem $\|\mathbf{z}_i\|^2 = z_{i1}^2 + \ldots z_{ip}^2$. Hence,

$$\frac{1}{n}\sum_{i=1}^{n}\|\mathbf{z}\|^2 = \sum_{j=1}^{p}\frac{1}{n}\sum_{i=1}^{n}\mathbf{z}_{ij}^2 = \sum_{j=1}^{p}(\text{Variance of } j\text{-th coordinate})$$

How many dimensions to take?

Since R is an isometry, $\|R^T\mathbf{z}\| = \|\mathbf{z}\|$, therefore

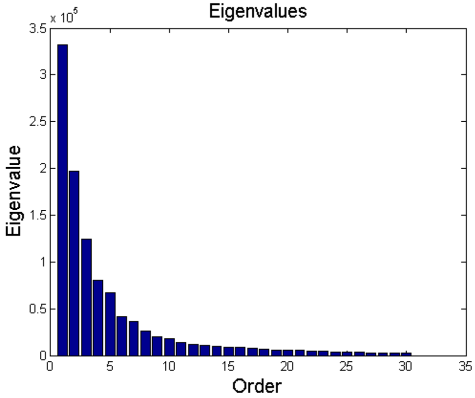
$$\frac{1}{n}\sum_{i=1}^{n}\|\mathbf{z}\|^2 = \sum_{j=1}^{p}(\text{Variance of } j\text{-th PC}) = \lambda_1 + \ldots + \lambda_p$$

What if we take just q PCs? The projected vectors $P_q\mathbf{z}_i$ will satisfy

$$\frac{1}{n}\sum_{i=1}^{n}\|P_q\mathbf{z}_i\|^2 = \lambda_1 + \ldots + \lambda_q$$

Since $\lambda_1 \geq \ldots \geq \lambda_p$ a small k might account for most of the TV.

How many dimensions to take?



How many dimensions to take?

Percentage of explained variance / total variation

$$\frac{\lambda_1 + \ \ldots + \lambda_q}{\lambda_1 + \ \ldots + \lambda_p}$$

How many dimensions to take?

Percentage of explained variance / total variation

$$\frac{\lambda_1 + \ \ldots + \lambda_q}{\lambda_1 + \ \ldots + \lambda_p}$$

Equal to

$$\frac{\sum_{i=1}^{n} \|P_q \mathbf{z}_i\|^2}{\sum_{i=1}^{n} \|\mathbf{z}_i\|^2}$$

Example 1: using PCA to understand plant evolution

Langlade et. al (PNAS, 2005) "Evolution through genetically controlled allometry space":

Example 1: using PCA to understand plant evolution

Langlade et. al (PNAS, 2005) "Evolution through genetically controlled allometry space":

- Broad motivation: understanding of genetics and evolution.

Example 1: using PCA to understand plant evolution

Langlade et. al (PNAS, 2005) "Evolution through genetically controlled allometry space":

- Broad motivation: understanding of genetics and evolution.
- Narrow focus: studying the leaf shape of *Antirrhinum* species and finding specific genes that affect it.

Example 1: using PCA to understand plant evolution

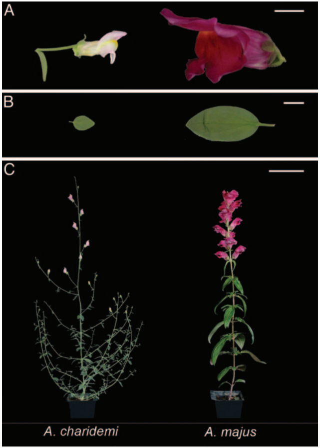


Fig. 1. Comparison between *A. charidemi* (*Left*) and *A. majus* (*Right*). (*A*) Individual flowers in side view. (*B*) Leaves from node 4. (*C*) Whole plants. (Scale bars, 1 cm in *A* and *B* and 10 cm in *C*.)

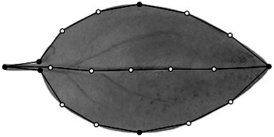
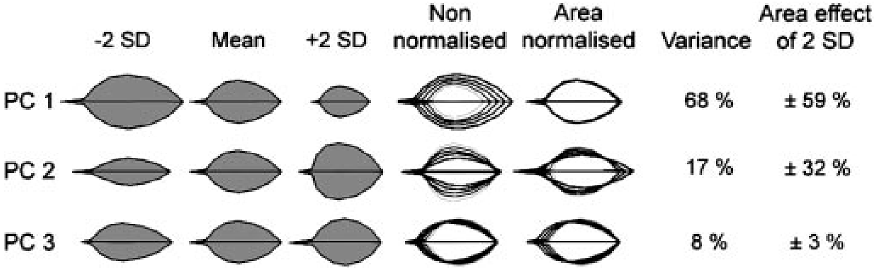


Fig. 2. Points used to capture leaf shape. Primary points (black circles) are placed at key landmarks and secondary points (white circles) are automatically spaced at equal intervals between primary points.

Example 1: using PCA to understand plant evolution



Example 1: using PCA to understand plant evolution

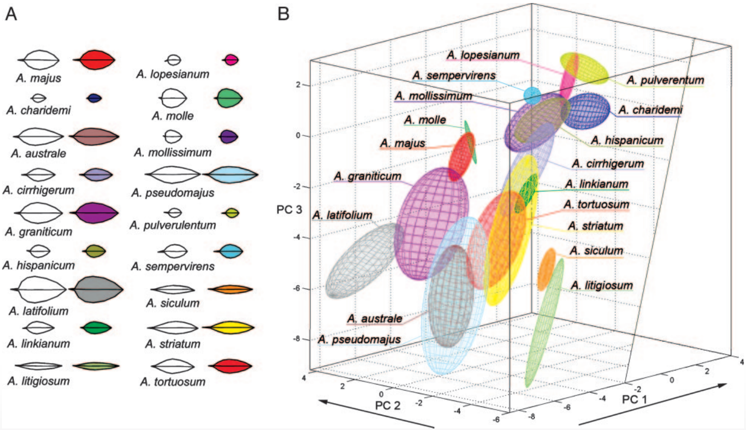


Fig. 5. Size and shape of leaves from 18 *Antirrhinum* species captured by the allometric model. (*A*) Average leaf outlines recorded for each *Antirrhinum* species (white) compared to the outline expressed with the three PCs of the allometric model (colored). (*B*) Representation of each species as a cloud in allometric space based on the F_2 between *A. majus* and *A. charidemi.* Each ellipsoid is based on leaf outlines from 2–14 individuals from each species. The unfilled region to the right corresponds to leaves with negative area and therefore does not represent a realistic part of the space.

Example 2: applying PCA to image patches

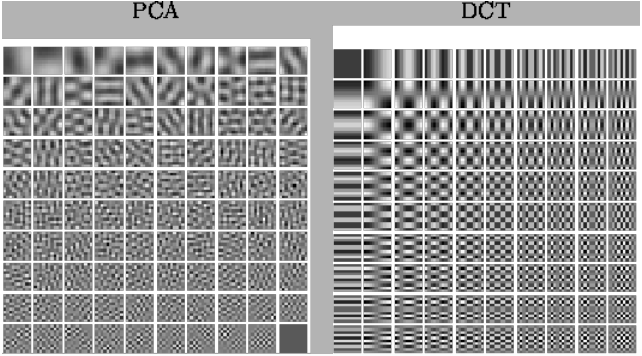


Figure: PCA loading vectors for 10x10 image patches vs. DCT basis

1. Basics of dimensionality reduction
2. Orthogonal projections
3. Principal Component Analysis
4. Examples
5. Alternative definitions of PCA

Alternative definition of PCA as an approximating hyperplane

Given *zero-centered* $\mathbf{z}_1, \ldots, \mathbf{z}_n \in \mathbb{R}^p$ find a q-dimensional subspace H that minimizes the sum of squared residuals

$$\min \sum_{i=1}^{n} \|\mathbf{z}_i - P_H \mathbf{z}_i\|^2$$

Equivalence to the variance-maximizing definition

$$
\begin{aligned}
\sum_{i=1}^{n} \|\mathbf{z}_i - P_H \mathbf{z}_i\|^2 &= \sum_{i=1}^{n} \sum_{j=q+1}^{p} \left(\mathbf{b}_j^T \mathbf{z}_i\right)^2 \\
&= \sum_{i=1}^{n} \sum_{j=q+1}^{p} \mathbf{b}_j^T \mathbf{z}_i \mathbf{z}_i^T \mathbf{b}_j \\
&= \sum_{j=q+1}^{p} \mathbf{b}_j^T \underbrace{\left(\sum_{i=1}^{n} \mathbf{x}_i \mathbf{x}_i^T \right)}_{\text{sample covariance!}} \mathbf{b}_j
\end{aligned}
$$

Equivalence to the variance-maximizing definition

Israeltech challenge

Q: Which set of orthogonal vectors $\mathbf{b}_{q+1}, \ldots, \mathbf{b}_p$ minimizes

$$\sum_{j=q+1}^{p} \mathbf{b}_j^T \Sigma \mathbf{b}_j$$

Equivalence to the variance-maximizing definition

Q: Which set of orthogonal vectors $\mathbf{b}_{q+1}, \ldots, \mathbf{b}_p$ minimizes

$$\sum_{j=q+1}^{p} \mathbf{b}_j^T \Sigma \mathbf{b}_j$$

A: The last $p - q$ coefficient vectors found by PCA $\mathbf{u}_{q+1}, \ldots, \mathbf{u}_p$.

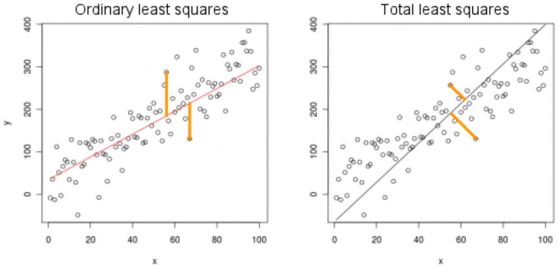
Equivalence to the variance-maximizing definition

Q: Which set of orthogonal vectors $\mathbf{b}_{q+1}, \ldots, \mathbf{b}_p$ minimizes

$$\sum_{j=q+1}^{p} \mathbf{b}_j^T \Sigma \mathbf{b}_j$$

A: The last $p - q$ coefficient vectors found by PCA $\mathbf{u}_{q+1}, \ldots, \mathbf{u}_p$. Therefore $H = Sp\{\mathbf{b}_1, \ldots, \mathbf{b}_q\}$.

PCA vs. linear regression



PCA: 3rd definition

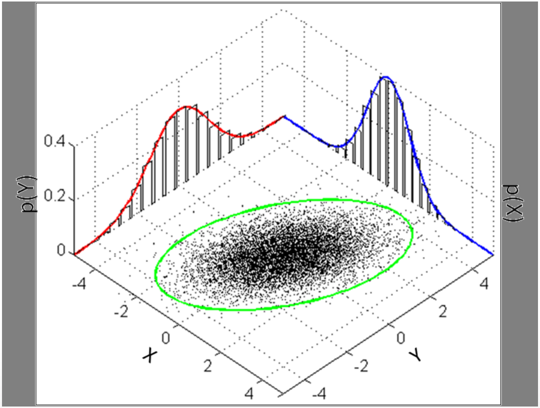
Let $\mathbf{z}_1, \ldots, \mathbf{z}_n \sim \mathcal{N}(0, \Sigma)$ be gaussian multivariate samples.
Recall that the MLE of Σ is just the sample covariance

$$\hat{\Sigma}_{\mathsf{MLE}} = \frac{1}{n} \sum_i \mathbf{z}_i \mathbf{z}_i^T$$

The PCA algorithm simply diagonalizes $\hat{\Sigma}_{\mathsf{MLE}}$:

- $\mathbf{u}_1, \ldots, \mathbf{u}_p$ are the *principal axes* of the multivariate gaussian.
- $\lambda_1, \ldots, \lambda_p$ give the lengths of the principal axes.

PCA: 3rd definition




QUESTIONS

Figure credits:

- Wine biplot copied from stats.stackexchange.com/a/7862.
- Squirrel picture by wikimedia user "Ray eye", licensed under the Creative Commons Attribution-Share Alike 2.0 Germany license.
- Orthogonal vs. nonorthogonal projections illustration by Jitse Niesen.
- 2D PCA illustrations on point clouds and eigenvalue decay by Sujin Jang.
- DCT vs. PCA of image patches taken from Tapani Raiko's Masters' Thesis.
- Gaussian multivariate distribution by Bscan, under Creative Commons CC0 1.0 Universal Public Domain Dedication.