Pattern Recognition

Lecture 2. Review of Linear Algebra and Probability

Dr. Shanshan ZHAO

shanshan.zhao@xjtlu.edu.cn

School of AI and Advanced Computing

Xi'an Jiaotong-Liverpool University

Academic Year 2021-2022

Table of Contents

What is a Linear Algebra?

Vector

A vector $x \in R_n$ is a list of n numbers, usually written as **column vector**.

$$\begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} \tag{1}$$

The **unit vector** e_i is a vector of all 0's, except entry i, which has value 1: $e_i = (0, ...0, 1, 0, ...0)$, This is also called **one-hot vector**.

Properties

$$\begin{aligned} \mathbf{x} = (x_1, x_2) \quad \mathbf{y} &= (y_1, y_2) \\ \mathbf{x} + \mathbf{y} &= (x_1, x_2) + (y_1, y_2) = (x_1 + y_1, x_2 + y_2) \\ \mathbf{x} - \mathbf{y} &= (x_1, x_2) - (y_1, y_2) = (x_1 - y_1, x_2 - y_2) \\ a\mathbf{x} &= a(x_1, x_2) = (ax_1, ax_2) \end{aligned}$$

Vector

create a vector with python

```python
import numpy as np
# method with list
v1 = [1,2,3]
v1 = np.array([v1]).T
print(v1)
# method with numpy
v2 = np.array([[1],[2],[3]])
print(v2)
v = v1+v2
print(v)
print(3*v)
```

code/vector1.py

Matrices

A matrix $A \in \mathbb{R}^{m \times n}$ with m rows and n columns is a 2d array of numbers, arranged as follows:

$$\begin{bmatrix} a_{11} & a_{12} & \cdots & a_{12} \\ a_{21} & a_{22} & \cdots & a_{12} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix}$$

If $m = n$, the matrix is said to be **square**.

(a) Lena grayscale (b) Lena standard image

Basic operations-Create a matrix

2. create a matrix: $v = \begin{bmatrix} 1 & 4 & 7 \\ 2 & 5 & 8 \\ 3 & 6 & 9 \end{bmatrix}$

```
# with the method of list
# TODO
M
print(M)
```

```
# with the method of array
# TODO
M =
print(M)
```

Basic operations-Create a matrix

```
# with the method of list
# TODO
M = [[1,4,7],[2,5,8],[3,6,9]]
M = np.array(M)
print(M)
```

```
# with the method of array
# TODO
M = np.array([[1,4,7],[2,5,8],[3,6,9]])
print(M)
```

Basic operations-matrix indexing

Task: get the result by indexing M:

$$v = \begin{bmatrix} 4 & 7 \\ 5 & 8 \end{bmatrix}$$

```
# TODO
```

Basic operations-matrix indexing

Task: get the result by indexing M:

$$v = \begin{bmatrix} 4 & 7 \\ 5 & 8 \end{bmatrix}$$

```
# TODO
print(M[0:2,1:3])
```

```
[[4 7]
 [5 8]]
```

Basic operations-other ways to create matrices

Many other functions for creating matrices/vectors provided Numpy

```
a = np.zeros((2,3)) # create an array with all zeros
print(a)
```

```
[[0. 0. 0.]
 [0. 0. 0.]]
```

```
b = np.ones((1,2)) # create an array with all ones
print(b)
```

```
c = np.full((2,4), 7) # create an array with constant values
print(c)
```

```
[[7 7 7 7]
 [7 7 7 7]]
```

```
d = np.eye(3) # create a 3 by 3 identity matrix
print(d)
```

```
[[1. 0. 0.]
 [0. 1. 0.]
 [0. 0. 1.]]
```

Vector-Vector Products

Given two vectors $x, y \in \mathcal{R}^n$, the quantity $x^T y$, sometimes called the ***inner product*** or ***dot product*** of the vectors, is a real number given by

$$x^T y \in \mathcal{R} = \begin{bmatrix} x_1, x_2, \cdots, x_n \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix} = \sum_{i=1}^{n} x_i y_i$$

Note that it is always the case that $x^T y = y^T x$
Given vectors $x \in \mathcal{R}^m$,$y \in \mathcal{R}^n$, $xy^T \in \mathcal{R}^{m \times n}$ is called the ***outer product*** or ***cross product***. It is a matrix :

$$xy^T \in \mathcal{R}^{m \times n} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} \begin{bmatrix} y_1, y_2, \cdots, y_n \end{bmatrix} = \begin{bmatrix} x_1 y_1 & x_1 y_2 & \cdots & x_1 y_n \\ x_2 y_1 & x_2 y_2 & \cdots & x_2 y_n \\ \vdots & \vdots & \ddots & \vdots \\ x_m y_1 & x_m y_2 & \cdots & x_m y_n \end{bmatrix}$$

Matrix-Vector Products

Given a matrix $A \in \mathcal{R}^n$ and a vector $x \in \mathcal{R}^n$, their product is a vector $y = Ax \in \mathcal{R}^m$. There are a couple ways of looking at matrix-vector multiplication.

If we write A by rows,then we can express Ax as,

$$y = Ax = \begin{bmatrix} - & a_1^T & - \\ - & a_1^T & - \\ & \vdots & \\ - & a_1^T & - \end{bmatrix} x = \begin{bmatrix} a_1^T x \\ a_2^T x \\ \vdots \\ a_m^T x \end{bmatrix}$$

If we write A by columns,then we can express Ax as,

$$y = Ax = \begin{bmatrix} | & | & & | \\ a_1 & a_2 & \cdots & a_n \\ | & | & & | \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} = \begin{bmatrix} a_1 \end{bmatrix} x_1 + \begin{bmatrix} a_2 \end{bmatrix} x_2 + \cdots + \begin{bmatrix} a_n \end{bmatrix} x_n$$

y is a ***linear combination*** of the columns of A, where the coefficients of the linear combination are given by the entries of x.

Matrix-Matrix Products

1. We can view matrix-matrix multiplication as a set of vector-vector products. The most obvious viewpoint, which follows immediately from the definition, is that the (i, j)th entry of C is equal to the inner product of the ith row of A and the jth column of B.

$$C = AB = \begin{bmatrix} - & a_1^T & - \\ - & a_1^T & - \\ & \vdots & \\ - & a_1^T & - \end{bmatrix} \begin{bmatrix} | & | & & | \\ b_1 & b_2 & \cdots & b_n \\ | & | & & | \end{bmatrix}$$

$$= \begin{bmatrix} a_1^T b_1 & a_1^T b_2 & \cdots & a_1^T b_n \\ a_2^T b_1 & a_2^T b_2 & \cdots & a_2^T b_n \\ \vdots & \vdots & \ddots & \vdots \\ a_m^T b_1 & a_m^T b_2 & \cdots & a_m^T b_n \end{bmatrix}$$

Matrix-Matrix Products

2. We can represent A by columns, and B by rows. This representation leads to a much trickier interpretation of AB as a sum of outer products.

$$C = AB = \begin{bmatrix} | & | & & | \\ a_1 & a_2 & \cdots & a_n \\ | & | & & | \end{bmatrix} \begin{bmatrix} - & b_1^T & - \\ - & b_1^T & - \\ & \vdots & \\ - & b_1^T & - \end{bmatrix} = \sum_{i=1}^{n} a_i b_i^T$$

Basic operations: matrix multiplications

practise

2.4 Multiplications

2.4.1 Dot Multiplication

2.4.2 Element-wise Multiplication

2.4.3 Inner(dot) or Outer(cross) product

2.4.4 Matrix product of two arrays

2.4 Multiplications

2.4.1 Dot Multiplication

```
[ ]: np.dot(3, 4)
```

$$M = \begin{bmatrix} 1 & 4 & 7 \\ 2 & 5 & 8 \\ 3 & 6 & 9 \end{bmatrix} v = \begin{bmatrix} 2 \\ 4 \\ 6 \end{bmatrix}$$

Figure: Guess before running the cell

Properties of Matrices

- Matrix multiplication is associative: (AB)C = A(BC).
- Matrix multiplication is distributive: A(B+C)=AB+AC.
- Matrix multiplication is *not* commutative: $AB \neq BA$.
- Identity matrix, denoted $I \in \mathcal{R}^{n \times n}$:$AI = A = IA$
- $(A^T)^T = A$
- $(AB)^T = B^T A^T$
- $(A + B)^T = A^T + B^T$

Determinant

Useful value computed from the elements of a square matrix A

$$det\left[a_{11}\right] = a_{11}$$

$$det\begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} = a_{11}a_{22} - a_{12}a_{21}$$

$$det\begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} = a_{11}a_{22}a_{33} + a_{12}a_{23}a_{31} + a_{13}a_{21}a_{32}$$

$$- a_{13}a_{22}a_{31} - a_{23}a_{32}a_{11} - a_{33}a_{12}a_{21}$$

further reading:
`https://mathinsight.org/determinant_linear_transformation`

Matrix Inverse

Does not exist for all matrices, necessary (but not sufficient) that the matrix is square

$$AA^{-1} = A^{-1}A = I$$

And,

$$A^{-1} = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix}^{-1} = \frac{1}{detA} \begin{bmatrix} a_{22} & -a_{12} \\ -a_{21} & a_{11} \end{bmatrix}, detA \neq 0$$

If $detA = 0$, A does not have an inverse.

2.5.3 Matrix Inverse

```
from numpy.linalg import inv

a = np.array([[1., 2.], [3., 4.]])
ainv = inv(a)
```

```
print('a:',a)
print('inv:',ainv)
```

Eigenvectors and Eigenvalues

An eigenvalue λ and eigenvector x satisfies

$$Mx = \lambda x$$

Where A is a square matrix.
for scalar λ, which can be rewritten $(M - \lambda I)x = 0$ Which has a soluton if and only if $det(M - \lambda I) = 0$

- The eigenvalues are the roots of this determinant which is polynomial in λ.
- Substitute the resulting eigenvalues back into $Mx = \lambda x$ and solve to obtain the corresponding eigenvetor.

2.5.4 Eigenvalues and Eigenvectors

```
:
from numpy import linalg as LA
w, v = LA.eig(np.diag((1, 2, 3)))
```

Basic operations: properties

practise

2.5 Properties

2.5.1 Transpose

2.5.2 Matrix Determinant

2.5.3 Matrix Inverse

2.5.4 Eigenvalues and Eigenvectors

2.5 Properties

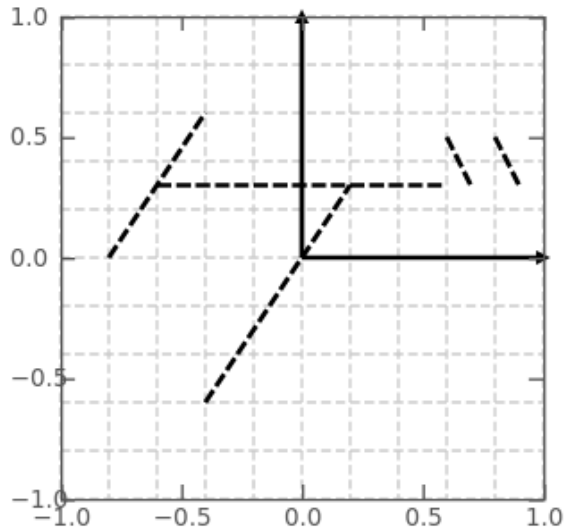
2.5.1 Transpose

```
[ ]: print(M)
     print(M.T)
```

```
[ ]: print(v)
```

Linear Algebra by Animies

`https://lihs.me/matrix-animation/`



$M = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$

Preset: Custom

auto play

Play Reset SVG

Powered By D3.js
Made By lihs.me
Source Code On GitHub

Linear Algebra by Animies 1

$$A = \begin{bmatrix} 1.5 & 0 \\ 0 & 0.5 \end{bmatrix}$$

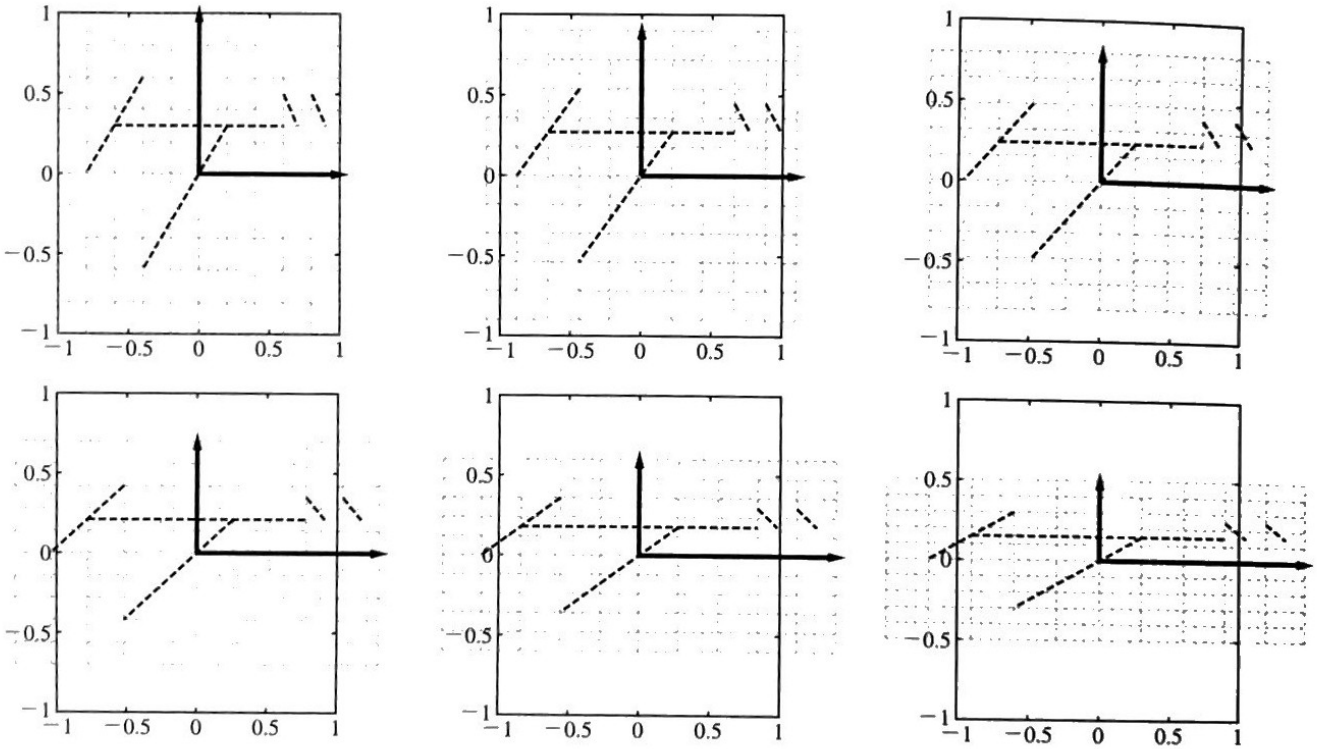


Figure: Preset:s0

Linear Algebra by Animies 2

$$A = \begin{bmatrix} 0 & 0 \\ 0 & 0.5 \end{bmatrix}$$

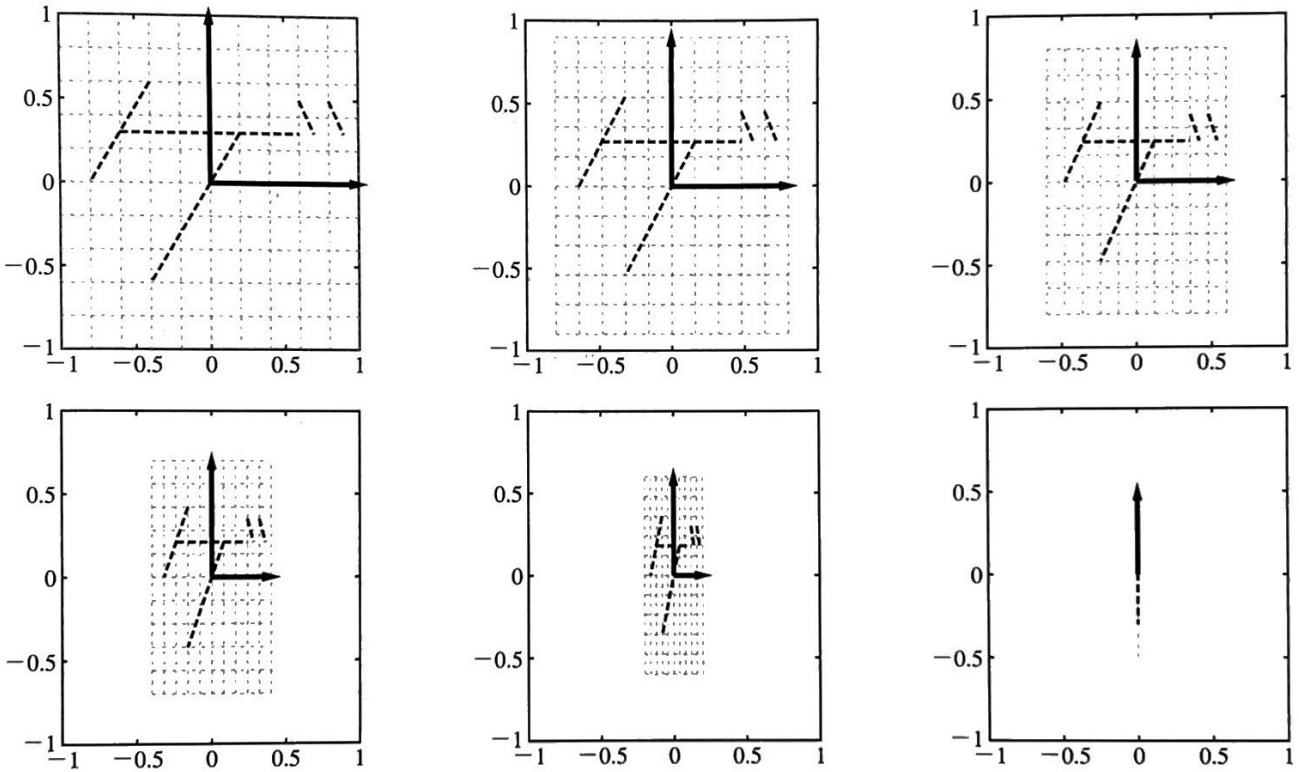


Figure: Preset:s1

Linear Algebra by Animies 3

$$A = \begin{bmatrix} 1.5 & 0 \\ 0 & -0.5 \end{bmatrix}$$

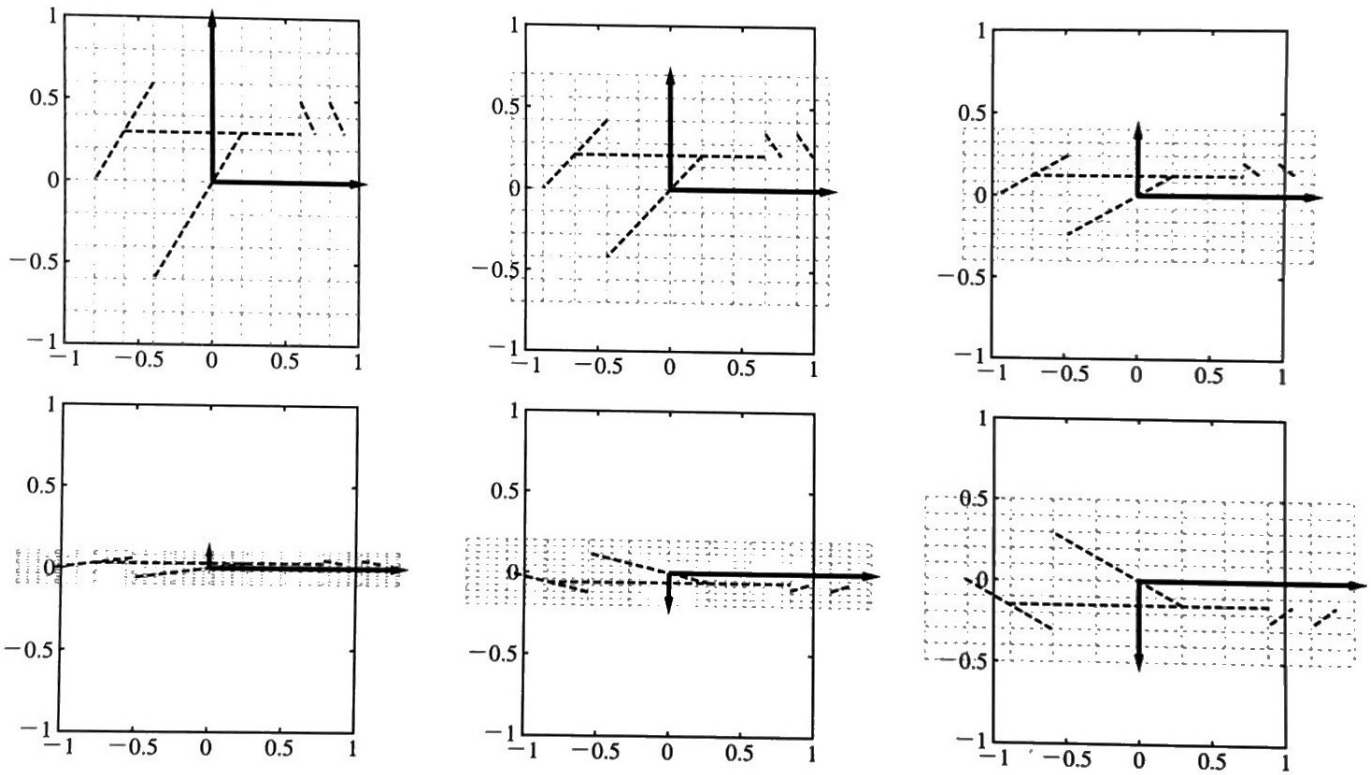


Figure: Preset:s2

Linear Algebra by Animies 4

$$A = \begin{bmatrix} 1 & -0.3 \\ -0.7 & 0.6 \end{bmatrix}$$

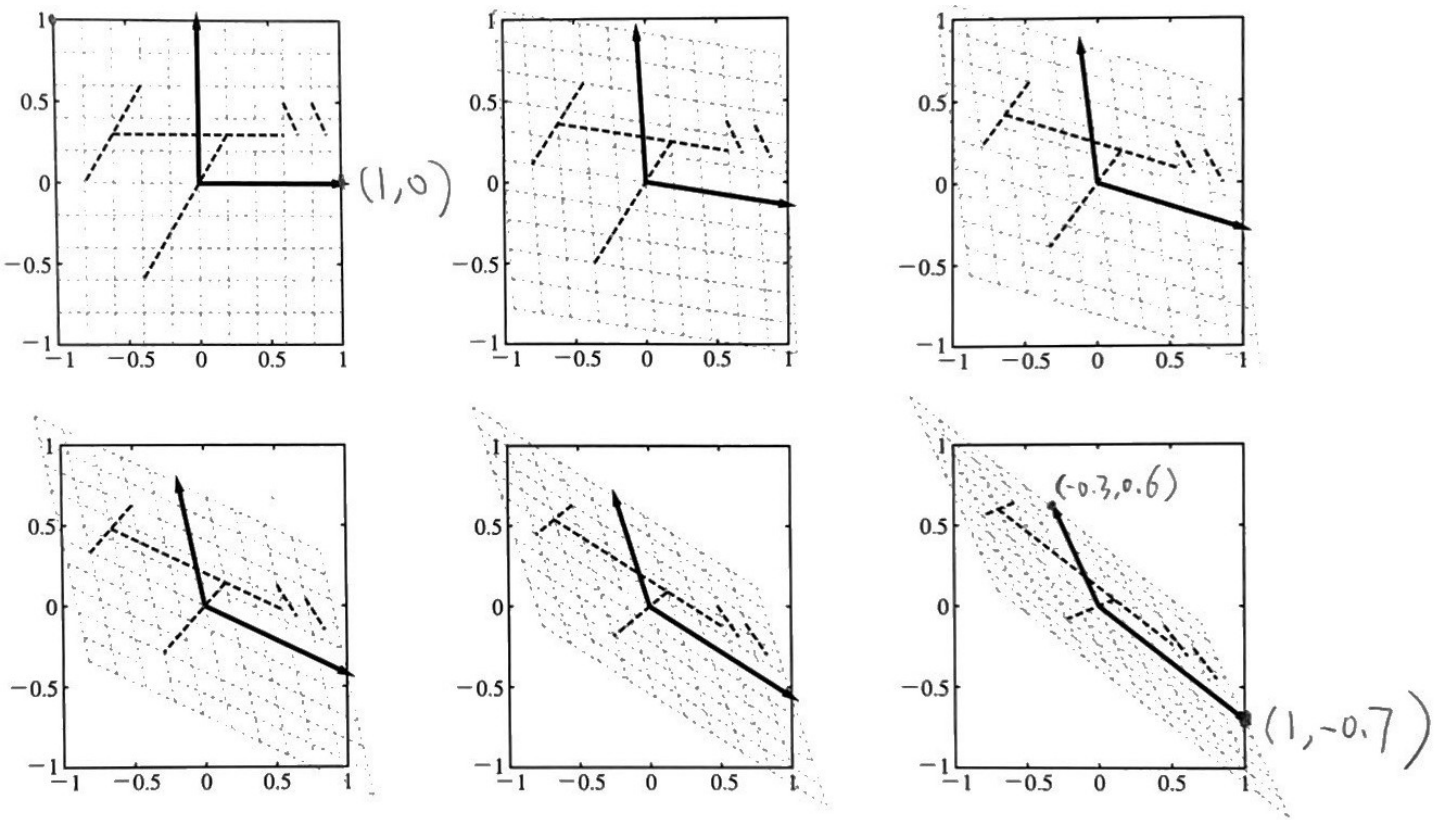


Figure: Preset:s3

Linear Algebra by Animies 5

$$A = \begin{bmatrix} 1 & -0.3 \\ -0.7 & 0.6 \end{bmatrix}$$

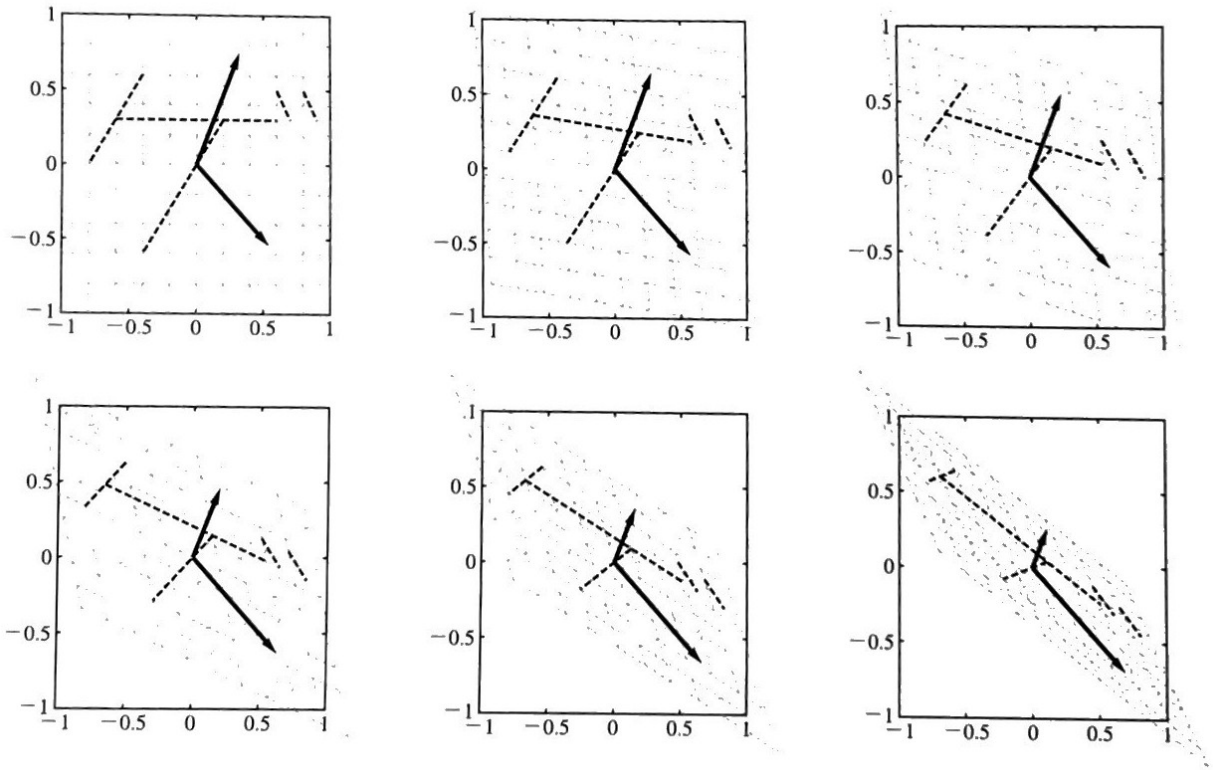


Figure: Preset:s4

Linear Algebra by Animies 6

$$A = \begin{bmatrix} 1 & -0.3 \\ -0.7 & 0.6 \end{bmatrix}$$

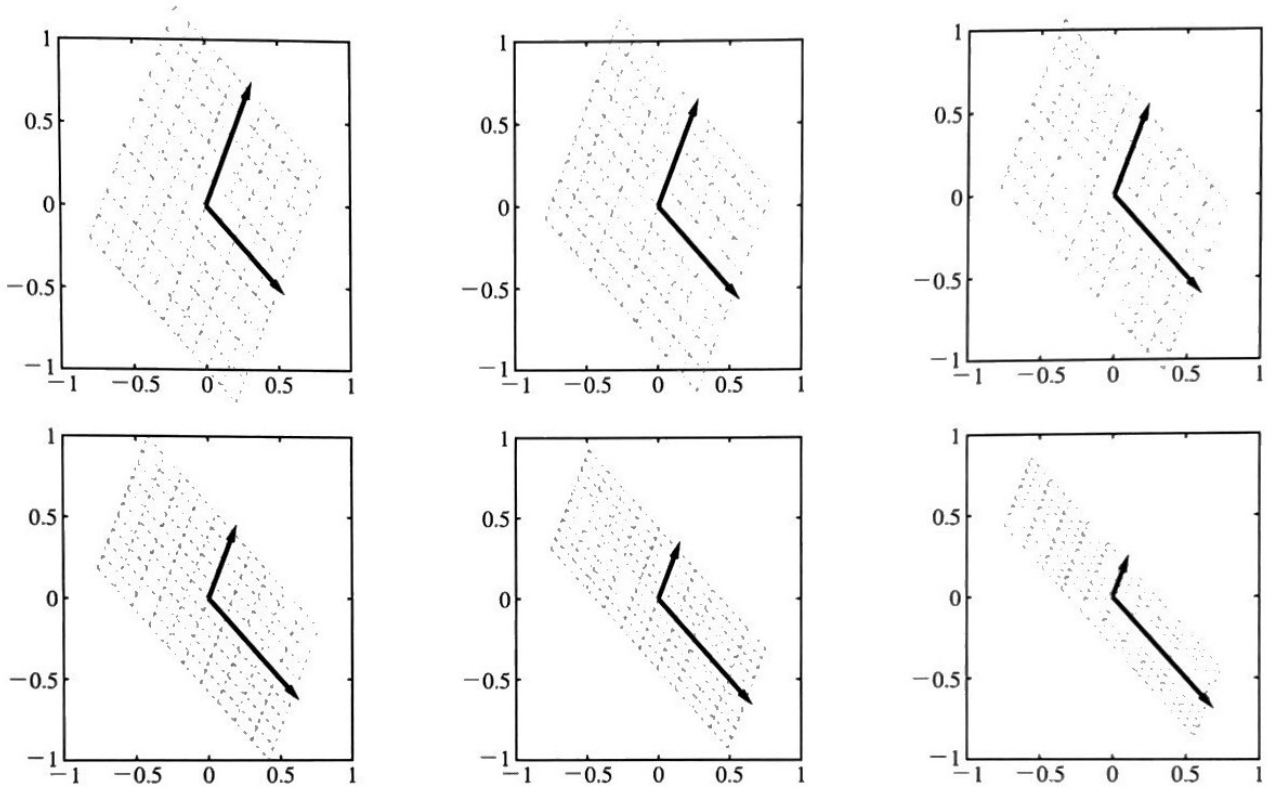


Figure: Preset:s5

Linear Algebra by Animies 7

$$A = \begin{bmatrix} 0.8 & -0.6 \\ 0.4 & -0.3 \end{bmatrix}$$

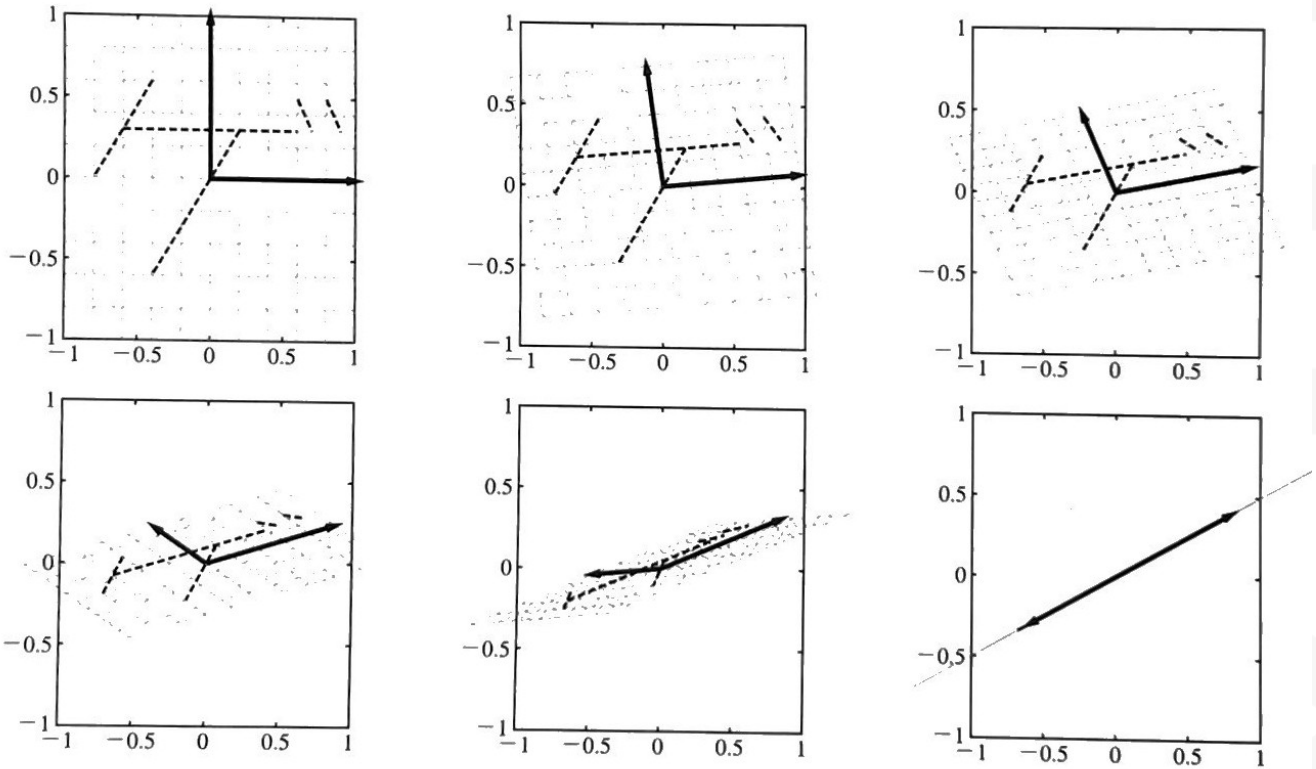


Figure: Preset:s6

Linear Algebra by Animies 8

$$A = \begin{bmatrix} 0.8 & -0.6 \\ 0.4 & -0.3 \end{bmatrix}$$

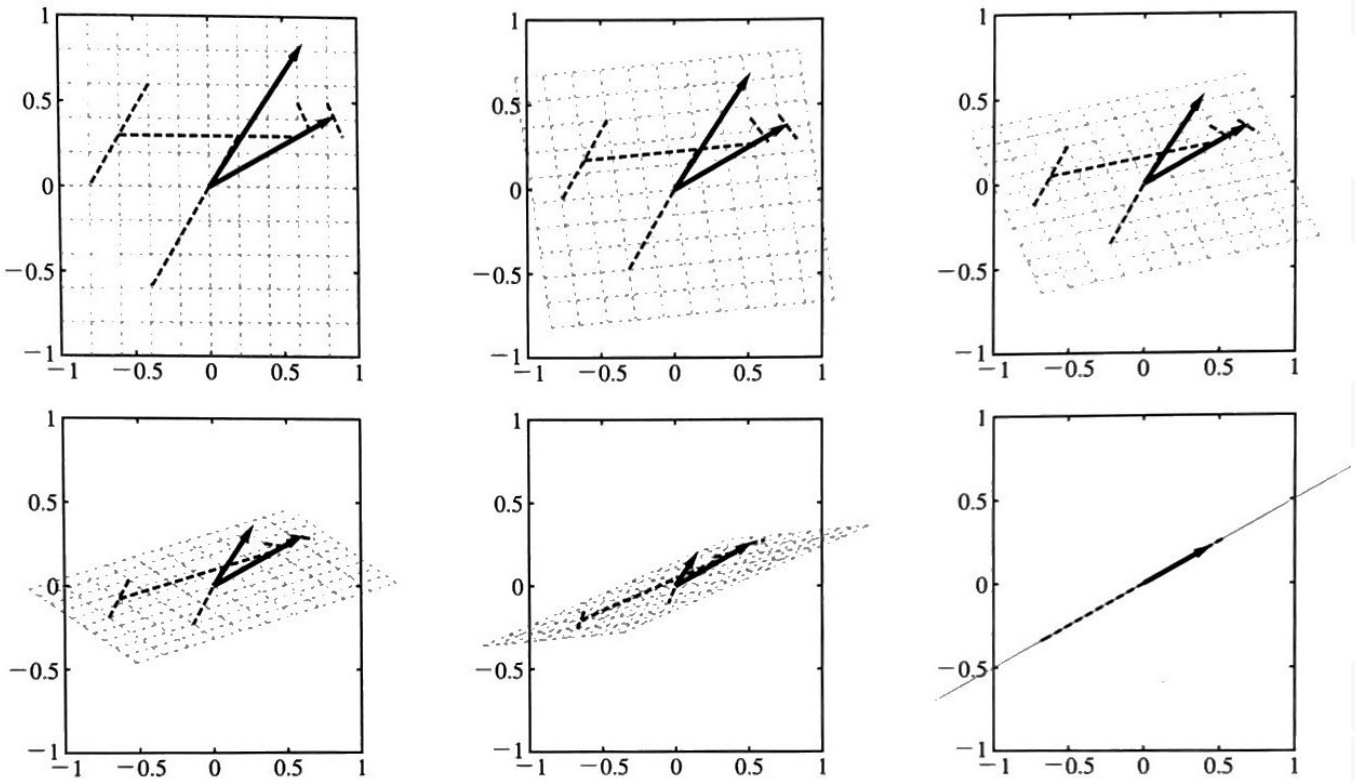


Figure: Preset:s7

Linear Algebra by Animies 9

$$A = \begin{bmatrix} 0.8 & -0.6 \\ 0.4 & -0.3 \end{bmatrix}$$

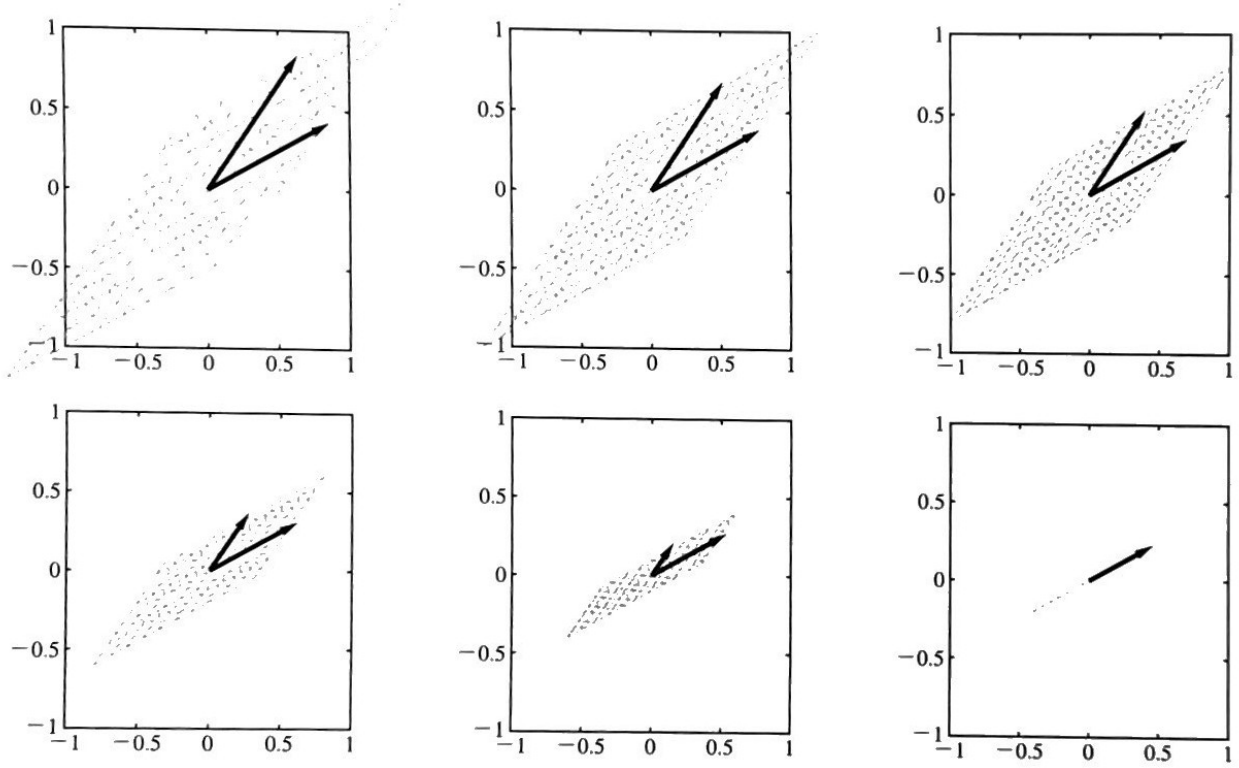


Figure: Preset:s8

Review of Probability theory

Random Variables

- **discrete variable:** takes either finite or countably infinite values.
- **continuous variable:** takes infinite number of possible values.

Cumulative distibution function(CDF)

Consider waiting for a bus, which runs every 30 minutes. We shall make the idealistic assumption that the buses are always exactly on time, thus a bus arrives every 30 minutes. If you are waiting for a bus, but don't know the timetable, then the precise length of time you need to wait is unknown. Let the continuous random variable X denote the length of time you need to wait for a bus.
We can write this as:

$$P(X < 0) = 0$$
$$P(0 \leq X \leq 30) = 1$$
$$P(X > 30) = 0$$

To obtain the probability of falling in an interval we can do the following:

$$P(a < x < b) = P(x \leq b) - P(x \leq a)$$
$$= F(b) - F(a)$$

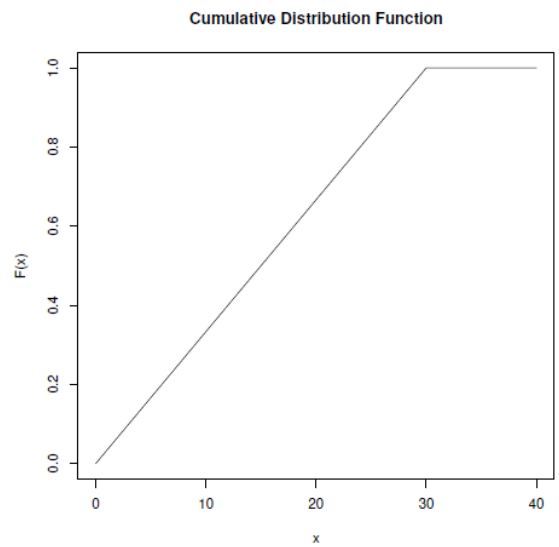


Figure: Cumulative distribution function of random variable X in the 'bus' example.

Probability density function(PDF)

Although we cannot define a probability distribution function for a continuous random variable, we can define a closely related function, called the *probability density function(pdf)*,$p(x)$:

$$p(x) = \frac{d}{dx}F(x) = F'(x)$$

$$F(x) = \int_{-\infty}^{x} p(x)dx$$

The pdf is the gradient of the cdf.

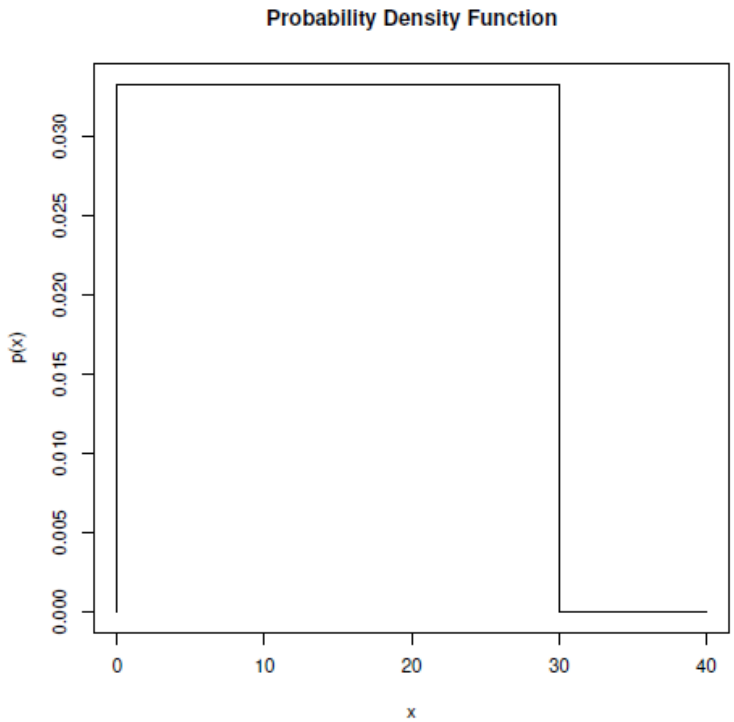


Figure: Probability density function of random variable X in the 'bus' example.

The Gaussian distribution

The normal distribution, also called the Gaussian distribution, is a probability distribution commonly used to model phenomena such as physical characteristics (e.g. height, weight, etc.) and test scores.
Why always Gaussian?

- it fits many natural phenomena
- the normal distribution exhibits a number of nice simplifying characteristics
- central limit theorem: the distribution of sample means approximates a normal distribution as the sample size gets larger, regardless of the population's distribution

Question:
Which of the following is/are true of normal distribution?
A. They are always symmetric
B. They are never fat-tailed
C. They always have a mean of 0

The Gaussian distribution

The *Gaussian(or Normal)* distribution is the most commonly encountered (and easily analysed) continuous distribution.

$$p(x|\mu,\sigma^2) = N(x;\mu,\sigma) = \frac{1}{\sqrt{2\pi\sigma^2}} exp(\frac{-(x-\mu)^2}{2\sigma^2})$$

The Gaussian is described by two parameters: the mean μ(location), and the variance σ^2(dispersion).

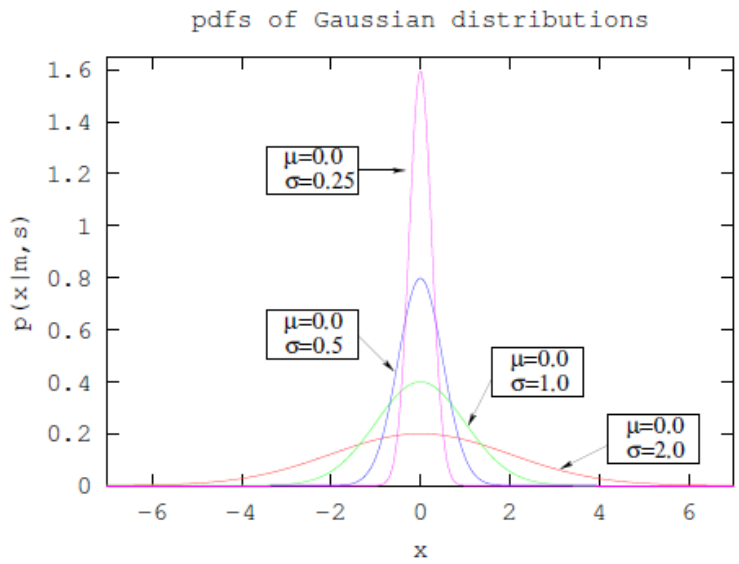


Figure: Four Gaussian pdfs with zero mean and different standard deviations.

The multivariate Gaussian distribution and covariance

The univariate(one-dimensioanl) Gaussian may be extended to the multivariate(multi-dimensional) case. The *D*-dimensional Gaussian is parameterised by a mean vector, $\mu = (\mu_1, ..., \mu_D)^T$, and a *covariance matrix* $\Sigma = (\sigma_{ij})$ (Note that Σ is a *D* by *D* dquare matrix, and σ_{ij} or Σ_{ij} denotes its element at i'th row and j'th column.) The probability density is

$$p(x|\mu, \Sigma) = \frac{1}{(2\pi)^{D/2}|\Sigma|^{1/2}} exp(-\frac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu))$$

The multivariate Gaussian distribution and covariance

multivariate Gaussian distribution pdf:

$$p(x|\mu,\Sigma) = \frac{1}{(2\pi)^{D/2}|\Sigma|^{1/2}} exp(-\frac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu))$$

Exercise:
Consider a 2-dimensional Gaussian distribution with a mean vector $\mu = (0,0)^T$ and a diagonal covariance matrix,i.e.$\Sigma = \begin{bmatrix} \sigma_{11} & 0 \\ 0 & \sigma_{22} \end{bmatrix}$, show that its pdf can be simplified to the product of two pdfs, each of which corresponds to a one-dimensional Gaussian distribution.

$$p(x|\mu,\Sigma) = p(x_1|\mu_1,\sigma_{11})p(x_2|\mu_2,\sigma_{22})$$

Thank You !