International Masters Computer Vision & Vibot & MaIA

Module: Applied Mathematics

Désiré Sidibé

Associate Professor
Université de Bourgogne-Franche Comté
Laboratoire LE2I, UMR CNRS 6306
12, rue la Fonderie - 71200 Le Creusot, France
ddsidibe@u-bourgogne.fr

Contents

Introduction

These are the lectures notes of the *Applied Maths* module that I started teaching in the Masters in Computer Vision in October 2009.

The primarily goal of the module is to provide students with most of the mathematical tools that are useful for a computer vision scientist. Since mathematics are a very broad subject, we will focus on four important topics. The document is accordingly divided in four chapters: linear algebra, probability and statistics, optimization, and state estimation. Though the subject of last chapter is closely related to that of the second one, we believe it deserves a full separate chapter. Moreover, we assume that the reader already has basic training in these topics.

Chapter 1 provides a brief review of some of the important results in linear algebra. For a more complete introduction on this fantastic subject, we recommend excellent textbooks such as [Str09, Lay99]. Professor Strang's book [Str09] is definitely the best textbook on the subject of linear algebra.

The main problem in linear algebra, and also one of the most important in applied maths, is solving a *linear system of equations* $A\mathbf{x} = b$ (many engineering problems can be reduced to solving linear systems). There are different ways of obtaining the solution vector $\mathbf{x}$:

- Direct solution to find $\mathbf{x}$ by elimination and back substitution.
- Matrix solution using the inverse of A: $\mathbf{x} = A^{-1}b$ (if A has an inverse).
- Vector space solution $\mathbf{x} = \mathbf{y} + \mathbf{z}$; i.e. particular solution (to $A\mathbf{y} = b$) plus nullspace solution (to $A\mathbf{z} = 0$).
- Linear least squares solution $\mathbf{x} = (A^T A)^{-1} A^T b$.

In the lectures, we are going to learn the ideas and principles for solving linear systems.

Chapter 2 provides a brief review of important definitions and results in probabilty and statistics. For a more complete introduction on this subject, please refer to [BT02, GS97].

Probability is the study of random variables or random processes and is a topic of great importance because randomness is everwhere! For instance, every sensor reading is a random variable (e.g., thermal noise, voltage noise, etc.) and there is some degre of uncertainty in every system.

Understanding how to model uncertainty and how to analize its effects is an essential part of every engineering system. The two key concepts are:

- How to model uncertainty? What do we mean by a 'random experiment'?
- How to process observations to reduce uncertainty?

Chapter 3 offers an overview of the basic principles of optimization, a key topic in applied maths. The optimization problem can be simply stated as follows:

- Given a function $f : \mathbb{R}^n \to \mathbb{R}$, find those $\mathbf{x} \in \mathbb{R}^n$ where f takes on a maximum or minimum value, and evaluate f at those points.

As maximizing f is just equivalent to minimizing $-f$, we will only consider minimization problems and, we will study gradient based methods such as gradient descent, Newton's methods, and conjugate gradients.

In chapter 4, we will consider the problem of filtering and state estimation which is a widely important topic in engineering. Basically, a *filter* is a procedure that estimates parameters of a system. These parameters are called system's *state variables* and formed the *state vector* of the system. The state vector $\mathbf{x}$ is defined as the minimal set of parameters that is sufficient to completely describe the unforced dynamical behaviour of the system.

Typically, the state vector $\mathbf{x}$ is unknown or cannot be measured. So we have to estimate it from noisy measures $\mathbf{y}$ called *observations*.

Suppose we are getting the observations through time, $\mathbf{y}(k), k = 1, 2, \ldots$. We can distinguish three estimation problems:

- **Filtering**: estimate the *current* state $\mathbf{x}(k)$ given all the data availabe up to and including $\mathbf{y}(k)$.
- **Smoothing**: estimate some *past* value of $\mathbf{x}(l)$, $l < k$, given all the data availabe up to and including $\mathbf{y}(k)$.
- **Prediction**: estimate some *future* value of $\mathbf{x}(l)$, $l > k$, given all the data availabe up to and including $\mathbf{y}(k)$.

Finally, the reader should know that these notes do not replace a textbook and they **definitely do not replace class lectures**! Many examples and practical aspects will be discussed during class lectures. These are just notes you can refer to, to quickly find a definition or a property.

If you find any mistake in this document, please fell free to send me a message. Besides, any comment or suggestion will be appreciated.

D. Sidibé, September 2017.

Chapter 1

Review of Linear Algebra

This chapter reviews the main ideas of Linear Algebra, a "fantastic, clean and beautiful subject" as stated by Gilber Strang. The primarily sources of this chapter are Strang's book [Str09], the appendix *Basic facts from Linear Algebra* in [MSKS06] and the author's own notes.

1.1 Introduction

The main problem in linear algebra, and also one of the most important in applied maths, is solving a *linear system of equations* $A\mathbf{x} = b$ (many engineering problems can be reduced to solving linear systems). There are different ways of obtaining the solution vector $\mathbf{x}$ and the goal of this chapter is to introduced some of those ideas and methods. The main questions to answer are:

- when does a solution exist?
- how many solution are there?
- does a solution exist for every vector b?

We will first learn to answer those three questions and the key idea is based on considering matrix operations as taking *linear combination of vectors*. The main result here is that

When we take all linear combinations of the column vectors of a matrix A, we get the column space of A. And if this space includes the vector b, then we can solve the equation $A\mathbf{x} = b$. [Str09]

Once we know a solution or many solutions exist, we have to find them. Finding the solution vector b is based on four main operations which lead to matrix factorization:

- Elimination which leads to LU decomposition $A = LU$
- Orthogonalization which leads to QR decomposition $A = QR$
- Eigenvalues and eigenvectors which lead to $A = S\Lambda S^{-1}$
- Singular values decomposition which gives $A = U\Sigma V^T$

We will also review the linear least squares (LLS) method and principal component analysis (PCA) which are two main practical applications of linear algebra.

1.2 Linear spaces

In these notes, we deal mostly with finite-dimensional *linear spaces*, which are also often called vector spaces. For generality, we consider the linear space to be n-dimensional.

Definition 1.2.1. (**A linear space or a vector space**). *A set of vectors V is considered a* linear space *over the field $\mathbb{R}$ if for any two vectors $v_1, v_2 \in V$ and any two scalars $\alpha, \beta \in \mathbb{R}$, the linear combination $v = \alpha v_1 + \beta v_2$ is also a vector in V.*

Furthermore, we have the following properties:

- The addition is commutative and associative; it has an identity 0 and each element has an inverse $-v$, such that $v + (-v) = 0$.
- The scalar multiplication respects the structure of $\mathbb{R}$; i.e. $\alpha(\beta)v = (\alpha\beta)v$, $1v = v$ and $0v = 0$.
- The addition and scalar multiplication are related by the distributive laws: $(\alpha + \beta)v = \alpha v + \beta v$ and $\alpha(v + u) = \alpha v + \alpha u$.

For example, $\mathbb{R}^n$ is a linear space over the field of real numbers $\mathbb{R}$. We use a column to represent a vector:

$$[x_1, x_2, \ldots, x_n]^T = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix},$$

where $[x_1, x_2, \ldots, x_n]^T$ means "the row vector $[x_1, x_2, \ldots, x_n]$ transposed".

Given two scalars $\alpha, \beta \in \mathbb{R}$ and two vectors $v_1 = [x_1, x_2, \ldots, x_n]^T \in \mathbb{R}^n$ and $v_2 = [y_1, y_2, \ldots, y_n]^T \in \mathbb{R}^n$, their **linear combination** is a componentwise summation weighted by α and β:

$$\begin{aligned} \alpha v_1 + \beta v_2 &= \alpha[x_1, x_2, \ldots, x_n]^T + \beta[y_1, y_2, \ldots, y_n]^T \\ &= [\alpha x_1 + \beta y_1, \alpha x_2 + \beta y_2, \ldots, \alpha x_n + \beta y_n]^T. \end{aligned}$$

1.2.1 Linear independence and basis

Definition 1.2.2. (**Subspace**). *A subset W of a linear space V is called a* subspace *if the zero vector 0 is in W and $v = \alpha w_1 + \beta w_2 \in W$ for all $\alpha, \beta \in \mathbb{R}$ and $w_1, w_2 \in W$.*

Definition 1.2.3. (**Spanned subspace**). *Given a set of vectors $S = \{v_i\}_{i=1}^m$, the* subspace spanned *by S is the set of all finite linear combinations $\sum_{i=1}^m \alpha_i v_i$ for all $\alpha_1, \alpha_2, \ldots, \alpha_n \in \mathbb{R}$. This subspace is usually denoted by span(S).*

For example, the two vectors $v_1 = [1, 0, 0]^T$ and $v_2 = [1, 1, 0]^T$ span a subspace of $\mathbb{R}^3$ whose elements are of the general form $v = [x, y, 0]^T$.

Definition 1.2.4. (**Linear independence**). *A set of vectors $S = \{v_i\}_{i=1}^m$ is* linearly independent *if the equation*

$$\alpha_1 v_1 + \alpha_2 v_2 + \ldots + \alpha_m v_m = 0$$

implies

$$\alpha_1 = \alpha_2 = \cdots = \alpha_m = 0.$$

On the other hand, a set of vectors $S = \{v_i\}_{i=1}^m$ is said to be *linearly dependent* if there exist $\alpha_1, \alpha_2, \ldots, \alpha_m \in \mathbb{R}$ not all zero such that

$$\alpha_1 v_1 + \alpha_2 v_2 + \ldots + \alpha_m v_m = 0.$$

Definition 1.2.5. (Basis.) *A set of vectors $B = \{b_i\}_{i=1}^n$ of a linear space V is said to be a* basis *if:*

1. *B is a linearly independent set*
2. *B spans the entire space V; i.e. $V = span(B)$.*

Fact 1.2.6. (Properties of a basis). *Suppose B and B' are two bases for a linear space V. Then:*

1. *B and B' contains exactly the same number of linearly independent vectors.*
   This number, say n, is the dimension *of the linear space V.*
2. *Let $B = \{b_i\}_{i=1}^n$ and $B = \{b'_i\}_{i=1}^n$. Then each basis vector of B can be expressed as a linear combination of those in B'; i.e.*

$$b_j = a_{1j} b'_1 + a_{2j} b'_2 + \cdots + a_{nj} b'_n = \sum_{i=1}^n a_{ji} b'_i, \tag{1.1}$$

   for some $a_{ij} \in \mathbb{R}, i, j = 1, 2, \ldots, n$.

3. *Any vector $v \in V$ can be written as a linear combination of vectors in either of the bases:*

$$v = x_1 b_1 + x_2 b_2 + \cdots + x_n b_n = x'_1 b'_1 + x'_2 b'_2 + \cdots + x'_n b'_n, \tag{1.2}$$

   where the coefficients $\{x_i \in \mathbb{R}\}_{i=1}^n$ and $\{x'_i \in \mathbb{R}\}_{i=1}^n$ are uniquely determined and are called the coordinates of v with respect to each basis.

In particular, if B and B' are two bases for the linear space $\mathbb{R}^n$, we may put the basis vectors as columns of two $n \times n$ matrices and also call them B and B' respectively:

$$B \doteq [b_1, b_2, \ldots, b_n], \qquad B' \doteq [b'_1, b'_2, \ldots, b'_n] \in \mathbb{R}^{n \times n}. \tag{1.3}$$

Then we can express the relationship between these two matrices in the matrix form $B = B'A$:

$$[b_1, b_2, \ldots, b_n] = [b'_1, b'_2, \ldots, b'_n] \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{bmatrix}. \tag{1.4}$$

The role of the $n \times n$ matrix A is to transform one basis (B') to the other (B). Since such a transformation can go the opposite way, the matrix A must be invertible. So we can write $B' = BA^{-1}$.

1.2.2 Column space and nullspace

Let A be a general $m \times n$ matrix that also conveniently represents a linear map from the vector space $\mathbb{R}^n$ to $\mathbb{R}^m$.

Definition 1.2.7. (Column space, nullspace and kernel).

- *The* column space *of A, denoted by $C(A)$ and also called* range *or* span *of A, is the subspace of $\mathbb{R}^m$ such that $y \in C(A)$ if and only if $y = Ax$ for some $x \in \mathbb{R}^n$.*
- *The* nullspace *of A, denoted by $N(A)$ and also called* kernel*, is the subspace of $\mathbb{R}^n$ such that $x \in N(A)$ if and only if $Ax = 0$.*

Note that the column space of a matrix A is exactly the span of all its columns vectors. Therefore, $C(A)$ is just the set of all linear combinations of the columns of A.

The nullspace of a matrix A is exactly the set of vectors which are orthogonal to all its row vectors.

For example, considering the following matrix $A = \begin{bmatrix} 1 & 1 & 2 \\ 2 & 1 & 3 \\ 3 & 1 & 4 \\ 4 & 1 & 5 \end{bmatrix}$, since the columns of A are in $\mathbb{R}^4$, $C(A)$ is a subspace of $\mathbb{R}^4$. On the other hand, the nullspace $N(A)$ contains all solutions to the equation $A\mathbf{x} = 0$. So $N(A)$ is a subspace of $\mathbb{R}^3$.

The notion of column space and nullspace is useful whenever the solution to a linear system of the form $Ax = b$ is considered. In terms of column and null spaces, this equation will have a solution if $b \in C(A)$ and will have a unique solution only if $N(A) = \oslash$ (the empty set). [1]

Definition 1.2.8. (Rank of a matrix). *The* rank of a matrix *is the dimension of its column space.*

$$rank(A) \doteq dim(C(A)). \tag{1.5}$$

Fact 1.2.9. (Properties of matrix rank). *For an arbitrary $m \times n$ matrix A, its rank has the following properties:*

1. $rank(A) = n - dim(N(A))$.
2. $0 \leq rank(A) \leq min\{m, n\}$.
3. *$rank(A)$ is equal to the maximum number of linearly independent column (or row) vectors of A.*
4. ***Sylvester's inequality:** Let B be an $n \times k$ matrix. Then AB is an $m \times k$ matrix and*

$$rank(A) + rank(B) - n \leq rank(AB) \leq min\{rank(A), rank(B)\}. \tag{1.6}$$

[1] In Matlab, the null space of a matrix A can be computed using the command $Z = null(A)$

5. *For any nonsingular matrices* $C \in \mathbb{R}^{m\times m}$ *and* $D \in \mathbb{R}^{n\times n}$, *we have*

$$rank(A) = rank(CAD). \tag{1.7}$$

The rank of a matrix A tells everything about the existence and number of solutions to a linear system of the form $Ax = b$.[2]

Fact 1.2.10. ***The rank tells everything.***
Let A be an $m \times n$ matrix of rank k. Then:

- If $k = m = n$, the linear system $Ax = b$ has *one unique solution*;
- If $k = n < m$, the linear system $Ax = b$ has either *no solution or a unique solution*;
- If $k = m < n$, the linear system $Ax = b$ has *infinitely many solutions*;
- If $k < n$ and $k < m$, the linear system $Ax = b$ has either *no solution or infinitely many solutions.*

1.3 Elimination and LU decomposition

Elimination process is the basic method for solving linear system of equations and is often called Gauss elimination. Say we want to solve the following system of equations

$$\begin{cases} x + 2y + z = 2 \\ 3x + 8y + z = 12 \\ 4y + z = 2 \end{cases}$$

which is equivalent to $Ax = b$, with $A = \begin{bmatrix} 1 & 2 & 1 \\ 3 & 8 & 1 \\ 0 & 4 & 1 \end{bmatrix}$ and $b = \begin{bmatrix} 2 \\ 12 \\ 2 \end{bmatrix}$.

The idea of the method is to eliminate one of the unknowns, say x, from all other $(n-1)$ equations and repeat the same operation with the second unknown y. The process gives

$$\underbrace{\left[\begin{array}{ccc} \boxed{1} & 2 & 1 \\ 3 & 8 & 1 \\ 0 & 4 & 1 \end{array}\right]}_{A} \underbrace{\left[\begin{array}{c} 2 \\ 12 \\ 2 \end{array}\right]}_{b} \Rightarrow \left[\begin{array}{ccc|c} \boxed{1} & 2 & 1 & 2 \\ 0 & \boxed{2} & -2 & 6 \\ 0 & 4 & 1 & 2 \end{array}\right] \Rightarrow \underbrace{\left[\begin{array}{ccc} \boxed{1} & 2 & 1 \\ 0 & \boxed{2} & -2 \\ 0 & 0 & \boxed{5} \end{array}\right]}_{U} \underbrace{\left[\begin{array}{c} 2 \\ 6 \\ -10 \end{array}\right]}_{c}$$

We have transformed $Ax = b$ into a new equivalent system $Ux = c$, and the principle of elimination is to transform the matrix A into a triangular matrix U.

The boxed numbers arer called *pivots*. Note that a pivot cannot be equal to zero. Here we have three pivots meaning the matrix is invertible and for sure we can solve the system $Ax = b$.

NOTE

- A pivot is always nonzero!

[2] In Matlab, the rank of a matrix A is just $rank(A)$.

- If a square $n \times n$ matrix has n pivots, then the matrix is invertible.
- The determinant of A is equal to the product of the pivots.
- **The number of pivots is the rank of the matrix**.

The reason for making A a triangular matrix U is that triangular systems are easier to solve. Indeed, we solve $Ux = c$ by back-substitution.

The final set of equations is:

$$\begin{cases} x + 2y + z = 2 \\ \quad\; 2y - 2z = 6 \\ \quad\quad\;\; 5z = -10 \end{cases}$$

We start by solving the last equation for z, then we go backward and find y and finally x. The solution of the linear system is then equal to $\begin{bmatrix} 2 \\ 1 \\ -2 \end{bmatrix}$

1.3.1 Elimination matrices and LU decomposition

The elimination process described above is better viewed when written with matrices. Lets do it step by step.

Step 1: we subtract 3 x row one from row two, which can be wrriten as

$$\begin{bmatrix} 1 & 0 & 0 \\ -3 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 2 & 1 \\ 3 & 8 & 1 \\ 0 & 4 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 2 & -2 \\ 0 & 4 & 1 \end{bmatrix} \Rightarrow E_{21}A = A_1.$$

Step 2: we subtract 2 x row two from row three, which can be wrriten as

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & -2 & 1 \end{bmatrix} \begin{bmatrix} 1 & 2 & 1 \\ 0 & 2 & -2 \\ 0 & 4 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 2 & -2 \\ 0 & 0 & 5 \end{bmatrix} \Rightarrow E_{32}A_1 = U.$$

Putting everything together, we finally have $E_{32}(E_{21}A) = U$. Which we can also write as $(E_{32}E_{21})A = U$.

If we call E the product of all elimination matrices, $E = E_{32}E_{21}$, then we have $EA = U$, where E is a lower triangular matrix and U is a upper triangular matrix.

Finally, taking the inverse of E, $L = E^{-1}$, we have $A = LU$, which is the *LU decomposition* of the matrix A.

The LU decomposition is useful because traingular systems are easier to solve. Since $A = LU$, the system $Ax = b$ is equivalent to the system $LUx = b$. So we can first solve the system $Ly = b$ wich is a lower triangular system (easy to solve), and then solve $Ux = y$ which is a upper triangular system (easy to solve).

We decompose the problem into two simple ones, which is a good idea!

1.3.2 Finding inverses

Definition 1.3.1. (Inverse of a square matrix). *A square matrix A is invertible if there exists a matrix A^{-1} such that $A^{-1}A = AA^{-1} = I$.*

We also know that A is invertible if and only if elimination process gives n pivots (not necessary different!).

Fact 1.3.2. (Properties of invertible matrices).

- *If A is invertible, then $Ax = 0$ has a unique solution which is $x = 0$.*
- *If A and B are invertible, then so is AB and the inverse of AB is*

$$(AB)^{-1} = B^{-1}A^{-1}$$

- *A diagonal matrix with no zero entry in the diagonal is invertible.*

Elimination process can be used to find the inverse of an invertible matrix. This process is called Gauss-Jordan elimination.

Say we want to find the inverse of $A = \begin{bmatrix} 1 & 3 \\ 2 & 7 \end{bmatrix}$.

We perform elimination on the augmented matrix $[A \;\; I]$.

$$[A \;\; I] = \begin{bmatrix} 1 & 3 & 1 & 0 \\ 2 & 7 & 0 & 1 \end{bmatrix} \to \begin{bmatrix} 1 & 3 & 1 & 0 \\ 0 & 1 & -2 & 1 \end{bmatrix} \to \begin{bmatrix} 1 & 0 & 7 & -3 \\ 0 & 1 & -2 & 1 \end{bmatrix} = [I \;\; B]$$

We started with $[A \;\; I]$ and ended with $[I \;\; B]$. If we again call E the product of all elimination matrices, we have $E[A \;\; I] = [I \;\; B]$.

The first equation $EA = I$ tells us that $E = A^{-1}$. The second equation $EI = B$ tells us that $E = B = A^{-1}$. So Gauss-Jordan elimination makes us go form A to A^{-1}.

NOTE

For a 2 by 2 matrix, there is a simple formula for getting the inverse (given that the matrix is invertible):

$$\begin{bmatrix} a & b \\ c & d \end{bmatrix}^{-1} = \frac{1}{ad - bc} \begin{bmatrix} d & -b \\ -c & a \end{bmatrix}$$

1.4 Orthogonality

Definition 1.4.1. (Inner product). *A function*

$$\langle ., . \rangle : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}$$

is an inner product[3] *if*

1. $\langle u, \alpha v + \beta w \rangle = \alpha \langle u, v \rangle + \beta \langle u, w \rangle \,, \; \forall \alpha, \beta \in \mathbb{R}$,
2. $\langle u, v \rangle = \langle v, u \rangle$,

[3] An inner product is sometimes called *dot product* and denoted by $u.v$

3. $\langle v, v\rangle \geq 0$, *and* $\langle v, v\rangle = 0 \Leftrightarrow v = 0$.

For each vector v, $\sqrt{\langle v, v\rangle}$ *is called its* norm.

Definition 1.4.2. (Orthogonality). *Two vectors* x, y *are said orthogonal if their inner product is zero :* $\langle x, y\rangle = 0$. *This is often indicated as* $x \perp y$.

The inner product induces the *standard* L_2*-norm* or *Ecludean norm*, $\|.\|_2$, which measures the length of each vector as:

$$\|x\|_2 \doteq \sqrt{x^T x} = \sqrt{x_1^2 + x_2^2 + \cdots + x_n^2}. \tag{1.8}$$

Theorem 1.4.3. *For any two vectors* $\mathbf{x}$ *and* $\mathbf{y}$*, we have*

$$\mathbf{x}^T\mathbf{y} = \|\mathbf{x}\|\|\mathbf{y}\| \cos\theta, \tag{1.9}$$

where θ *is the angle between* $\mathbf{x}$ *and* $\mathbf{y}$.

NOTE

- If x and y are orthogonal vectors, then $\|x\|^2 + \|y\|^2 = \|x + y\|^2$
- The zero vector is orthogonal to everybody

1.4.1 The fundamental theorem of linear algebra

We already know about the column space, $C(A)$, and the nullspace, $N(A)$, of a $m \times n$ matrix A. We can also define two subspaces related to the transposed matrix A^T, namely the rowspace and the left nullspace. The rowspace of A is the column space of A^T and the left nullspace of A is the nullspace of A^T. These four subpaces, two subspaces of $\mathbb{R}^m$ ($C(A)$ and $N(A^T)$) and two subspaces of $\mathbb{R}^n$ ($C(A^T)$ and $N(A)$), are called the *fundamental subspaces* [Str09].

Let us firt introduce the orthogonal complement to a linear subspace.

Definition 1.4.4. (Orthogonal complement to a subspace). *Given a subspace* S *of* $\mathbb{R}^n$*, we define its* orthogonal complement *to be the subspace* $S^\perp \subseteq \mathbb{R}^n$ *such that* $x \in S^\perp$ *if and only if* $x^T y = 0$ *for all* $y \in S$*, i.e.*

$$S^\perp = \{x \in \mathbb{R}^n : \langle x, y\rangle = 0,\ \forall y \in \mathbb{R}^n\}.$$

The orthogonal complement provides a particularly nice direct sum *decomposition of the vector space:* $\mathbb{R}^n = S \oplus S^\perp$.

With respect to any linear map A from $\mathbb{R}^n$ to $\mathbb{R}^m$, the space $\mathbb{R}^n$ can be decomposed as a direct sum of two subspaces,

$$\mathbb{R}^n = N(A) \oplus N(A)^\perp,$$

and $\mathbb{R}^m$ can be decomposed similarly as

$$\mathbb{R}^m = C(A) \oplus C(A)^\perp.$$

We have the following relationships:

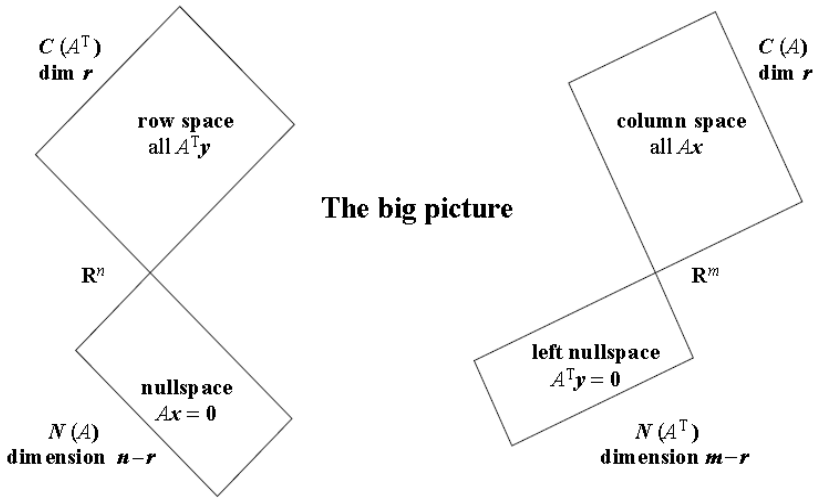

Figure 1.1: Relation between the four fundamental subspaces.

Theorem 1.4.5. *Let A be a linear map from $\mathbb{R}^n$ to $\mathbb{R}^m$. Then:*

(*a*) $N(A)^\perp = C(A^T)$,

(*b*) $C(A)^\perp = N(A^T)$,

(*c*) $N(A^T) = N(AA^T)$,

(*d*) $C(A) = N(AA^T)$.

Thus, the rowspace of A is the orthogonal complement to the nullspace of A in $\mathbb{R}^n$, and the the left nullspace of A is the orthogonal complement to the column space of A in $\mathbb{R}^m$. There is an important relation between these four subspaces bescribed by the fundamental theorem of linear algebra and depicted by Figure 1.1.

Theorem 1.4.6. (The fundamental theorem of linear algebra). *Let A be an $m \times n$ matrix of rank r. Then, we have the following decomposition:*

$$\mathbb{R}^n = C(A^T) \oplus N(A) \qquad \textit{and} \qquad \mathbb{R}^m = C(A) \oplus N(A^T), \tag{1.10}$$

where the dimensions are

$$\begin{array}{rcl} \dim(\mathrm{C}(A)) & = & r \\ \dim(\mathrm{C}(A^T)) & = & r \\ \dim(\mathrm{N}(A)) & = & n-r \\ \dim(\mathrm{N}(A^T)) & = & m-r \end{array}$$

1.4.2 The QR decomposition

The Gram-Schmidt procedure transforms a nonsingular matrix A into an orthogonal one. So the goal is to make the columns of A an orthonormal basis.

Theorem 1.4.7. (Gram-Schmidt procedure).
For every $A \in GL(n)$, there exists a lower triangular matrix $L \in \mathbb{R}^{n\times n}$ and an orthogonal matrix $E \in O(n)$ such that

$$A = LE. \tag{1.11}$$

How does this transformation of a nonsingular matrix into an orthogonal one works in practice? Denote the ith row of the given matrix A by a_i for $i = 1, \ldots, n$. The Gram-Schmidt procedure constructs L and E iteratively from the row vectors a_i:

$$\begin{array}{lcll} l_1 & \doteq & a_1 & \to \quad e_1 \doteq l_1 / \left\|l_1\right\|_2, \\ l_2 & \doteq & a_2 - \langle a_2, e_1\rangle e_1 & \to \quad e_2 \doteq l_2 / \left\|l_2\right\|_2, \\ \vdots & \vdots & \vdots & \vdots \qquad\qquad \vdots \\ l_n & \doteq & a_n - \sum_{i=1}^{n-1} \langle a_{i+1}, e_i\rangle e_i & \to \quad e_n \doteq l_n / \left\|l_n\right\|_2. \end{array}$$

Then $E = [e_1^T, e_2^T, \ldots, e_n^T]$, and the matrix L is obtained as

$$L = \begin{bmatrix} \left\|l_1\right\|_2 & 0 & \ldots & 0 \\ \langle a_2, e_1\rangle & \left\|l_2\right\|_2 & \ldots & 0 \\ \vdots & \ddots & \ddots & \vdots \\ \langle a_2, e_1\rangle & \ldots & \langle a_n, e_{n-1}\rangle & \left\|l_n\right\|_2 \end{bmatrix}$$

By construction E is orthogonal; i.e. $EE^T = E^T E = I$.

There are a few useful variations to the Gram-Schmidt procedure. By transposing $A = LE$, we get $A^T = E^T L^T \doteq QR$. Notice that $R = L^T$ is an upper triangular matrix. Thus, by applying Gram-Schmidt procedure to the transpose of a matrix, we can also decompose it into the form QR where Q is an orthognal matrix and R an upper triangular matrix. Such a decomposition is called the *QR decomposition*[4].

Furthermore, by inverting $A^T = E^T L^T$, we get $A^{-T} = L^{-T} E \doteq KE$. Notice that $K = L^{-T}$ is still an upper triangular matrix. Thus, we can also decompose any matrix into the form of an upper triangular matrix followed by on orthogonal one.

1.4.3 Projections

Theorem 1.4.8. (Orthogonal projection).
The projection of $\mathbf{x}$ onto $\mathbf{y}$ is the vector given by

$$\mathbf{b} = P_{\mathbf{y}}\mathbf{x}, \tag{1.12}$$

where $P_{\mathbf{y}}$ is the projection matrix defined by

$$P_{\mathbf{y}} = \frac{\mathbf{y}\mathbf{y}^T}{\mathbf{y}^T\mathbf{y}}. \tag{1.13}$$

[4] In Matlab, this can be done by the command $[Q, R] = qr(A)$.

Fact 1.4.9. (Properties of projection matrices). *A projection matrix P satisfies the following properties*

- $P^T = P$ *(symmetric)*
- $P^2 = P$

1.4.4 Orthogonal matrices

Definition 1.4.10. (Orthonormal basis). *Let $\mathbf{u}_1, \ldots, \mathbf{u}_n$ be a set of vectors in $\mathbb{R}^n$. This set of vectors forms an* orthonormal *basis of $\mathbb{R}^n$ if*

$$\mathbf{u}_i^T \mathbf{u}_j = \begin{cases} 1 & \text{if } i{=}j \\ 0 & \text{otherwise} \end{cases} \tag{1.14}$$

Let define a matrix $\mathbf{U}$ whose colums are the the $\mathbf{u}_i$'s:

$$\mathbf{U} = [\mathbf{u}_1 \ \cdots \ \mathbf{u}_n].$$

Then, the above equation (1.14) can be written in matrix form as

$$\mathbf{U}^T\mathbf{U} = I. \tag{1.15}$$

Definition 1.4.11. (Orthonormal matrix).
A matrix A that satisfies equation (1.15) is said to be orthogonal.

An important property of orthogonal matrices is given by the following theorem, i.e. an orthogonal matrix preserves length!

Theorem 1.4.12. (Length preservation).
The length (norm) of a vector $\mathbf{x}$ is not changed by multiplication by an orthogonal matrix $\mathbf{U}$:

$$\|\mathbf{U}\mathbf{x}\| = \|\mathbf{x}\|.$$

Theorem 1.4.13. *Let $\mathbf{U}$ be an orthogonal matrix. Then the matrix $\mathbf{U}\mathbf{U}^T$ projects any vector $\mathbf{b}$ onto the column space of $\mathbf{U}$.*

Furthermore, the difference between $\mathbf{b}$ and its projection $\mathbf{p}$ onto $C(\mathbf{U})$ is orthogonal to $C(\mathbf{U})$:

$$\mathbf{U}^T(\mathbf{b} - \mathbf{p}) = 0.$$

1.4.5 Least squares approximations

We cannot always solve $A\mathbf{x} = \mathbf{b}$ and the usual reason is *too many equations*, i.e. the matrix A has more rows than columns or there are more equations than unknwons ($m > n$). So, the n columns span a small subspace of $\mathbb{R}^m$ and because all measurements are not perfect, $\mathbf{b} \notin C(A)$.

We know we cannot solve the system if $\mathbf{b}$ is outside the column space $C(A)$. But, in practice we have to find not the exact solution, which is impossible, but the best approximate solution. This is achieved by the linear least squares (LLS) method.

Every vector $\mathbf{b} \in \mathbb{R}^m$ can be decomposed into two parts: $\mathbf{b} = \mathbf{p} + \mathbf{e}$, where $\mathbf{p}$ is in the column space $C(A)$ and $\mathbf{e}$ is in the nullspace of A^T. Since we cannot

solve $A\mathbf{x} = \mathbf{b}$, we want to find the solution $\widehat{\mathbf{x}}$ that makes the error $\|A\mathbf{x} - \mathbf{b}\|^2$ as small as possible.

We have

$$\|A\mathbf{x} - \mathbf{b}\|^2 = \|A\mathbf{x} - \mathbf{p} - \mathbf{e}\|^2 = \|A\mathbf{x} - \mathbf{p}\|^2 + \|\mathbf{e}\|^2$$

because $\mathbf{p}$ and $\mathbf{e}$ lie in orthogonal subspaces.

Now, note that the system $A\mathbf{x} = \mathbf{p}$ is solvable since $\mathbf{p} \in C(A)$. Let $\widehat{\mathbf{x}}$ be the vector such that $A\widehat{\mathbf{x}} = \mathbf{p}$. By taking $\mathbf{x} = \widehat{\mathbf{x}}$, the error becomes $\|A\widehat{\mathbf{x}} - \mathbf{b}\|^2 = \|\mathbf{e}\|^2$ which is the least possible error.

Therefore, the idea of linear least squares is to solve $A\widehat{\mathbf{x}} = \mathbf{p}$, which is solvable, instead of $A\mathbf{x} = \mathbf{b}$ which is not solvable. The remaining question is how do we solve $A\widehat{\mathbf{x}} = \mathbf{p}$?

Normal equation and LLS solution

We want to solve $A\widehat{\mathbf{x}} = \mathbf{p} = \mathbf{b} - \mathbf{e}$. Let multiply on the left by A^T:

$$A^T(A\mathbf{x}) = A^T\mathbf{b} - A^T\mathbf{e} = A^T\mathbf{b},$$

because $\mathbf{e} \in N(A^T)$.

Finding $\widehat{\mathbf{x}}$ then reduces to solving *the normal equation*

$$A^TA\mathbf{x} = A^T\mathbf{b}. \tag{1.16}$$

If the square matrix A^TA is invertible, then the LLS solution is given by

$$\widehat{\mathbf{x}} = (A^TA)^{-1}A^T\mathbf{b}. \tag{1.17}$$

NOTE

- We can use the LLS equation 1.17, if and only if the matrix A^TA is invertible.
- A^TA is invertible if and only if A has linearly independent columns.

1.5 Eigenvalues and eigenvectors

A linear map from $\mathbb{R}^n$ to itself is represented by a square $n \times n$ matrix A. For such a map, we sometimes are interested in subspaces of $\mathbb{R}^n$ that are 'invariant' under the map.This notion turns out to be closely related to *eigenvectors* of the matrix A.

Definition 1.5.1. (Eigenvalues and eigenvectors of a matrix). *Let A be an $n \times n$ complex matrix in $\mathbb{C}^{n\times n}$. A nonzero vector $v \in \mathbb{C}^n$ is said to be an* eigenvector *of A if*

$$Av = \lambda v,\ \lambda \in \mathbb{C}. \tag{1.18}$$

The scalar λ is called an eigenvalue *of A.*

The set of all eigenvalues of a matrix A is called its *spectrum*, denoted by $\sigma(A)$.

The Matlab command $[V, D] = eig(A)$ produces a diagonal matrix D of eigenvalues and a full-rank matrix V whose columns are the corresponding eigenvectors, so that $AV = VD$.

We give the following facts about eigenvalues and eigenvectors of a matrix without proof.

Fact 1.5.2. (Properties of eigenvalues and eigenvectors). *Given a matrix $A \in \mathbb{R}^{n\times n}$, we have:*

1. $\sigma(A) = \sigma(A^T)$
2. *The eigenvectors of A associated with different eigenvalues are linearly independent.*
3. *All eigenvalues of A are the roots of the (characteristic) polynomial equation $det(\lambda I - A) = 0$. Hence, $det(A)$ is equal to the product of all eigenvalues of A.*
4. *if $B = PAP^{-1}$ for some nonsigular matrix P, then $\sigma(B) = \sigma(A)$.*
5. *If A is a real matrix, then $\lambda \in \mathbb{C}$ is an eigenvalue implies that its conjugate $\overline{\lambda}$ is also an eigenvalue. That is, $\sigma(A) = \overline{\sigma}(A)$ for real matrices.*

For example lets consider the matrix $A = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$.

We have $A \begin{bmatrix} 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$, so $\begin{bmatrix} 1 \\ 1 \end{bmatrix}$ is an eigenvector associated to the eigenvalue $\lambda_1 = 1$.

We have $A \begin{bmatrix} -1 \\ 1 \end{bmatrix} = \begin{bmatrix} 1 \\ -1 \end{bmatrix}$, so $\begin{bmatrix} -1 \\ 1 \end{bmatrix}$ is an eigenvector associated to the eigenvalue $\lambda_2 = -1$.

One can verify that the sum of eigenvalues equals the trace of A, and the product of eigenavlues equals the determinant of A.

How to solve $Ax = \lambda x$?

It is not always possible to easily find eigenvalues and eigenvectors just looking at the matrix. We therefore have to explicitly solve the equation $Ax = \lambda x$.

We rewrite the equation in the form $(A - \lambda I)x = 0$. If there is a nonzero solution, then the matrix $(A - \lambda I)$ is singular. Thus, $det(A - \lambda I) = 0$. This is the **characteristic equation** of A.

Therefore, to find the eigenvalues/eigenvectors of A, we have to:

1. First find the eigenvalues which are the roots of the characteristic equation $det(A - \lambda I) = 0$.
2. Once we know the eigenvalues, λ's, we just have to find the nullspaces of the matrices $(A - \lambda I)$ and any vector in these linear spaces is an eigenvector.

Consider an example, $A = \begin{bmatrix} 3 & 1 \\ 1 & 3 \end{bmatrix}$.

We first find the eigenvalues of A

$$det(A-\lambda I) = \begin{vmatrix} 3-\lambda & 1 \\ 1 & 3-\lambda \end{vmatrix} = (3-\lambda)^2 - 1 = (\lambda-4)(\lambda-2)$$

Thus the eigenvalues are equal to $\begin{cases} \lambda_1 = 4 \\ \lambda_1 = 2 \end{cases}$

We finally find the eigenvectors

- For $\lambda_1 = 4$

$$A - \lambda_1 I = A - 4I = \begin{bmatrix} -1 & 1 \\ 1 & -1 \end{bmatrix}$$

So a vector in the nullspace of $(A-4I)$ is $x_1 = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$

- For $\lambda_2 = 2$

$$A - \lambda_2 I = A - 2I = \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix}$$

So a vector in the nullspace of $(A-2I)$ is $x_2 = \begin{bmatrix} -1 \\ 1 \end{bmatrix}$

1.5.1 Diagonalization and powers of A

.

Let A be a $n \times n$ matrix with n independent eigenvectors $X_i, i = 1 \dots n$. If we put all eigenvectors in the columns of a matrix S, then we have

$$\begin{aligned} AS &= A[X_1\ X_2\ \cdots X_n] \\ &= [AX_1\ AX_2\ \cdots AX_n] \\ &= [\lambda_1 X_1\ \lambda_2 X_2\ \cdots \lambda_n X_n] \\ &= [X_1\ X_2\ \cdots X_n] \begin{bmatrix} \lambda_1 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \lambda_n \end{bmatrix} \\ &= S\Lambda \end{aligned}$$

Since the eignevectors of A are independent, the matrix S is invertible. So, we have

$$AS = S\Lambda \Rightarrow A = S\Lambda S^{-1}$$

This is called the diagonalization of A.

The reader can verify that the matrix A of the previous example can be diagonalized as:

$$A = \begin{bmatrix} 1 & -1 \\ 1 & 1 \end{bmatrix} \begin{bmatrix} 4 & 0 \\ 0 & 2 \end{bmatrix} \begin{bmatrix} 1/2 & 1/2 \\ -1/2 & 1/2 \end{bmatrix}$$

NOTE

- We can diagonalize A only if A has n independent eigenvectors.
- $\forall\ k \geq 1,\ A^k = S\Lambda^k S^{-1}$

1.5.2 Symmetric matrices

Definition 1.5.3. (Symmetric matrix). *A matrix $S \in \mathbb{R}^{n\times n}$ is called* symmetric *if $S^T = S$.*

Definition 1.5.4. (Positive Definite Matrix). *A symmetric matrix S is called* positive (semi-)definite*, if $x^T Sx > 0$ (or $x^T Sx \geq 0$) for all $x \in \mathbb{R}^n$, denoted $S > 0$ (or $S \geq 0$).*

Fact 1.5.5. (Properties of symmetric matrices). *If S is a real symmetric matrix, then:*

1. *All eigenvalues of S must be real, i.e. $\sigma(S) \subset \mathbb{R}$.*
2. *Let (λ, v) be an eigenvalue-eigenvector pair. If $\lambda_i \neq \lambda_j$, then $v_i \perp v_j$; i.e. eigenvectors corresponding to distinct eigenvalues are orthogonal.*
3. *There always exist n orthonormal eigenvectors of S, which form a basis for $\mathbb{R}^n$.*
4. *S is positive (semi-)definite if all eigenvalues are positive (nonnegative).*
5. *If $S \geq 0$ and $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_n$, then*

$$max_{\|x\|_2=1} \langle x, Sx\rangle = \lambda_1, \text{ and } \quad min_{\|x\|_2=1} \langle x, Sx\rangle = \lambda_n.$$

From point 3, we see that if $V = [v_1, v_2, \ldots, v_n] \in \mathbb{R}^{n\times n}$ is the matrix of all eigenvectors, and $\Lambda = diag\{\lambda_1, \lambda_2, \ldots, \lambda_n\}$ is the diagonal matrix of the corresponding eigenvalues, then we can write

$$S = V\Lambda V^T,$$

where V is an orthonormal matrix. This decomposition of S into a diagonal matrix Λ and an orthogonal matrix V, $S = V\Lambda V^T$ is called the *diagonalization* of S.

NOTE
The inverse problem of retrieving A from the symmetric matrix $S = A^T A$ is usually solved by *Choleski factorization.*
For the given S, its eigenvalues must be nonnegative. Thus, we have $S = V\Lambda V^T = A^T A$ for $A = \Lambda^{\frac{1}{2}} V^T$, where $\Lambda^{\frac{1}{2}} = diag\{\sqrt{\sigma_1}, \ldots, \sqrt{\sigma_1}\}$ is the 'square root' of the diagonal matrix Λ.
Notice that since $RR^T = I$ for any orthogonal matrix, the solution for A is not unique: RA is also a solution!

1.6 Vector and matrix norms

A vector norm is a scalar function that assigns a nonnegative scalar to every vector $x \in \mathbb{R}^n$.

Definition 1.6.1. (Vetcor norm). *A norm $\|.\| : \mathbb{R}^n \to [0, +\infty)$ is a scalar function with the following properties:*

- Nonnegativity

$$\|x\| \geq 0 \; \forall x \in \mathbb{R}^n, \textit{with } \|x\| = 0 \textit{ if and only if } x = 0.$$

- Homogeneity

$$\|\alpha x\| = |\alpha| \|x\|, \; \forall x \in \mathbb{R}^n \textit{ and any } \alpha \in \mathbb{R}.$$

- Triangular inequality

$$\|x + y\| \leq \|x\| + \|y\|, \; \forall x, y \in \mathbb{R}^n.$$

The most commonly used vector norms are the Euclidean norm, the 1-norm and the ∞-norm, respectively, given by

$$\|x\| = \sqrt{x^T x} = \sqrt{\sum_{i=1}^{n} x_i^2} \tag{1.19}$$

$$\|x\|_1 = \sum_{i=1}^{n} |x_i| \tag{1.20}$$

$$\|x\|_\infty = \max_i |x_i| \tag{1.21}$$

More generally, the p-norm of a vector is defined by

$$\|x\|_p = (\sum_{i=1}^{n} |x_i|^p)^{1/p} \tag{1.22}$$

Theorem 1.6.2. (Chauchy-Schawrtz inequality). *For any two vectors* $\mathbf{x}$ *and* $\mathbf{y}$*, we have*

$$|\mathbf{x}^T \mathbf{y}| \leq \|\mathbf{x}\| \|\mathbf{y}\|. \tag{1.23}$$

There are several matrix norms, but we will only consider those which are *induced* by vector norms.

Definition 1.6.3. (Matrix norm). *We write* $\|A\|$ *the matrix norm induced by Euclidean vector norm, which is given by*

$$\|A\| = \sup_{x \neq 0} \frac{\|Ax\|}{\|x\|}. \tag{1.24}$$

NOTE

- The norm $\|A\|_1$ and $\|A\|_\infty$ are defined in a similar way.
- The norm $\|A\|$ is also known as the spectral norm and satisfies the following relation

$$\|A\| = \max\{\sqrt{\lambda}; \lambda \text{ is an eigenvalue of } A^T A\}.$$

- The induced 2-norm of a matrix $A \in \mathbb{R}^{m\times n}$ is different from the '2-norm' of A viewed as a vector in $\mathbb{R}^{mn}$. To distinguish them, the latter one is conventionally called the *Frobenius* norm of A, precisely defined as

$$\|A\|_f = \sqrt{\sum_{i,j} a_{ij}^2}.$$

Notice that $\sum_{i,j} a_{ij}^2$ is nothing but the trace of A^TA (or AA^T). Thus, we have

$$\|A\|_f \doteq \sqrt{\text{trace}(A^TA)} = \sqrt{\sigma_1^2 + \sigma_2^2 + \ldots + \sigma_n^2}.$$

1.7 Singular value decomposition

The singular value decomposition (SVD) is a useful tool to capture essential features of a matrix (that represent a linear map), such as its rank, range space, null space, and induced norm, as well as to 'generalize' the concept of eigenvalue-eigenvector pairs to non-square matrices.

1.7.1 Algebraic derivation

Given a matrix $A \in \mathbb{R}^{m\times n}$, we have the following theorem.

Theorem 1.7.1. (Singular value decomposition of a matrix). *Let $A \in \mathbb{R}^{m\times n}$ have rank p. Furthermore, suppose, without loss of generality, that $m \geq n$. Then*

- *$\exists U \in \mathbb{R}^{m\times p}$ whose columns are orthonormal,*
- *$\exists V \in \mathbb{R}^{n\times p}$ whose columns are orthonormal, and*
- *$\exists \Sigma \in \mathbb{R}^{p\times p}$, $\Sigma = \text{diag}\{\sigma_1, \sigma_2, \ldots, \sigma_p\}$ diagonal with $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_p$*

such that $A = U\Sigma V^T$.

The factorization $A = U\Sigma V^T$ is the SVD of the matrix A and the above theorem shows that such factorization always exists.

It has to be noticed that the rank of A is equal to the number of non-zero singular values.

We explain how SVD is obtained by construction bellow.

- Compute A^TA: it is symmetric and positive semi-definite of dimension $n \times n$. Then order its eigenvalues in decreasing order and call them $\sigma_1^2 \geq \sigma_2^2 \geq \cdots \geq \sigma_n^2 \geq 0$. Call the σ_i^2's *singular values.*
- From an orthonormal set of eigenvectors of A^TA, create an orthonormal basis for $\mathbb{R}^n$ such that

$$span\{v_1, v_2, \ldots, v_p\} = range(A^T),\ span\{v_{p+1}, \ldots, v_n\} = null(A).$$

Note that the latter eigenvectors correspond to the zero singular values, since $null(A^TA) = null(A)$.

- Define u_i such that $Av_i = \sigma_i u_i$, $\forall i = 1, \ldots, p$, and see that the set $\{u_i\}_{i=1}^p$ is orthonormal.
- Complete the basis $\{u_i\}_{i=1}^p$, which spans range(A) (by construction), to all $\mathbb{R}^m$.
- Then,

$$A[v_1, v_2, \ldots, v_n] = [u_1, u_2, \ldots, u_m] \begin{bmatrix} \sigma_1 & 0 & \cdots & \cdots & \cdots & 0 \\ 0 & \sigma_2 & \cdots & \cdots & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \sigma_p & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \vdots & \vdots & 0_n \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \cdots & \cdots & \cdots & 0_m \end{bmatrix}$$

  which we name $A\widetilde{V} = \widetilde{U}\widetilde{\Sigma}$.
- Hence, $A = \widetilde{U}\widetilde{\Sigma}\widetilde{V}^T$.

Then, the claim follows by deleting the columns of $\widetilde{U}$ and the rows of $\widetilde{V}^T$ that multiply the zero singular values.

In Matlab, the command to compute the SVD of a given matrix A is $[U, S, V] = svd(A)$, which returns matrices U, S, V satisfying $A = USV^T$.

1.7.2 Geometric interpretation

Notice that in the SVD of a square matrix $A = U\Sigma V^T \in \mathbb{R}^{n\times n}$, columns of $U = [u_1, u_2, \ldots, u_n]$ and columns of $V = [v_1, v_2, \ldots, v_n]$ form orthonormal bases for $\mathbb{R}^n$. The SVD essentially states that if A (as a linear map) maps a point x to y, then coordinates of y w.r.t. the basis U are related to coordinates of x w.r.t. the basis V by the diagonal matrix Σ that scales each coordinate by the corresponding singular value.

Theorem 1.7.2. *Let $A \in \mathbb{R}^{n\times n} = U\Sigma V^T$ be a square matrix. Then A maps the unit sphere $\mathbb{S}^{n-1} = \{x \in \mathbb{R}^n : \|x\|_2 = 1\}$ to an ellipsoide with semi-axes $\sigma_i u_i$, where u_i is the ith column of U.*

The theorem is illustrated in Figure 1.2 for the case $n = 2$.

1.7.3 Relation with eigen-decomposition

From $A = U\Sigma V^T$, we see that

$$\begin{aligned} A^T A &= (U\Sigma V^T)^T (U\Sigma V^T) \\ &= (V\Sigma U^T)(U\Sigma V^T) \\ &= V\Sigma^2 V^T \end{aligned}$$

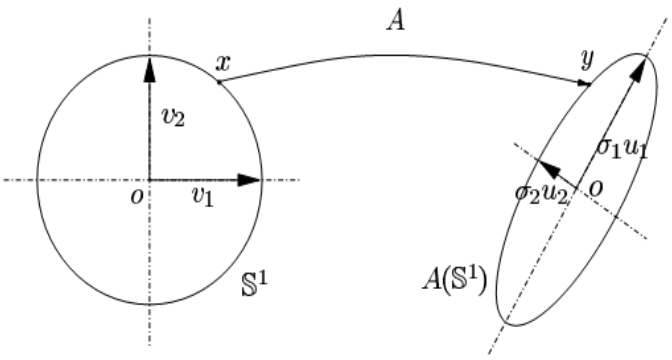

Figure 1.2: The image of a unit sphere on the left under a nonsigular map $A \in \mathbb{R}^{2\times 2}$ is an ellipsoide on the right.

This last equation $A^TA = V\Sigma^2V^T$ is the diagonalization of the symmetric matrix A^TA. Similarly, we can show $AA^T = U\Sigma^2U^T$ which is the eigendecomposition (or diagonalization) of the symmetric matrix AA^T.

Therefore, we can conclude that:

- the columns of the orthogonal matrix V are the eigenvectors of A^TA,
- the columns of the orthogonal matrix U are the eigenvectors of AA^T,
- the singular values of A are the square roots of eigenvalues of A^TA (or AA^T).

1.7.4 Some properties of the SVD

Problems involving orthogonal projections onto invariant subspaces of A, such as the linear least-squares (LLS) problem, can be easily solved using the SVD.

Fact 1.7.3. (Rank and inverse).

- *The rank of a matrix is equal to the number of non-zero singular values.*
- *A square $n \times n$ matrix A is nonsingular* iff $\sigma_i \neq 0\ \forall i$.
- *If A is a $n \times n$ nonsingular matrix, then A^{-1} is given by*

$$A^{-1} = V\Sigma^{-1}U,$$

where $\Sigma^{-1} = diag\{\frac{1}{\sigma_1}, \cdots, \frac{1}{\sigma_n}\}$.

1.7.5 Sum of rank one matrices

The SVD of an $m \times n$ matrix A of rank r is given by $A = U\Sigma V^T$, which can also be written as

$$A = \sum_{i=1}^{r} \sigma_i u_i v_i^T,$$

where u_i and v_i are the columns of U and V respectively.

Note that each term of this sum, i.e. each $u_i v_i^T$, is a $m \times n$ matrix of rank one. So, the matrix A is a sum of rank one matrices that are orthogonal with respect to the matrix inner product.

Truncating the sum at p terms defines a rank p matrix $A_p = \sum_{i=1}^{p} \sigma_i u_i v_i^T$. If we approximate A with A_p, then we make an error equals to $E_p = A - A_p = \sum_{i=p+1}^{k} \sigma_i u_i v_i^T$. It can be shown that A_p is the best rank p approximation to the matrix A.

The low rank approximation of a matrix can, for example, be used for image compression.

1.7.6 Generalized inverse

Definition 1.7.4. (Generalized (Moore Penrose) inverse). *Given a matrix $A \in \mathbb{R}^{m\times n}$ of rank k with its SVD $A = U\Sigma V^T$, we define* the generalized inverse *of A to be*

$$A^{\mp} = V\Sigma^{\mp}U^T, \qquad \text{where} \qquad \Sigma^{\mp} = \begin{bmatrix} \Sigma^{-1} & 0 \\ 0 & 0 \end{bmatrix}_{n\times m}.$$

The generalized inverse is also called the pseudo-inverse.

In Matlab, the pseudo-inverse of a matrix is computed by the command $X = pinv(A)$.

Fact 1.7.5. (Properties of generalized inverse).

- $AA^{\mp}A = A, \qquad A^{\mp}AA^{\mp} = A^{\mp}$.

The generalize inverse can then be used to solve linear equations in general.

Proposition 1.7.6. (Least-squares solution of a linear system). *Consider the problem $Ax = b$ wit $A \in \mathbb{R}^{m\times n}$ of rank $r \leq \min(m, n)$. The solution x^* that minimizes $\|Ax - b\|_2$ is given by $x^* = A^{\mp}b$.*

Proposition 1.7.7. (Condition number). *Consider the problem $Ax = b$ and consider a 'perturbed' full-rank problem $(A + \delta A)(x + \delta x) = b$. Then we have:*

$$\frac{\|\delta x\|_2}{\|x\|_2} \leq \left\|A^{\mp}\right\|_2 \|A\|_2 \frac{\|\delta A\|_2}{\|A\|_2} \doteq k(A)\frac{\|\delta A\|_2}{\|A\|_2}, \tag{1.25}$$

where $k(A) = \|A^{\mp}\|_2 \|A\|_2$ is called the condition number *of A.*

It is easy to see that $k(A) = \sigma_1/\sigma_n$ if A is invertible.

1.7.7 SVD line fitting

Fitting a line to a set of points is a very important practical problem. The linear least squares (LLS) method provides a solution, but the SVD of a matrix also yields a simple method.

Let $\mathbf{p}_i = (x_i, y_i)^T$ be a set of m points on the plane ($m \geq 2$), and let

$$ax + by + c = 0$$

be the equation of a line.

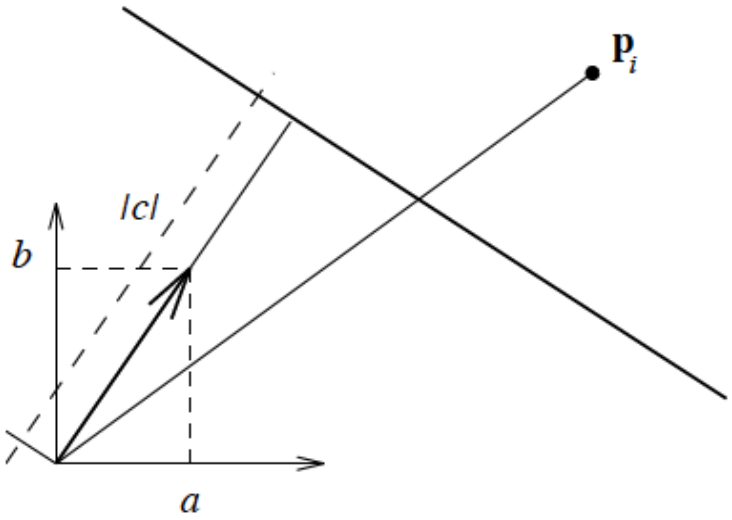

Figure 1.3: Line geometry in 2D space.

Without loss of generality, we can assume that

$$\|\mathbf{n}\| = a^2 + b^2 = 1. \tag{1.26}$$

This is because, multiplying the lefthand side of the line equation by a nonzero constant does not change the line. The vector $\mathbf{n} = (a, b)^T$ is called the *line normal*.

As shown by figure 1.3, the distance from the line to the origin is $|c|$, and the distance between the line and a point $\mathbf{p}_i$ is given by

$$d_i = |ax_i + by_i - c| = |\mathbf{p}_i^T \mathbf{n} - c|. \tag{1.27}$$

To find the best-fit line, we have to minimize the sum of the squared distances. Thus, if we let $\mathbf{d} = (d_1, \ldots, d_m)^T$ and $P = (\mathbf{p}_1, \ldots, \mathbf{p}_m)^T$, then the best-fit line achieves the

$$\min_{\|\mathbf{n}\|=1} \|\mathbf{d}\|^2 = \min_{\|\mathbf{n}\|=1} \|P\mathbf{n} - c\mathbf{1}\|^2, \tag{1.28}$$

where $\mathbf{1}$ is a vector of m zeros.

Best line fit

To fit a line (a, b, c) to a set of m points $\mathbf{p}_i$ collected in the $m \times 2$ matrix $P = (\mathbf{p}_1, \ldots, \mathbf{p}_m)^T$, use the following procedure.

1. compute the centroid of the points

$$\mathbf{p} = \frac{1}{m} P^T \mathbf{1}$$

2. form the matrix of centered coordinates

$$Q = P - \mathbf{1}\mathbf{p}^T$$

3. compute the SVD of Q

$$Q = U\Sigma V^T$$

4. the line is the second column of the 2×2 matrix V:

$$\mathbf{n} = (a, b)^T = \mathbf{v}_2$$

5. the fird coefficient of the line is

$$c = \mathbf{p}^T\mathbf{n}$$

6. th eresidual of the fit is

$$\min_{\|\mathbf{n}\|=1} \|\mathbf{d}\|^2 = \sigma_2$$

NOTE
A useful exercise is to figure out why this procedure works!
Hint: start with the fact that, since the parameter c does not appear in the constraint ($\|\mathbf{n}\| = 1$), at the minimum of the optimization problem, we must have

$$\frac{\partial \|\mathbf{d}\|^2}{\partial c} = 0.$$

1.8 Principal component analysis

Principal component analysis (PCA) is one of the most widely used technique for data analysis. It is a technique that compute a linear transformation to map a high dimensional space into a lower dimensional space.

Why do we need to reduce dimension? Because problems arise when performing recognition in a high dimensional space. This is known as the *curse of dimensionality*. For instance, humans have excellent ability to discern patterns in 1,2 or 3-dimensions, but these capabilities degrade drastically for 4 or higher dimensions.

Significant improvements in classification results can be achieved by first mapping the data into a lower dimensionality space. This is what PCA does!

PCA is a feature extraction approach: i.e. we want to create a subset of new features which are combinations of the existing features. The problem can then be stated as follows:

Fact 1.8.1. Feature extraction problem. *Given a feature space $x_i \in \mathbb{R}^N$, find a mapping $y = f(x)$, $f : \mathbb{R}^N \to \mathbb{R}^M$, with $M < N$ such that the new feature vector $y_i \in \mathbb{R}^M$ preserves as much as possible the information or structure in $\mathbb{R}^N$.*

In general, the optimal mapping $y = f(x)$ is not a linear function. However, in practice, feature extraction is commonly limited to linear transforms: $y = Tx$.

$$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_N \end{bmatrix} \overset{linear\ mapping}{\rightarrow} \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ x_M \end{bmatrix} = \begin{bmatrix} T_{11} & T_{12} & \cdots & T_{1N} \\ T_{21} & T_{22} & \cdots & T_{2N} \\ \vdots & \vdots & \ddots & \vdots \\ T_{M1} & T_{M2} & \cdots & T_{MN} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_N \end{bmatrix}$$

Reducing dimensionality necessary implies *information loss*. Since we want to preserve as much as possible information, we have to minimize the representation error $\|x - y\|$.

Interestingly (but with no surprise) the optimal mapping, i.e. the one that minimizes this error, has something to do with the notion of eigenvectors and eigenvalues!

We now explain how to perform PCA.

1.8.1 Data representation

Say we have N samples, each of which is a point in $\mathbb{R}^L$. We represent our data as a data matrix $\mathbf{X} = [X_1, \cdots, X_N]$, where each column X_i represents a sample point of dimension L. Thus, $\mathbf{X}$ is a matrix of size $L \times N$.

The covariance matrix of the data is then given by the equation:

$$\mathbb{C}_X = \widetilde{\mathbf{X}}\widetilde{\mathbf{X}}^T, \tag{1.29}$$

where $\widetilde{\mathbf{X}} = [X_1 - \bar{X}, \cdots, X_N - \bar{X}]$ is the zero-mean data matrix ($\bar{X}$ being the mean vector).

The covariance matrix $\mathbb{C}_X$ is a square $L \times L$ symmetric matrix that encodes the correlation between the different features; the diagonal element of $\mathbb{C}_X$ contain the variances of each feature, and the off-diagonal elements correspond to the covariances between different features.

1.8.2 PCA derivation

First, note that if all our features were uncorrelated then the covariance matrix would be diagonal. Since the goal is to remove redundancy in the data (which corresponds to removing correlation between the variables), we have to find a transformation (a change of basis) that makes $\mathbb{C}_X$ is diagonal matrix.

In other words, we want to find a matrix P, such that if $\mathbf{Y} = P\mathbf{X}$, then the covariance matrix of $\mathbf{Y}$ is diagonal.

Since $\mathbb{C}_X$ is a symmetric matrix, we can diagonalize it as $\mathbb{C}_X = V\Lambda V^T$. Which gives $\Lambda = V^T\mathbb{C}_X V$.

Let choose $P = V^T$, i.e. $\mathbf{Y} = P\mathbf{X} = V^T\mathbf{X}$. Then, the covariance matrix of

transformed data $\mathbf{Y}$ is

$$\begin{aligned}\mathbb{C}_Y &= \mathbf{Y}\mathbf{Y}^T \\ &= (V^T\mathbf{X})(V^T\mathbf{X})^T \\ &= V^T(\mathbf{X}\mathbf{X}^T)V \\ &= V^T\mathbb{C}_X V \\ &= \Lambda\end{aligned}$$

So, setting $P = V^T$ makes the covariance matrix of transformed data to be diagonal, what we wanted to achieve.

The rows of the matrix P are the vectors of the new basis on which to re-express the data. This vectors are called *principal components.*

We can observe that

- The principal components of $\mathbf{X}$ are the eigenvectors of the covariance matrix $\mathbb{C}_X$.
- The corresponding eigenvalues give the amount of information carried by each principal component.

1.8.3 Dimension reduction

Usually, we want to reduce the dimensionality of the problem, i.e. we want to represent each of our data point in a lower dimensional space $\mathbb{R}^K$, with $K \ll L$.

Dimensionality reduction is achieved by projecting the data points onto the K principal components corresponding to the K largest eigenvalues of the covariance matrix $\mathbb{C}_X$.

One question is how to choose the number of principal components (or how to fix the value of K)? To choose K, we use the following criterion which takes into account the amount of information carried by each eigenvector:

How many components to keep?
Choose K such that

$$(\sum_{i=1}^{K}\lambda_i)/(\sum_{i=1}^{N}\lambda_i) > Threshold.$$

Typical threshold values are 0.9 or 0.95.

It can be shown that the reconstruction error $e = \left\|X - \widehat{X}\right\|$ is minimized using the principal components. This error is equal to

$$e = \frac{1}{2}\sum_{i=K+1}^{N}\lambda_i.$$

Data normalization
Finally, note that the principal components depend on the *units* and *range* of

the original data. Therefore, we should always normalize the data prior to using PCA. A common normalization method is to transform all the data to have zero mean and unit standard deviation:

$$X_i' = \frac{X_i - \mu}{\sigma},$$

μ and σ being, respectively, the mean and standard deviation of the X_i's.

1.8.4 PCA algorithm

Here, we summarize the methodology to perform PCA. Let $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N$ be vectors in $\mathbb{R}^L$.

- Step 1: compute the mean vector $\overline{\mathbf{x}} = \frac{1}{N}\sum_{i=1}^{N} \mathbf{x}_i$.
- Step 2: subtract the mean $\Phi_i = \mathbf{x}_i - \overline{\mathbf{x}}$ for $i = 1, \ldots, N$.
- Step 3: form the $L \times N$ matrix $\mathbf{A} = [\Phi_1\ \Phi_2\ \cdots\ \Phi_N]$ and compute

$$\Sigma = \frac{1}{L}\sum_{i=1}^{N} \Phi_i \Phi_i^T = \frac{1}{L}\mathbf{A}\mathbf{A}^T.$$

- Step 4: compute the eigenvalues $\lambda_1 > \lambda_2 > \cdots > \lambda_L$ and the corresponding eigenvectors $\mathbf{u_1}, \ldots, \mathbf{u_L}$ of Σ.
- Step 5: since Σ is symmetric, its eigenvectors form a basis. So any vector $\mathbf{x} - \overline{\mathbf{x}}$ can be expressed as $\mathbf{x} - \overline{\mathbf{x}} = \sum_{i=1}^{L} b_i \mathbf{u_i}$.

  To perform dimensionality reduction, keep only the vectors corresponding to the K largest eigenvalues:

$$\widehat{\mathbf{x}} - \overline{\mathbf{x}} = \sum_{i=1}^{K} b_i \mathbf{u_i} \text{ where } K << N.$$

1.8.5 Size trick

In many applications, it happens that we have few data but many variables, i.e. $N \ll L$. For example, if we have 100 images each of size 400x600, then our data matrix $\mathbf{X}$ has dimensions $N = 100$ and $L = 240,000$.

In such a case, computing the eigenvalues/eigenvectors of $\mathbb{C}_X$ as above may be difficult if L is too large.

The **size trick** here is to work with the other covariance matrix

$$\mathbb{C}_X' = \widetilde{\mathbf{X}}^T \widetilde{\mathbf{X}}.$$

This new matrix is of size $N \times N$ instead of $L \times L$ as $\mathbb{C}_X$.

Let $\mathbb{C}_X' = V'\Lambda' V'^T$, be the eigen-decomposition of $\mathbb{C}_X'$.

The fact is that both matrices $\mathbb{C}_X$ and $\mathbb{C}_X'$ have the same non-zero eigenvalues (they have same rank). Hence, we have $\Lambda = \Lambda'$.

What about the eigenvectors?
Let $\mathbf{x}$ be an eigenvector of $\mathbb{C}_X'$. Then $\mathbb{C}_X'\mathbf{x} = \lambda\mathbf{x}$.

That is $\widetilde{\mathbf{X}}^T\widetilde{\mathbf{X}}\mathbf{x} = \lambda\mathbf{x} \Rightarrow (\widetilde{\mathbf{X}}\widetilde{\mathbf{X}}^T)\widetilde{\mathbf{X}}\mathbf{x} = \lambda\widetilde{\mathbf{X}}\mathbf{x} \Rightarrow \Sigma(\widetilde{\mathbf{X}}\mathbf{x}) = \lambda\widetilde{\mathbf{X}}\mathbf{x}$.

In other words, we have $\mathbb{C}_X(\widetilde{\mathbf{X}}\mathbf{x}) = \lambda(\widetilde{\mathbf{X}}\mathbf{x})$, and we can conclude that the eigenvectors of $\mathbb{C}_X$ and those of C'_X are related by the equation

$$V = \widetilde{\mathbf{X}}V'.$$

1.8.6 PCA & SVD

From Section 1.7, we know that the SVD of a matrix A is given by $A = U\Lambda V^T$, where U and V are orthogonal matrices.

Moreover, we have also seen that the columns of V are the eigenvectors of AA^T, while the columns of U are eigenvectors of A^TA.

So, we can perform PCA without computing the covariance matrix, but from SVD directly (of course, after transforming the data to have zero-mean). The advantage of this option is that we directly apply the size-trick. In short, if $\mathbf{X}$ is an $L \times N$ matrix with $N < L$, then we know the rank of $\mathbf{X}$ is at most equal to N (see properties of rank in Section 1.2.2). So we don't need to compute all L singular values (and vectors) of $\mathbf{X}$ as many of them will be zero. We can simply compute N singular values (and the corresponding vectors), and this is exactly the size-trick explained above.

Chapter 2

Review of Probability and Statistics

2.1 Basics of probability

Probability is the study of *random variables*, a random variable being the outcome of a random experiment, i.e. an experiment whose result is not known in advance (e.g. throwing a die).

In these notes, random variables will be abbreviated by r.v.

2.1.1 Probability space

Definition 2.1.1. (Probability space).
A probability space is a triplet $\Omega, \mathcal{F}, P$ *where*

- Ω *is a nonempty set, called the* sample space*;*
- $\mathcal{F}$ *is a collection of subsets of* Ω *closed under countable set operations -such a collection is called a* σ-filled*. The elements of* $\mathcal{F}$ *are called* events*;*
- P *is a countably additive function from* $\mathcal{F}$ *to* $[0,1]$ *such that* $P(\Omega) = 1$*, called a* probability measure*.*

Let's give an example of probability space.

Exemple 2.1.1. Choosing uniformly in $\{1, 2, \ldots, N\}$.

Uniformly choosing a value ω in $\{1, 2, \ldots, N\}$ means that the N values are equally likely to be selected. In this case, the sample space is $\Omega = \{1, 2, \ldots, N\}$. For any subset $A \subset \Omega$, we define $P(A) = |A|/N$, where $|A|$ is the number of elements in A. For instance, $P(\{2, 5\}) = 2/N$.

Proposition 2.1.2. Axioms of probability.

1. $0 \leq P(A) \leq 1 \; \forall A \subset \Omega$,
2. $P(\Omega) = 1$,

3. *If $A_1, A_2, \ldots, A_n$ are mutually exclusive events (i.e. $P(A_i \cap A_j) = 0$), then:*

$$P(A_1 \cup A_2 \cup \cdots \cup A_n) = \sum_{i=1}^{n} P(A_i).$$

Fact 2.1.3. Other laws of probability.

$$P(A) = 1 - P(\overline{A}).$$

$$P(A \cup B) = P(A) + P(B) - P(A \cap B).$$

$$P(A) = P(A \cap B) + P(A \cap \overline{B}).$$

2.1.2 Conditional probability and independence

The mathematical notion of conditional probabilities formalizes the idea of how observations modify our belief about the likelihood of events.

Definition 2.1.4. Conditional probability.
The conditional probability is the probability of an event given some evidence. Formally, we define the probability of A given B by

$$P(A|B) = \frac{P(A \cap B)}{P(B)}. \tag{2.1}$$

Note that this definition makes sense only if $P(B) > 0$. If $P(B) = 0$, we define $P(A|B) = 0$.

Furthermore, the above definition leads to the following formulas, known respectively as the *chain rule* and as the *law of total probability*:

$$P(A \cap B) = P(A|B)P(B) = P(B|A)P(A). \tag{2.2}$$

$$P(A) = P(A|B)P(B) + P(A|\overline{B})P(\overline{B}). \tag{2.3}$$

One important result regarding conditional probability is the Bayes theorem.

Theorem 2.1.5. Bayes theorem.

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}. \tag{2.4}$$

This formula is trivially obtain from the chain rule. However, it is of crucial importance in practice. First, Bayes rule can also be written as

$$P(A|B) = \frac{P(B|A)P(A)}{P(B|A)P(A) + P(B|\overline{A})P(\overline{A})}.$$

Now, think of event A as a possible 'cause' of some effect B. For instance, your alarm can sound either if there is fire or also if there is no fire (false alarm). Given that the alarm sounds, what is the probability that there is no fire?

It may happen that knowing that an event occurs does not change the probability of another event. In such a case, the events are said to be *independent*.

Definition 2.1.6. Independence.
Two events A and B are independent *if*

$$P(A \cap B) = P(A)P(B).$$

It is important not to confuse *independent* and *disjoint*! If two events A and B are disjoint, then they are independent only if at least one of them has probability 0 (proof left as exercice).

Definition 2.1.7. General definition of independence.
A collection of events $A_i, i \in I$ are mutually independent if for any subcollection $\{i, j, \ldots, k\} \subset I$, one has

$$P(A_i \cap A_j \cap \cdots \cap A_k) = P(A_i)P(A_j) \cdots P(A_k).$$

Exemple 2.1.2. Is it true that

$$P(A \cap B \cap C) = P(A|B)P(B|C)P(C)?$$

If true, provide a proof; if false, give a counterexample.

2.2 Random variables

A random variable (r.v.) is the value we assign to the outcome of a random experiment (i.e., a function that assigns a real number to each event).

Definition 2.2.1. Distribution.
A r.v. X is discrete *if it takes values in a countable set $\{x_n, n \geq 1\} \subset \mathbb{R}$. We can define $P(X = x_n) = p_n$ with $p_n > 0$ and $\sum_n p_n = 1$. The collection $\{x_n, p_n, n \geq 1\}$ is then called the* Probability Mass Function (pmf) *of the r.v. X.*

In general, the function $\{P(X \leq x) =: F(x), x \in \mathbb{R}\}$ - called the cumulative distribution function (cdf) *of X - completely characterizes the 'statistics' of X. For short, we call the cdf, the* distribution *of X.*

Proposition 2.2.2. *A function $F_X(.) : \mathbb{R} \to \mathbb{R}$ is the cdf of some r.v. iff*

- *$lim_{x \to -\infty} F_X(x) = 0$;*
- *$lim_{x \to +\infty} F_X(x) = 1$;*
- *If $x < y$, then $F_X(x) \leq F_X(y)$;*
- *$F_X(x)$ is right-continuous, i.e. $y \downarrow x$ implies $F_X(y) \to F_X(x)$.*

Definition 2.2.3. *A r.v. X is* continuous *if one can write*

$$P(X \in (a, b]) = F_X(x) = \int_a^b f_X(x)dx,$$

for all real number $a < b$.
In this expression, $f_X(.)$ is a nonnegative function called the probability density function (pdf) *of X.*

We can see from the above definition that the pdf $f_X(.)$ is the derivative of the cdf $F_X(.)$.

2.2.1 Examples of random variables

Now we review some important r.v. Let X be a discrete r.v.

- **Bernoulli**
  We say that X has a *Bernoulli distribution* with parameter $p \in [0,1]$, and we write $X =_D \mathcal{B}(p)$ if
  $$P(X = 1) = p \text{ and } P(X = 0) = 1 - p. \tag{2.5}$$

- **Binomial**
  X has a *binomial distribution* with parameters $n \in \{1, 2, \ldots\}$ and $p \in [0,1]$, and we write $X =_D \mathcal{B}(n,p)$ if
  $$P(X = m) = \left(\begin{array}{c} n \\ m \end{array} \right) p^m (1-p)^{n-m}, \text{ for } m = 0, 1, \ldots, n. \tag{2.6}$$

- **Geometric**
  X has a *geometric distribution* with parameter $p \in (0,1]$, and we write $X =_D \mathcal{G}(p)$ if
  $$P(X = n) = p(1-p)^{n-1} \text{ for } n \geq 0. \tag{2.7}$$

- **Poisson**
  The r.v. X has a *Poisson distribution* with parameter λ, and we write $X =_D \mathcal{P}(\lambda)$ if
  $$P(X = n) = \frac{\lambda^n}{n!} e^{-\lambda} \text{ for } n \geq 0. \tag{2.8}$$

A r.v. X with geometric distribution has the remarkable property of being *memoryless*. That is,

$$P(X \geq n + m | X \geq n) = P(X \geq m), \ \forall m, n \geq 0. \tag{2.9}$$

The interpretation of this result is that if X is the lifetime of a product, then the residual lifetime $X - n$ of that product, if it is still operating after n years, has the same distribution as that of a new product.

Now, let X be a continuous r.v.

- **Uniform**
  X is uniformly distributed in the interval $[a,b]$ where $a < b$, and we write $X =_D \mathcal{U}[a,b]$ if
  $$f_X(x) = \left\{ \begin{array}{l} \frac{1}{b-a}, \text{ if } x \in [a,b] \\ 0, \text{ otherwise.} \end{array} \right. \tag{2.10}$$

- **Exponential**
  X is exponentially distributed with rate $\lambda > 0$, and we write $X =_D Exp(\lambda)$, if
  $$f_X(x) = \left\{ \begin{array}{l} \lambda e^{-\lambda x}, \text{ if } x > 0 \\ 0, \text{ otherwise.} \end{array} \right. \tag{2.11}$$

Exponentially distributed r.v. are memoryless in the sense that

$$P(X > s + t | X > t) = P(X > s), \ \forall s, t > 0. \tag{2.12}$$

The interpretation of this property is the same as for the geometric distribution.

2.2.2 Expectation and moments

Suppose we play a game of chance a large number K of times. Each time we play, we have a probability p_n of winning x_n, for $n \geq 1$. So we expect to win approximately Kp_n times out of K. Accordingly, our total earnings should be approximately equal to $\sum_n Kp_n x_n$. Thus, our earnings should on average be $\sum_n p_n x_n$ per instance of the game. That value, the average earnings per experiment, is the *interpretation* of the expected value of the r.v. that represents our earnings.

Let's now formally define the *expected value* $\mathbb{E}(X)$ of a r.v. X.

Definition 2.2.4. Expected value.
For a discrete r.v., $\mathbb{E}(X) = \sum_n x_n P(X = x_n)$.
For a continuous r.v., $\mathbb{E}(X) = \int_{-\infty}^{\infty} x f_X(x) dx$.

If the sums yield ∞ or $-\infty$, we say that the expectation of the r.v. is not define.

Definition 2.2.5. Moments.

- *The n-th moment of a r.v. X is* $\mathbb{E}(X^n)$.
- *The* variance *of a r.v. X is define by*

$$\mathbb{V}ar(X) \doteq \mathbb{E}[(X - \mathbb{E}(X))^2] = \mathbb{E}(X^2) - \{\mathbb{E}(X)\}^2. \tag{2.13}$$

The variance measures the 'spread' of the distribution around the mean. A r.v. with a zero variance is constant. The larger the variance, the more 'uncertain' the r.v. is, in the mean square sense.

2.2.3 Function of random variables

Let X be a r.v. and $h : \mathbb{R} \to \mathbb{R}$ be a function. Since X is some function from Ω to $\mathbb{R}$, so is $h(X)$. We have the following result:

Theorem 2.2.6. *If $h(.)$ is differentiable, then the density of the r.v. $Y = h(X)$ is given by*

$$f_Y(y) = \frac{1}{|h'(x)|} f_X(x)|_{h(x)=y}. \tag{2.14}$$

For example, if $X =_D \mathcal{U}[0,1]$ and $Y = (3+X)^2$, then

$$f_Y(y) = \frac{1}{2(3+x)} \chi_{x\in[0,1]}|_{(3+x)^2=y} = \frac{1}{2\sqrt{y}} \chi_{y\in[\sqrt{3},2]}.$$

Proposition 2.2.7. Expected value.

- *If X is a discrete r.v., then*

$$\mathbb{E}(h(X)) = \sum_n h(x_n) p_n, \text{ where } p_n = P(X = x_n). \tag{2.15}$$

- *If X is a continuous r.v., then*

$$\mathbb{E}(h(X)) = \int_{-\infty}^{\infty} h(x) f_X(x) dx. \tag{2.16}$$

Let's go back to the above example, where $X =_D \mathcal{U}[0,1]$ and $Y = (3+X)^2$. Using Equation 2.16 we find

$$\mathbb{E}(h(X)) = \int_0^1 (3+x)^2 dx = [\frac{1}{3}(3+x)^3]_0^1 = \frac{1}{3}[4^3 - 3^3] = \frac{37}{3}.$$

Using the pdf of Y, f_Y, we find

$$\mathbb{E}(h(X)) = \mathbb{E}(Y) = \int_{\sqrt{3}}^2 y f_Y(y) dy = \int_{\sqrt{3}}^2 y \frac{1}{2\sqrt{y}} dy.$$

If we do the change of variables $y = (3+x)^2$, so that $dy = 2(3+x)dx = 2\sqrt(y)dx$, then we can write the integral above as

$$\mathbb{E}(h(X)) = \int_0^1 (3+x)^2 dx,$$

so it works!!!

Fact 2.2.8. Linearity of expectation.

Let X, Y be two r.v.'s and $f, g : \mathbb{R} \to \mathbb{R}$ be two functions. Then, we have:

$$\mathbb{E}(af(X) + bg(Y)) = a\mathbb{E}(f(X)) + b\mathbb{E}(g(Y)) \tag{2.17}$$

2.2.4 Useful inequalities

Inequalities are often useful to estimate some expected values. Here are a few particularly useful ones.

- **Exponential bound:**

$$1 + x \leq exp(x). \tag{2.18}$$

- **Chebychev:**

$$P(X \leq a) \leq \mathbb{E}(X^2)/a^2. \tag{2.19}$$

- **Markov inequality:** If $f(.)$ is nonnegative and nondecreasing on $[a, +\infty)$, then

$$P(X \leq a) \leq \{\mathbb{E}(f(X))\}/f(a). \tag{2.20}$$

- **Jensen inequality:** If $f(.)$ is convex, then

$$\mathbb{E}(f(X)) \geq f(\mathbb{E}(X)). \tag{2.21}$$

2.2.5 Joint statistics

Here we look at a collection of r.v., i.e. a collection of functions of the outcome of the same random experiment. For each experiment, nature chooses a single value of $\omega \in \Omega$. The different observations, say X, Y, Z are all functions of the same ω. That is, we model these observations as $X(\omega)$, $Y(\omega)$, and $Z(\omega)$. The fact is that observing $X(\omega)$ provides some information about $Y(\omega)$. And the interesting question is how we can use the information that some observations contain about some other r.v. that we do not observe directly?

Definition 2.2.9. Joint distribution. *Let $\{X(\omega), Y(\omega)\}$ be a pair of r.v.'s.*

The joint distribution *of $\{X(\omega), Y(\omega)\}$ is specified by the joint cumulative distribution function $F_{X,Y}$ defined as:*

$$F_{X,Y}(x, y) = P(X \leq x \text{ and } Y \leq y),\ x, y \in \mathbb{R}.$$

In the continuous case we have:

$$F_{X,Y}(x, y) = \int_{-\infty}^{x} \int_{-\infty}^{y} f_{X,Y}(u, v) du dv,$$

where the nonnegative function $f_{X,Y}(x, y)$ is called the joint pdf *of the r.v.'s.*

Note that the joint distribution contains more information than the two individual distributions. Individual distributions can be obtained from the joint distribution but the converse is not true!

Proposition 2.2.10. Marginalization.

In the discrete case:

$$F_X(x) = \sum_y F_{X,Y}(x, y). \tag{2.22}$$

In the continuous case:

$$F_X(x) = \int_{-\infty}^{\infty} F_{X,Y}(x, y) dy. \tag{2.23}$$

Definition 2.2.11. Covariance. *The* covariance *of a pair of r.v.'s (X, Y) is defined as*

$$Cov(X, Y) = \mathbb{E}[(X - \mathbb{E}(X))(Y - \mathbb{E}(Y))] = \mathbb{E}(XY) - \mathbb{E}(X)\mathbb{E}(Y). \tag{2.24}$$

The covariance is a measure of dependence. The idea is that if $\mathbb{E}(XY)$ is larger than $\mathbb{E}(X)\mathbb{E}(Y)$, then X and Y tend to be large or small together more than if they were independent.

The *correlation* coefficient between X and Y is given by:

$$\rho_{XY} = \frac{\mathrm{Cov}\,(X, Y)}{\sqrt{\mathbb{V}\mathrm{ar}\,(X)\mathbb{V}\mathrm{ar}\,(Y)}}. \tag{2.25}$$

Figure 2.1 illustrates the meaning of correlation. The r.v.'s are positively (respectively negatively or un-) correlated if $\mathrm{Cov}\,(X, Y) > 0$ (respectively if $\mathrm{Cov}\,(X, Y) < 0$ or if $\mathrm{Cov}\,(X, Y) = 0$).

Note that

$$X \text{ and } Y \text{ independent} \quad \Rightarrow \quad X \text{ and } Y \text{ uncorrelated.}$$
$$\not\Leftarrow$$

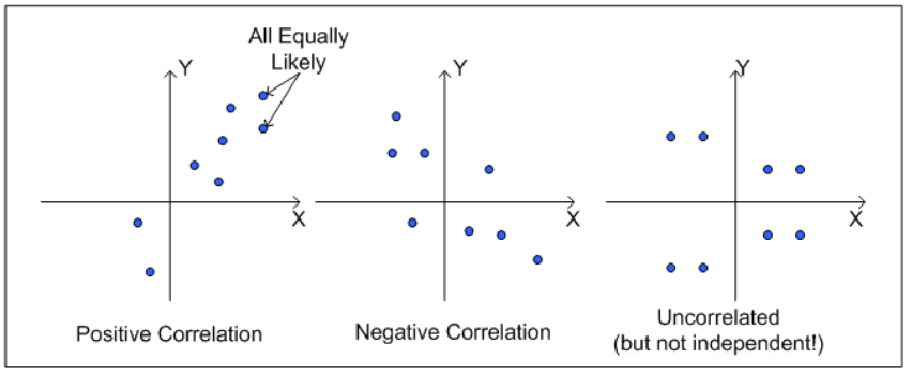

Figure 2.1: Meaning of correlation.

2.2.6 Random vectors

A random vector is a vector $\mathbf{X}$ whose elements are random variables $X_1, \ldots, X_n$. Let's note $\mathbf{X} = [X_1, \ldots, X_n]^T$.

We define the expected value of $\mathbf{X}$ to be the vector of expected values $\mathbb{E}(X_i)$, i.e. $\mathbb{E}(\mathbf{X}) = [\mathbb{E}(X_1), \ldots, \mathbb{E}(X_n)]^T$.

For a random matrix $\mathbf{W}$ whose entry (i, j) is the random variable $W_{i,j}$ for $i = 1, \ldots, m$ and $j = 1, \ldots, n$, we define $\mathbb{E}(\mathbf{W})$ to be the matrix whose entry (i, j) is $\mathbb{E}(W_{i,j})$.

Definition 2.2.12. Covariance matrices.

If $\mathbf{X}$ *and* $\mathbf{Y}$ *are two random vectors, we define the* covariance matrix *of* $\mathbf{X}$ *and* $\mathbf{Y}$ *by:*

$$\Sigma_{\mathbf{X},\mathbf{Y}} \doteq Cov(\mathbf{X}, \mathbf{Y}) \doteq \mathbb{E}[(\mathbf{X} - \mathbb{E}(\mathbf{X}))(\mathbf{Y} - \mathbb{E}(\mathbf{Y}))^T] = \mathbb{E}(\mathbf{X}\mathbf{Y}^T) - \mathbb{E}(\mathbf{X})\mathbb{E}(\mathbf{Y}^T) \tag{2.26}$$

We also have the following results:

$$\text{Cov}(\mathbf{AX}, \mathbf{BY}) = \mathbf{A}\text{Cov}(\mathbf{X}, \mathbf{Y})\mathbf{B}^T.$$

And

$$\mathbb{E}(\mathbf{X}^T\mathbf{Y}) = tr\{\mathbb{E}(\mathbf{X}\mathbf{Y}^T)\}.$$

We say that two random variables $\{X, Y\}$ are independent if

$$P(X \subset A \text{ and } Y \subset B) = P(X \subset A)P(Y \subset B),$$

for all subsets A and B of $\mathbb{R}$.

More generally, a collection of random variables are said to be mutually independent if the probability that any finite subcollection of them belongs to any given subsets is the product of the probabilities.

Theorem 2.2.13. Independence.

- *The random variables X and Y are said independent if and only if the joint cdf $F_{X,Y}(x,y)$ is equal to $F_X(x)F_Y(y)$, for all x, y.*

  A collection of r.v.'s are mutually independent if the joint cdf of any sub-collection is the product of the cdf.

- *If the r.v.'s X, Y have a joint pdf $f_{X,Y}(x,y)$, they are independent if and only if $f_{X,Y}(x,y) = f_X(x)f_Y(y)$, for all x, y.*

- *If X and Y are independent, then $f(X)$ and $g(X)$ are independent.*

- *If X and Y are independent, then $\mathbb{E}(XY) = \mathbb{E}(X)\mathbb{E}(Y)$. (The converse is not true!).*

- *If $X, Y, \ldots, W$ are independent, then*

$$\mathbb{E}(XY \cdots W) = \mathbb{E}(X)\mathbb{E}(Y) \cdots \mathbb{E}(W).$$

- *If X and Y are continuous and independent, then*

$$f_{X+Y}(x) = \int_{-\infty}^{\infty} f_X(u) f_Y(x-u) du. \tag{2.27}$$

  Thus, the pdf of the sum of two independent r.v.'s is the convolution *of their pdf.*

2.2.7 Conditional expectation

Conditional expectation tells us how to use the observation of a r.v. $Y(\omega)$ to estimate another r.v. It is the best guess about $X(\omega)$ given $Y(\omega)$ if we want to minimize the mean squared error.

Definition 2.2.14. Conditional expectation.

- *Suppose that the pair of discrete r.v.'s (X, Y) takes values in $\{x_1, \ldots, x_m\} \times \{y_1, \ldots, y_n\}$. We define the conditional expectation of X given $Y = y_j$ as*

$$\mathbb{E}[X|Y = y_j] = \sum_i x_i P(X = x_i | Y = y_j). \tag{2.28}$$

  And we define the conditional expectation of X given Y as

$$\mathbb{E}[X|Y] = \sum_j \mathbb{E}[X|Y = y_j]. \tag{2.29}$$

- *In the case where (X, Y) have a joint density $f(x,y)$ and marginal densities $f_X(x)$ and $f_Y(y)$ we define*

$$\mathbb{E}[X|Y = y] = \int_{-\infty}^{\infty} x f_{X|Y}(x|y) dx, \tag{2.30}$$

  where the conditional density of X given that $Y = y$ is defined as

$$f_{X|Y}(x|y) = \frac{f_{X,Y}(x,y)}{f_Y(y)}. \tag{2.31}$$

Theorem 2.2.15. MMSE.
$\mathbb{E}[X|Y]$ *is the function* $g(Y)$ *that minimizes* $\mathbb{E}((X - g(Y))^2)$.

This means that $\mathbb{E}[X|Y]$ is the best guest about X based on Y (best in the sense of minimum mean squared error (MMSE)).

Lemma 2.2.16. *A random variable* $g(Y)$ *is equal to* $\mathbb{E}[X|Y]$ *if and only if*

$$\mathbb{E}(g(X)h(Y)) = \mathbb{E}(Xh(X)) \text{ for any function } h(.). \tag{2.32}$$

We give some nice properties of conditional expectation.

Theorem 2.2.17. Properties of conditional expectation.

- *Linearity:*

$$\mathbb{E}[a_1X_1 + a_2X_2|Y] = a_1\mathbb{E}[X_1|Y] + a_2\mathbb{E}[X_2|Y]; \tag{2.33}$$

- *Known factor:*

$$\mathbb{E}[Xk(Y)|Y] = k(Y)\mathbb{E}[X|Y]; \tag{2.34}$$

- *Averaging:*

$$\mathbb{E}(\mathbb{E}[X|Y]) = \mathbb{E}(X); \tag{2.35}$$

- *Independence: If* X *and* Y *are independent, then*

$$\mathbb{E}[X|Y] = \mathbb{E}(X); \tag{2.36}$$

- *Smoothing:*

$$\mathbb{E}[\mathbb{E}[X|Y, Z]|Y] = \mathbb{E}[X|Y]. \tag{2.37}$$

2.3 Gaussian random variables

The Gaussian distribution is the most used distribution in practice. (This is because of the central limit theorem).

Definition 2.3.1. Standard Gaussian random variable.
We say that X *is a standard Gaussian (or standard normal) r.v., and we write* $X =_D \mathcal{N}(0, 1)$, *if*

$$f_X(x) = \frac{1}{\sqrt{2\pi}} exp\{-x^2/2\}, \text{ for } x \in \mathbb{R}.$$

Definition 2.3.2. X *is a Gaussian r.v. with mean* μ *and variance* σ^2, *and we write* $X =_D \mathcal{N}(\mu, \sigma^2)$, *if* $X = \mu + \sigma Y$, *where* $Y = \mathcal{N}(0, 1)$.
The pdf of X *is given by*

$$f_X(x) = \frac{1}{\sqrt{2\pi\sigma^2}} exp\{-\frac{(x-\mu)^2}{2\sigma^2}\}, \text{ for } x \in \mathbb{R}.$$

Random variables $\mathbf{X}$ are said te be jointly Gaussian if $\mathbf{u}^T\mathbf{X}$ is Gaussian for any vector $\mathbf{u}$.

Theorem 2.3.3. Independence and correlation.
Jointly Gaussian random variables are independent iff they are uncorrelated.

Note that this property is not true in general. Only, Gaussian r.v.'s have this nice property!

For jointly Gaussian r.v.'s the derivation of the conditional expectation is remarkably simple: the conditional expectation is linear in the observations.

Theorem 2.3.4. *Let (X, Y) be two jointly Gaussian random variables. Then*

- *We have*
$$\mathbb{E}[X|Y] = \mathbb{E}(X) + \frac{Cov(X,Y)}{\mathbb{V}ar(Y)}(Y - \mathbb{E}(Y)). \tag{2.38}$$
- *Let $(\mathbf{X}, \mathbf{Y})$ be two jointly Gaussian random vectors.*
  1. *If $\Sigma_{\mathbf{Y}}$ is invertible, then*
  $$\mathbb{E}[\mathbf{X}|\mathbf{Y}] = \mathbb{E}(\mathbf{X}) + \Sigma_{\mathbf{X},\mathbf{Y}}\Sigma_{\mathbf{Y}}^{-1}(\mathbf{Y} - \mathbb{E}(\mathbf{Y})). \tag{2.39}$$
  2. *if $\Sigma_{\mathbf{Y}}$ is not invertible, then*
  $$\mathbb{E}[\mathbf{X}|\mathbf{Y}] = \mathbb{E}(\mathbf{X}) + \Sigma_{\mathbf{X},\mathbf{Y}}\Sigma_{\mathbf{Y}}^{\mp}(\mathbf{Y} - \mathbb{E}(\mathbf{Y})), \tag{2.40}$$
  where $\Sigma_{\mathbf{Y}}^{\mp}$ is such that $\Sigma_{\mathbf{X},\mathbf{Y}} = \Sigma_{\mathbf{X},\mathbf{Y}}\Sigma_{\mathbf{Y}}^{\mp}\Sigma_{\mathbf{Y}}$

The next theorem justifies the use of Gaussian distribution in real life for large n.

Theorem 2.3.5. Central limit theorem.
Let $\{X_n, n \geq n\}$ be a set of independent, identically distributed (i.i.d.) r.v.'s governed by an arbitrary probability distribution with mean μ and finite variance σ^2. Let's define the r.v. Y_n as:

$$Y_n \doteq \frac{1}{n}\sum_{i=1}^{n} X_i.$$

Then, we have

$$Y_n \to_D \mathcal{N}(\mu, \sigma^2/n) \text{ as } n \to \infty.$$

2.4 Estimation

The estimation problem is roughly as follows: we observe $Y \in \mathbb{R}$ and we must compute an estimate of $X \in \mathbb{R}$ based on Y that is closed to X in some sense.

An estimator of X given Y is a function g of Y. The estimator $g(Y)$ is *unbiased* if $\mathbb{E}(g(Y)) = \mathbb{E}(X)$, i.e., if its mean is the same as that of X. (We know that $\mathbb{E}[X|Y]$ is unbiased).

As the number of observations gets larger, we look at the estimator $\hat{X}_n$ of X given $(Y_1, \ldots, Y_n)$: $\hat{X}_n = g_n(Y_1, \ldots, Y_n)$. We say that $\hat{X}_n$ is *asymptotically unbiased* if $\lim \mathbb{E}(\hat{X}_n) = \mathbb{E}(X)$.

Here we focus on a class of estimators that are linear in the observations.

Definition 2.4.1. Linear least squares estimator (LLSE).

Let (X, Y) be a pair of r.v.'s on some probability space. The linear least squares estimator *(LLSE) of X given Y, denoted by $L[X|Y]$, is the linear function $a + bY$ of Y that minimizes $\mathbb{E}((X - a - bY)^2)$.*

In the multivariate case, let $\mathbf{X}, \mathbf{Y}$ be vectors of r.v.'s on some probability space. The LLSE of $\mathbf{X}$ given $\mathbf{Y}$, denoted $L[\mathbf{X}|\mathbf{Y}]$, is the linear function $\mathbf{a} + \mathbf{BY}$ that minimizes $\mathbb{E}(\|\mathbf{X} - \mathbf{a} - \mathbf{BY}\|^2)$.

The next theorem summarizes the key results.

Theorem 2.4.2. LLSE.

We have

$$L[X|Y] = \mathbb{E}(X) + \frac{Cov(X, Y)}{\mathbb{V}ar(Y)}(Y - \mathbb{E}(X)). \tag{2.41}$$

Also, if $\Sigma_{\mathbf{Y}}$ is invertible,

$$\mathbb{L}[\mathbf{X}|\mathbf{Y}] = \mathbb{E}(\mathbf{X}) + \Sigma_{\mathbf{X},\mathbf{Y}}\Sigma_{\mathbf{Y}}^{-1}(\mathbf{Y} - \mathbb{E}(\mathbf{Y})). \tag{2.42}$$

Finally, if $\Sigma_{\mathbf{Y}}$ is not invertible,

$$\mathbb{L}[\mathbf{X}|\mathbf{Y}] = \mathbb{E}(\mathbf{X}) + \Sigma_{\mathbf{X},\mathbf{Y}}\Sigma_{\mathbf{Y}}^{\mp}(\mathbf{Y} - \mathbb{E}(\mathbf{Y})), \tag{2.43}$$

where $\Sigma_{\mathbf{Y}}^{\mp}$ is such that $\Sigma_{\mathbf{X},\mathbf{Y}}\Sigma_{\mathbf{Y}}^{\mp} = \Sigma_{\mathbf{Y},\mathbf{Y}}$.

For r.v. that are jointly Gaussian, the linear least squares estimator is the conditional expectation, i.e. $L[X|Y] = \mathbb{E}[X|Y]$. However, in non-Gaussian cases, the conditional expectation is not linear and we have

$$\mathbb{E}((X - \mathbb{E}[X|Y])^2) \leq \mathbb{E}((X - L[X|Y])^2).$$

The inequality is strict unless the conditional expectation happens to be linear as in the case of Gaussian r.v.

Recursive estimation

Suppose we have found an estimate of X given Y, i.e. $L[X|Y]$ and we get a new observation Z. How do we update the estimate? That is, how do we obtain $L[X|Y, Z] = bY + cZ$ if we know $\hat{X} = L[X|Y] = aY$?

Answer to this question is given by $L[X|Y, Z] = L[X|Y] + k(Z - L[Z|Y])$, where we chose k that minimizes the mean squared error.

These ideas lead to the Kalman filter which is a recursive estimator linear in the observations (we will consider the derivation of the Kalman filter in chapter 4).

2.5 Stochastic processes

A *stochastic process* can be defined as a collection of random variables $X_1, X_2, \ldots, X_t$ parameterized by time. That is we are interested in the evolution in time of random variables. There are a number of aspects of a stochastic process that are interesting:

- dependencies between variables in the sequence;

- the frequency at which 'boundary events' occur;
- various kind of long-term averages.

Here we consider only two kind of stochastics processes: *arrival processes* and *Markov processes*.

Arrival processes such as the Bernoulli and Poisson processes model the frequency of 'arrivals' or 'successes' in some time-dependent model. In general, we consider models in which interarrival time are independent random variables.

Markov processes are sequences where the next value depends only on the current value, i.e.:

$$P(X_{t+1} = k|X_{1:t}) = P(X_{t+1} = k|X_t).$$

2.5.1 Arrival processes

The Bernoulli process and the Poisson process are two arrival processes, which are discrete- and continuous-time analogs of each other, respectively.

Definition 2.5.1. Bernoulli process.
Let $X = \{X_n, n \geq 1\}$ be i.i.d. with $P(X_n = 1) = p$ and $P(X_n = 0) = 1 - p$. The discrete-time random process is called the Bernoulli process.

A Bernoulli process is a sequence of i.i.d. Bernoulli trials, each of which takes unit time. The key property of a Bernoulli process is that it is *memoryless*, i.e.:

$$P(T = t + n|T > n) = P(T = t). \tag{2.44}$$

This process models, for example, flipping a coin. If we denote by 'success' the event corresponding to getting a head, we have the following results:

- *The number S of 'successes' in n trials* is a r.v. governed by a Binomial distribution: $S =_D \mathcal{B}(n, p)$.

$$f_S(k) = \left(\begin{array}{c} n \\ k \end{array} \right) p^k (1-p)^{n-k}. \tag{2.45}$$

- *The number T of trials up to and including the fisrt 'success'* is a r.v. governed by a geometric distribution: $T =_D \mathcal{G}(p)$.

$$f_T(n) = p(1-p)^{n-1}. \tag{2.46}$$

- *The time Y_k until the kth 'success*' is just the sum of the first k interarrival times $T_1, T_2, \ldots, T_k$ which are i.i.d. geometric r.v. Y_k is distributed according to the *Pascal pmf of order k*:

$$f_{Y_k}(n) = \left(\begin{array}{c} n-1 \\ k-1 \end{array} \right) p^k (1-p)^{n-k}. \tag{2.47}$$

  Note that if $k = 1$, then the Pascal pmf is equal to the geometric pmf. Thus, the time until the first 'success' is distributed according to $\mathcal{G}(p)$.

The Poisson process is the continuous-time analog of the Bernoulli process.

Definition 2.5.2. Poisson process.

Let $P(k,\tau)$ be the probablility that k arrivals occur during an interval of length τ. Then a process is a Poisson process if the following properties are met:

- Time-homogeneity*: $P(k,\tau)$ is the same for any interval of length τ;*
- Independence*: the number of arrivals during an interval is independent of the history of arrivals outside the interval.*
- Small interval probabilities*: $P(k>1,\tau)$ is negligible in comparison to $P(0,\tau)$ and $P(1,\tau)$ as τ decreases.*

Note that the pdf describing the number of arrivals in an interval of length τ is a Poisson distribution with parameter $\lambda\tau$:

$$P(k,\tau) = e^{-\lambda\tau}\frac{(\lambda\tau)^k}{k!}. \tag{2.48}$$

We have the following results:

- *The time T until the fisrt arrival* is a r.v. governed by an exponential distribution: $T =_D Exp(\lambda)$.

$$f_T(t) = \lambda e^{-\lambda t}. \tag{2.49}$$

- *The time Y_k until the kth arrival* is just the sum of the first k interarrival times. Y_k is distributed according to the *Erlang pdf of order k*:

$$f_{Y_k}(n) = \frac{\lambda^k y^{k-1} e^{-\lambda y}}{(k-1)!}. \tag{2.50}$$

Definition 2.5.3. Stationarity.

Let $X = \{X(t), t \in \mathbb{R}\}$ be a random process. We say that X is stationary *if $X = \{X(t+u), t \in \mathbb{R}\}$ has the same distribution for all $u \in \mathbb{R}$.*

In other words, the statistics do not change over time.

Definition 2.5.4. Time reversibility.

Let $X = \{X(t), t \in \mathbb{R}\}$ be a random process. We say that X is time-reversible *if $X = \{X(t), t \in \mathbb{R}\}$ and $X = \{X(u-t), t \in \mathbb{R}\}$ have the same distribution for all $u \in \mathbb{R}$.*

2.5.2 Markov chains

Roughly speaking, a Markov chain models the random motion in time of an object in a countable set. The key feature is that the futur motion depends only on the current location (no memory of the past). Consequently, the description of the current state (location) fully captures all the information that could influence the future evolution of the process.

Definition 2.5.5. Markov chain.

A sequence of random variables $\mathbf{X} = \{X_n, n \geq 0\}$ *taking values in a countable set* $\mathbf{S} = \{1, \ldots, m\}$ *is a* Markov chain *if it satisfies the* Markov property*, i.e. if*

$$P(X_{n+1} = j | X_0 = i_0, X_1 = i_1, \ldots, X_n = i) = P(X_{n+1} = j | X_n = i) \; \forall i, j \in \mathbf{S}, n \geq 0.$$

We note $P_{ij} = P(X_{n+1} = j | X_n = i)$ *and the matrix* P *whose entry* (i, j) *is* P_{ij} *is called the* transition probability matrix *of the Markov chain.* P *is a nonnegative matrix and verifies* $\sum_{j=1}^{m} P_{i,j} = 1 \; \forall i$.

The set $\mathbf{S}$ *is called the* state space *of the Markov chain, and an element of* $\mathbf{S}$ *is called a* state.

The Markov chain is called time homogeneous if

$$P(X_{n+1} = j | X_n = i) = P(X_1 = j | X_0 = i) \; \forall i, j \in \mathbf{S}, n \geq 0.$$

Thus, for a time homogeneous Markov chains, the probability of the transition is independent of n.

We define the *n-step transition probability* as the distribution of the state at some future time, conditioned on the current sate: $r_{ij}(n) = P(X_n = j | X_0 = i)$.

Proposition 2.5.6. Chapman-Kolmogorov equation.

This equation states that:

$$r_{ij}(n) = \sum_{k=1}^{m} r_{ik}(n-1) P_{kj}. \tag{2.51}$$

Proof:

$$\begin{aligned} r_{ij}(n) &= P(X_n = j | X_0 = i) \\ &= \sum_{k=1}^{m} P(X_{n-1} = k | X_0 = i) P(X_n = j | X_{n-1} = k, X_0 = i) \\ &= \sum_{k=1}^{m} P(X_{n-1} = k | X_0 = i) P(X_n = j | X_{n-1} = k) \\ &= \sum_{k=1}^{m} r_{ik}(n-1) P_{kj} \end{aligned}$$

The properties of a Markov chain are determined largely (completely for finite Markov chains) by the 'topology' of their state transition diagram.

Definition 2.5.7. *A state* $j \in \mathbf{S}$ *is* accessible *from* i *if there is some* n *such that* $r_{ij} > 0$*. (It is possible to reach* j *from* i *in* n *steps). We define the set* $A(i)$ *as:*

$$A(i) \doteq \{j \in \mathbf{S} | j \text{ is accessible from } i\}.$$

Definition 2.5.8. *A state* i *is* recurrent *if*

$$j \in A(i) \Rightarrow i \in A(j).$$

In other words, given enough time, we will always return to i.

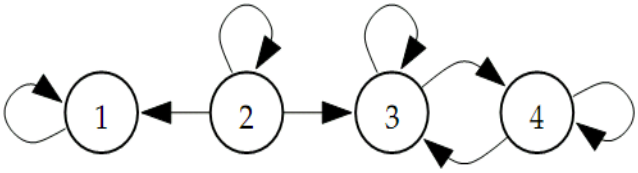

Figure 2.2: Example of Markov chain. States 1,3 and 4 are recurrent, while state 2 is transient.

Definition 2.5.9. *A state i is* transient *if it is not recurrent.*

Definition 2.5.10. *If state i is recurrent, then $A(i)$ forms a* recurrent class. *That is all states in $A(i)$ are accessible from each other and no states outside $A(i)$ are accessible from a state in $A(i)$.*

Figure 2.2 depicts a Markov chain with recurrent and transient states.

A Markov chain can be decomposed into one or more recurrent classes plus some transient states. Once the process enters a recurrent class, it remains in that class forever and all states in the class are visited an infinite number of times.

Definition 2.5.11. *Let R be some recurrent class. R is said to be* periodic *if its states can be grouped into $d > 1$ disjoint subsets $S_1, S_2, \ldots, S_p$ such that all transitions from S_k lead to S_{k+1} and all transitions from S_p lead to S_1.*

A recurrent class that is not periodic is called *aperiodic*.

It is useful to analyze the long-term state occupancy behaviour of a Markov chain, i.e. what is the behaviour of $r_{ij}(n)$ as n becomes large?

Proposition 2.5.12. *For a Markov chain with a single aperiodic recurrent class, we have*

$$\lim_{n\to\infty} r_{ij}(n) = \pi_j \approx P(X_n = j). \tag{2.52}$$

π_j is called the steady-state probability *of j.*

Combining the above equation with the Chapman-Kolmogorov equation give us the *balance equations*:

$$\lim_{n\to\infty} \sum_{k=1}^{m} r_{ik}(n-1)P_{kj} = \pi_j \tag{2.53}$$

$$\sum_{k=1}^{m} \pi_k P_{kj} = \pi_j \tag{2.54}$$

If we consider the balance equations together with the normalization equation $\sum_{k=1}^{m} \pi_k = 1$, we obtain a system of equations which can be solved to get the π_j's.

The steady-state probabilties can be viewed as the *expected state frequencies*. That is, for a Markov chain with a single aperiodic recurrent class,

$$\pi_j = \lim_{n\to\infty} \frac{v_{ij(n)}}{n}, \tag{2.55}$$

where $v_{ij}(n)$ is the expected value of the number of visits to j within the first n transitions, starting from i.

It is also useful to analyze the short-term behaviour of Markov chains. A state k is said to be *absorbing* if $P_{kk} = 1$.

Proposition 2.5.13. Expected time to absorption.
It is the time until we enter some recurrent state. Let

$$\begin{aligned} \mu_i &= \mathbb{E}[\#\ \textit{transitions until a recurrent state is reached}|X_0 = i] \\ &= \mathbb{E}[\min\{n \geq 0|X_n\ \textit{is recurrent}\}|X_0 = i]. \end{aligned}$$

Then we have

$$\mu_i = \begin{cases} 0 & \textit{if } i \textit{ is recurrent} \\ 1 + \sum_{j=1}^m P_{ij}\mu_j & \textit{if } i \textit{ is transient} \end{cases} \tag{2.56}$$

These equations can be solved to find the unique expected time to absorption.

Proposition 2.5.14. Mean first passage time t_i.
It is the time until we reach recurrent state s, starting from i.

$$t_i = \begin{cases} 0 & \textit{if } i = s \\ 1 + \sum_{j=1}^m P_{ij}t_j & \textit{otherwise} \end{cases} \tag{2.57}$$

Proposition 2.5.15. Mean recurrence time t_s^*.
It is the number of transitions up to the first return to s, starting from s.

$$t_s^* = 1 + \sum_{j=1}^m P_{sj}t_j \tag{2.58}$$

Exemple 2.5.1. You flip a coin repeatedly until you get three successive heads. What is the average number of coin flips?

Chapter 3

Optimization

The notes for this part are yet to be written !
See you in the classroom. ☺

Chapter 4

Introduction to Sate Estimation

This chapter provides a brief review of linear and nonlinear estimation methods. We consider the celebrated Kalman Filter (KF) for linear Gaussian dynamic systems, its improvements to deal with nonlinear systems, known as the Extended Kalman Filter (EKF) and the Unscented Kalman Filter (UKF). We finally consider the case of non-Gaussian systems and introduce the Particle Filter (PF).

We assume that the reader already has some knowlegde of the mathematical concepts involved. These include some familiarity with linear algebra and probability ans statistics (Bayes rule, expectation, conditional expectation, etc). So you could refer to the first two chapters of this course.

4.1 Kalman Filter

The Kalman filter (KF), introduced in a 1960 paper by R. E. Kalman [Kal60], provides a recursive solution to the linear optimal filtering problem.

KF is built on the state-space representation of the system. Consider a discrete-time dynamical system governed by the following linear equation

$$x_k = Ax_{k-1} + Bu_{k-1} + w_{k-1}, \tag{4.1}$$

with measurements given by

$$z_k = Hx_k + v_k. \tag{4.2}$$

Equations (4.1) and (4.2) are, respectively, called the *state (model) equation* and the *measurements equation*.

In Eq. (4.1), the state of the system at time k is given by the vector x_k. Deterministic inputs to the system at time $k-1$ are given by u_{k-1}, and random noise affecting the system at time $k-1$ is given by w_{k-1}. w_k is called the *process noise*.

In Eq. (4.2), v_k represents random noise in the observation z_k and is called the *measurement noise*.

The process and measurement noises (w_k and v_k) are assumed to be independent, white and Gaussian with zero mean and covariance matrices defined

by

$$E[w_n w_k^T] = \begin{cases} Q_k & \text{for } n = k \\ 0 & \text{for } n \neq k \end{cases} \tag{4.3}$$

$$E[v_n v_k^T] = \begin{cases} R_k & \text{for } n = k \\ 0 & \text{for } n \neq k \end{cases} \tag{4.4}$$

Furthermore, the initial state x_0, is assumed to be uncorrelated with the process and measurement noises: $x_0 \perp w_k, v_k$.

4.1.1 MMSE derivation

In this formulation, the problem is stated as follows: given the entire set of observations, $z_1, z_2, \ldots, z_k$, find an optimal, in the minimum mean-square error sense, estimate $\hat{x}_k$ of the state x_k.

So the function we want to minimize is

$$J_k = \mathbb{E}[(\tilde{x}_k)^2],$$

where $\tilde{x}_k = x_k - \hat{x_k}$ is the estimation error at time k and $\mathbb{E}[.]$ is the expectation operator.

Recall that KF is recursive. Say that we have found an estimate $\hat{x}_{k-1}$ at time $k-1$. We get a measurement z_k at time k and we want to use this new information to update the estimate of the unknown x_k. Let $\hat{x}_k^-$ denote our a priori estimate of the state, i.e. the estimation obtained using all observations up to and including time $k-1$. This is called the *a priori* estimate of x_k. Similarly, the estimate $\hat{x}_k$ given measurement z_k is called the *a posteriori* estimate.

We can define two errors: a *a priori error* e_k^- and a *a posteriori error* e_k as

$$e_k^- = x_k - \hat{x}_k^- \text{ and } e_k = x_k - \hat{x}_k.$$

We also define the corresponding estimate error covariance matrices:

$$P_k^- = \mathbb{E}[e_k^- e_k^{-T}]. \tag{4.5}$$

$$P_k = \mathbb{E}[e_k e_k^T]. \tag{4.6}$$

One iteration of KF implies two steps:

- **A prediction step**: we predict state x_k from state $\hat{x}_{k-1}$ (i.e. from observations $z_{1:k-1}$). This gives us $\hat{x}_k^-$ and the associated error covariance matrix P_k^-.
- **An update step**: we use observation z_k to update $\hat{x}_k^-$. This gives us $\hat{x}_k$ and the associated error covariance matrix P_k.

Prediction step

In the prediction step, we want to get an estimate of the state at time k from the estimate $\hat{x}_{k-1}$ at time $k-1$. We just use Eq. (4.1) to get $\hat{x}_k^-$:

$$\hat{x}_k^- = A\hat{x}_{k-1} + Bu_{k-1}. \tag{4.7}$$

Note that the process noise has zero mean, so $\mathbb{E}[w_{k-1}] = 0$ do not appear in the last equation.

The a priori estimate error covariance matrix is then

$$P_k^- = \mathbb{E}[(x_k - \hat{x}_k^-)(x_k - \hat{x}_k^-)^T]$$

Replacing $\hat{x}_k^-$ by its expression in Eq. (4.7) and using the fact that the state vector and the process noise are uncorrelated, we get

$$P_k^- = A\mathbb{E}[(x_{k-1} - \hat{x}_{k-1})(x_{k-1} - \hat{x}_{k-1})^T]A^T + \mathbb{E}[w_{k-1}w_{k-1}^T],$$

which gives the following equation

$$P_k^- = AP_{k-1}A^T + Q_{k-1}. \tag{4.8}$$

Update step

In the update step, we use measurement z_k to update the a priori estimate $\hat{x}_k^-$. The idea is to write the a posteriori estimate $\hat{x}_k$ as a linear combination of $\hat{x}_k^-$ and z_k:

$$\hat{x_k} = \hat{x}_k^- + K_k(z_k - H\hat{x}_k^-). \tag{4.9}$$

How does this formulation make sense? First, since we are looking for a linear estimator it makes sense to write it as a linear combination. But I assume you are not totally convinced by that!

The term $(z_k - H\hat{x}_k^-)$ in Eq. (4.9) is called the measurement *innovation* or the *residual*. It is in fact, the difference between the a priori estimate of the measurement $H\hat{x}_k^-$ and the actual measurement z_k. So if the residual is zero, i.e. if the predicted value of the measurement is exactly the observed value, then there is no need to update the estimate. Hence, we have $\hat{x}_k = \hat{x}_k^-$.

On the contrary, if the residual is not zero, then we have to combine the a priori estimate $\hat{x}_k^-$ and the residual $(z_k - H\hat{x}_k^-)$ in such a way that the a posteriori error is minimized. This is why we have that factor K_k called the *Kalman gain*.

The optimal Kalman gain K_k is obtained by minimizing the a posteriori error. It can be shown that the error is minimum when the trace of the error covariance matrix is minumum. The a posteriori error covariance matrix is

$$P_k = \mathbb{E}[e_k e_k^T] = \mathbb{E}[(x_k - \hat{x}_k)(x_k - \hat{x}_k)^T],$$

which after a few calculations gives

$$P_k = (I - K_kH)P_k^-(I - K_kH)^T + K_kRK_k^T.$$

Taking the trace of P_k and setting its derivative w.r.t. K_k equals to zero, we obtain the optimal Kalman gain

$$K_k = P_k^- H^T(HP_k^- H^T + R_k)^{-1}. \tag{4.10}$$

Using the optimal Kalman gain given by Eq. (4.10), the expression of the a posteriori covariance matrix can be simplified as

$$P_k = (I - K_kH)P_k^-. \tag{4.11}$$

Now that we have an estimate of the system state at time k, $\hat{x}_k$, we can use it to predict the state at time $k+1$; this gives us $\hat{x}_{k+1}^-$. With a new measurement z_{k+1} at time $k+1$, we update the prediction to get the state estimate $\hat{x}_{k+1}$. And so forth ... recursivity!

We may summarize the discrete KF by the following algorithm:

---

1. Start: $\hat{x}_0^-$, P_0^-

2. Compute Kalman gain:

$$K_k = P_k^- H^T (H P_k^- H^T + R_k)^{-1}$$

3. Compute state estimate:

$$\hat{x_k} = \hat{x}_k^- + K_k(z_k - H\hat{x}_k^-)$$

4. Update error covariance:

$$P_k = (I - K_k H) P_k^-$$

5. Update prediction and predicted error covariance:

$$\hat{x}_{k+1}^- = A\hat{x}_k + Bu_k$$

$$P_{k+1}^- = AP_kA^T + Q_k$$

6. Do $k = k + 1$ and go to step 2.

4.1.2 Bayesian derivation

In this section, we describe the KF as a Bayesian estimation problem. This formulation will be used later on these notes.

In the Bayesian approach to dynamic state estimation, we construct the posterior probability density function (pdf) of the state based on all available observations. Basically, we use the Bayes theorem (but not only that!).

Let us consider the evolution, in time, of the state sequence $\{x_k, k \in \mathbb{N}\}$ of a system given by

$$x_k = f_k(x_{k-1}, u_{k-1}) + w_{k-1}, \tag{4.12}$$

where $f_k : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^n$ is a function, possibly nonlinear, of the state, $\{u_{k-1}, k \in \mathbb{N}\}$ is the input sequence, and $\{w_{k-1}, k \in \mathbb{N}\}$ is an i.i.d. process noise sequence.

The sequence of measurements is given by

$$z_k = h_k(x_k) + v_k, \tag{4.13}$$

where $h_k : \mathbb{R}^n \times \mathbb{R}^l \to \mathbb{R}^p$ is a function, possibly nonlinear, and $\{v_k, k \in \mathbb{N}\}$ is an i.i.d. measurement noise sequence.

From a Bayesian perspective, our goal is to obtain the pdf $p(x_k|z_{1:k})$, i.e. we want to estimate the conditional pdf of the state x_k at time k given all observations up to time k: $z_{1:k} = \{z_i, i = 1, \ldots, k\}$.

If we assume that the initial pdf $p(x_0|z_0) \equiv p(x_0)$ is known a priori, then the pdf $p(x_k|z_{1:k})$ can be obtained in two stages: prediction and update.

Prediction

Suppose that the pdf $p(x_{k-1}|z_{1:k-1})$ at time $k-1$ is available. Then, using the system model equation (4.12), we get the prior pdf of the state a time k via the Chapman-Kolmogorov equation:

$$p(x_k|z_{1:k-1}) = \int p(x_k|x_{k-1})p(x_{k-1}|z_{1:k-1})dx_{k-1}. \tag{4.14}$$

Note that we have used the fact that Eq. (4.12) describes a first order Markov process, which implies that $p(x_k|x_{k-1}, z_{1:k-1}) = p(x_k|x_{k-1})$.

Update

At step k, we use the available measurement z_k to update the prior pdf via Bayes' rule

$$p(x_k|z_{1:k}) = \frac{p(z_k|x_k)p(x_k|z_{1:k-1})}{p(z_k|z_{1:k-1})}, \tag{4.15}$$

where the normalization constant

$$p(z_k|z_{1:k-1}) = \int p(z_k|x_k)p(x_k|z_{1:k-1})dx_k,$$

depends on the likelihood function $p(z_k|x_k)$ (defined by the measurement equation (4.13)) and the known statistics of v_k.

Let us now assume that the process and measurement noises are governed by Gaussian distributions of known parameters, i.e. $w_{k-1} \sim \mathcal{N}(0, Q_{k-1})$ and $v_k \sim \mathcal{N}(0, R_k)$. Furthermore, we assume that they are independent.

We also assume that the functions $f_k(x_{k-1}, u_{k-1})$ and $h_k(x_k)$ are linear functions represented, respectively, by the matrices A_k and H_k. Then, the state-space representation of the system given by Eq. (4.12) and (4.13), can be written as

$$x_k = A_k x_{k-1} + B_k u_{k-1} + w_{k-1}. \tag{4.16}$$

$$z_k = H_k x_k + v_k. \tag{4.17}$$

You should already be familiar with the last two equations!

Finally, using the prediction and update equations, Eq.(4.14) and (4.15), we obtain the recursive equations of the Kalman filter:

$$\begin{aligned} p(x_{k-1}|z_{1:k-1}) &\sim \mathcal{N}(m_{k-1|k-1}, P_{k-1|k-1}) && (4.18)\\ p(x_k|z_{1:k-1}) &\sim \mathcal{N}(m_{k|k-1}, P_{k|k-1}) && (4.19)\\ p(x_k|z_{1:k}) &\sim \mathcal{N}(m_{k|k}, P_{k|k}) && (4.20) \end{aligned}$$

where

$$\begin{aligned} m_{k|k-1} &= A_k m_{k-1|k-1} && (4.21)\\ P_{k|k-1} &= A_k P_{k-1|k-1} A_k^T + Q_{k-1} && (4.22)\\ m_{k|k} &= m_{k|k-1} + K_k(z_k - H_k m_{k|k-1}) && (4.23)\\ P_{k|k} &= (I - K_k H_k) P_{k|k-1} && (4.24) \end{aligned}$$

and where K_k is the Kalman gain given by

$$K_k = P_{k|k-1} H_k^T (H_k P_{k|k-1} H_k^T + R_k)^{-1}. \tag{4.25}$$

Note that these equations are totally equivalent to those given in the previous section! In the Bayesian framework, we recursively predict and update the mean $m_{k|k}$ and the covariance matrix $P_{k|k}$ of the pdf.

4.2 Extended Kalman Filter

Up to this point, we have considered that the system can be described by linear equations (the state-space equations). But in many practical situations, one or both of the state equation and the measurements equation may be nonlinear. Recall that the functions f_k and h_k in Eq. (4.12) and (4.13) may be nonlinear. In such a case, the solution consists in a local linearization of the equations. The Extended Kalman Filter (EKF) is based on this approximation.

Say we have the following state and measurement equations

$$x_k = f_k(x_{k-1}, u_{k-1}) + w_{k-1},$$

and

$$z_k = h_k(x_k) + v_k.$$

Assume that an estimate $\hat{x}_{k-1}$ has been obtained at time $k-1$. We may predict the state estimate at time k as

$$\hat{x}_k^- = f_k(\hat{x}_{k-1}, u_{k-1}). \tag{4.26}$$

The true state vector is related to the a priori estimated one by

$$x_k = \hat{x}_k^- + \Delta x_k.$$

Then, we linearize the state equation around $\hat{x}_{k-1}$, using a Taylor series expansion of order 1 and get

$$x_k = \hat{x}_k^- + \Delta x_k = f_k(\hat{x}_{k-1}, u_{k-1}) + \frac{\partial f_k}{\partial x}\Big|_{\hat{x}_{k-1}} \Delta x_{k-1} + w_{k-1}.$$

Replacing $\hat{x}_k^-$ by its expression in Eq. (4.26), we obtain

$$\Delta x_k = \frac{\partial f_k}{\partial x}\Big|_{\hat{x}_{k-1}} \Delta x_{k-1} + w_{k-1}. \tag{4.27}$$

If we perform a similar linearization of the measurement equation around $\hat{x}_k^-$, we obtain

$$z_k - h_k(\hat{x}_k^-) = \frac{\partial h_k}{\partial x}\Big|_{\hat{x}_k^-} \Delta x_k + v_k. \tag{4.28}$$

The last two equations can be rewritten as

$$\Delta x_k = F_k \Delta x_{k-1} + w_{k-1}, \tag{4.29}$$

and

$$z_k - h_k(\hat{x}_k^-) = H_k \Delta x_k + v_k, \tag{4.30}$$

where

$$F_k = \frac{\partial f_k}{\partial x}\Big|_{\hat{x}_{k-1}},$$

and

$$H_k = \frac{\partial h_k}{\partial x}\Big|_{\hat{x}_k^-}.$$

Equations (4.29) and (4.30) are the state space representation of a linear system, with state vector Δx and observation $\tilde{z} = z - h(\hat{x})$.

We can now use the conventional KF to solve this 'new' system and we obtain the following prediction and update equations:

Prediction

$$\hat{x}_k^- = f_k(\hat{x}_{k-1}, u_{k-1}). \tag{4.31}$$

$$P_k^- = F_k P_{k-1} F_k^T + Q_{k-1}. \tag{4.32}$$

Update

$$K_k = P_k^- H_k^T (H_k P_k^- H_k^T + R_k)^{-1}. \tag{4.33}$$

$$\hat{x}_k = \hat{x}_k^- + K_k(z_k - h_k(\hat{x}_k^-)). \tag{4.34}$$

$$P_k = (I - K_k H_k) P_k^-. \tag{4.35}$$

Note that for the EKF, the matrices F_k and H_k depend on the observations (linearization around estimated points) and must be re-calculated at each time k.

Finally, there are two main issues to consider when using the EKF:

- *linearization* can leads to highly unstable filters if the assumption of local linearity is violated;
- computation of the Jacobian matrices (F_k and H_k) may be nontrivial depending on the function f_k and h_k.

4.3 Unscented Kalman Filter

As mentioned in the previous section, the EKF linearizes all the nonlinear models so that the conventional KF can be used. This linearization implies the computation of Jacobian matrices, which may be difficult. Furthermore, if the local linearity assumption is not met, the filter may diverge. The unscented Kalman filter (UKF), first proposed by Julier and Uhlmann [JU97], deals with these two drawbacks.

The main idea of UKF is the use of a transformation, the *unscented transformation*, that appropriately approximates the nonlinearity of the distributions.

4.3.1 The unscented transform

The unscented transform is a method for calculating the statistics of a random variable which undergoes a nonlinear transformation. As mentioned by Julier and Uhlmann, it is based on the intution that it is easier to approximate a Gaussian distribution than it is to approximate an arbitrary nonlinear function or transformation [JU97].

Let $\mathbf{x}$ be a random variable with mean $\overline{\mathbf{x}}$ and covariance $\mathbf{P}_{xx}$. We define another random variable $\mathbf{y}$ related to $\mathbf{x}$ through a nonlinear function

$$\mathbf{y} = f(\mathbf{x}).$$

Say we want to calculate the mean $\overline{\mathbf{y}}$ and covariance $\mathbf{P}_{yy}$ of $\mathbf{y}$. The unscented transformation works as follows:

- Choose a set a points, called *sigma points*, so that their sample mean and sample covariance are $\overline{\mathbf{x}}$ and $\mathbf{P}_{xx}$;
- Apply the nonlinear transformation f to each point. The statistics of the transformed points give an approximation of $\overline{\mathbf{y}}$ and $\mathbf{P}_{yy}$.

To calculate the statistics of $\mathbf{y}$, we approximate the n-dimensional random vector $\mathbf{x}$ by $2n+1$ sigma points $\mathcal{X}_i$ given by:

$$\mathcal{X}_0 = \overline{\mathbf{x}}. \tag{4.36}$$

$$\mathcal{X}_i = \overline{\mathbf{x}} + \left(\sqrt{(n+\kappa)\mathbf{P}_{xx}}\right)_i \qquad i = 1, \ldots, n. \tag{4.37}$$

$$\mathcal{X}_i = \overline{\mathbf{x}} - \left(\sqrt{(n+\kappa)\mathbf{P}_{xx}}\right)_{i-n} \qquad i = n+1, \ldots, 2n. \tag{4.38}$$

where $\kappa \in \mathbb{R}$ is usually set to 0 or $3-n$, and $\left(\sqrt{(n+\kappa)\mathbf{P}_{xx}}\right)_i$ is the ith column of the matrix square root of $(n+\kappa)\mathbf{P}_{xx}$ (e.g., lower triangular Cholesky factorization).

These sigma points are propagated trhough the nonlinear function f,

$$\mathcal{Y}_i = f(\mathcal{X}_i)\ i = 0, \ldots, 2n, \tag{4.39}$$

and the mean and covariance of $\mathbf{y}$ are approximated by the weighted sample mean and covariance of the transformed points,

$$\overline{\mathbf{y}} \approx \sum_{i=0}^{2n} W_i \mathcal{Y}_i. \tag{4.40}$$

$$\mathbf{P}_{yy} \approx \sum_{i=0}^{2n} W_i \{\mathcal{Y}_i - \overline{\mathbf{y}}\}\{\mathcal{Y}_i - \overline{\mathbf{y}}\}^T. \tag{4.41}$$

with weights W_i given by

$$W_0 = \kappa/(n+\kappa). \tag{4.42}$$

$$W_i = 1/\{2(n+\kappa)\}\ i = 1, \ldots, 2n. \tag{4.43}$$

4.3.2 The unscented KF

In order to used the unscented transformation as a recursive filter, we need to reconstruct the sate vector as a concatenation of the original state vector and the process noise vector. This gives an $n^a = n + q$ dimensional vector $x_k^a = [x_k \; w_k]^T$.

The state equation is rewritten as a function of $x^a(k)$,

$$x_k = f(x_{k-1}^a, u_{k-1}),$$

and the unscented transform uses $2n^a + 1$ sigma points which are drawn from

$$\hat{x}_{k|k}^a = \begin{pmatrix} \hat{x}_{k|k} \\ 0 \end{pmatrix} \qquad \text{and} \qquad \mathbf{P}_{k|k}^a = \begin{bmatrix} \mathbf{P}_{k|k} & 0 \\ 0 & \mathbf{Q}_k \end{bmatrix}.$$

The algorithm of the UKF is as follows:

---

1. Initialize with
$$\hat{X}_0 = \mathbb{E}[x_0]$$
$$\mathbf{P}_0 = \mathbb{E}[(x_0 - \hat{x}_0)(x_0 - \hat{x}_0)^T]$$
2. For $k \in \{1, \ldots, \infty\}$, calculate sigma points
$$\mathcal{X}_{k-1} = [\hat{x}_{k-1} \qquad \hat{x}_{k-1} + \gamma\sqrt{\mathbf{P}_{k-1}} \qquad \hat{x}_{k-1} - \gamma\sqrt{\mathbf{P}_{k-1}}]$$
3. Prediction
$$\begin{aligned}
\mathcal{X}_{k|k-1} &= f[\mathcal{X}_{k-1}, u_{k-1}] \\
\hat{x}_k^- &= \sum_{i=0}^{2n^a} W_i \mathcal{X}_{i,k|k-1} \\
\mathbf{P}_k^- &= \sum_{i=0}^{2n^a} W_i [\mathcal{X}_{i,k|k-1} - \hat{x}_k^-][\mathcal{X}_{i,k|k-1} - \hat{x}_k^-]^T + \mathbf{Q}_k \\
\mathcal{Z}_{k|k-1} &= h(\mathcal{X}_{k|k-1}) \\
\hat{z}_k^- &= \sum_{i=0}^{2n^a} W_i \mathcal{Z}_{i,k|k-1}
\end{aligned}$$
4. Update
$$\begin{aligned}
\mathbf{P}_{\tilde{z}_k \tilde{z}_k} &= \sum_{i=0}^{2n^a} W_i [\mathcal{Z}_{i,k|k-1} - \hat{z}_k^-][\mathcal{Z}_{i,k|k-1} - \hat{z}_k^-]^T + \mathbf{R}_k \\
\mathbf{P}_{x_k z_k} &= \sum_{i=0}^{2n^a} W_i [\mathcal{X}_{i,k|k-1} - \hat{x}_k^-][\mathcal{Z}_{i,k|k-1} - \hat{z}_k^-]^T \\
K_k &= \mathbf{P}_{x_k z_k} \mathbf{P}_{\tilde{z}_k \tilde{z}_k}^{-1} \\
\hat{x}_k &= \hat{x}_k^- + K_k(z_k - \hat{z}_k^-) \\
\mathbf{P}_k &= \mathbf{P}_k^- - K_k \mathbf{P}_{\tilde{z}_k \tilde{z}_k} K_k^T
\end{aligned}$$

where $\gamma = \sqrt{n + \kappa}$ and $\tilde{z}_k = z_k - \hat{z}_k^-$.

The UKF have been proven to achieve better performance than the EKF for nonlinear systems. Note that no explicit calculations of Jacobian matrices are necessary to implement the algorithm.

4.4 Particle Filter

The particle filter (PF) is a sequential Monte Carlo (MC) method used for Bayesian filtering. The basic idea is that randomly chosen weighted samples, called *particles*, are used to approximate a probability distribution function.

Let us consider a system governed by the following state-space representation

$$x_k = f_k(x_{k-1}, u_{k-1}) + w_{k-1},$$

and

$$z_k = h_k(x_k) + v_k.$$

As shown in section 4.1.2, the optimal Bayesian estimation problem is solved by a recursive prediction/update scheme:

- prediction

$$p(x_k|z_{1:k-1}) = \int p(x_k|x_{k-1})p(x_{k-1}|z_{1:k-1})dx_{k-1}.$$

- update

$$p(x_k|z_{1:k}) = \frac{p(z_k|x_k)p(x_k|z_{1:k-1})}{p(z_k|z_{1:k-1})}.$$

However, this recursive propagation of the posterior pdf cannot be used in practice because it cannot be determined analytically (requires solving the integrals!). So, we need approximate methods such as PF. Note that, EKF and UKF are also approximate methods, but they do rely on the assumption of Gaussian distribution (they propagate only the first two moments of the distribution). PF on the contrary, can deal with any type of distribution.

4.4.1 Sequential Importance Sampling (SIS)

The key idea of PF is to represent the posterior pdf by a set of random samples with associated weights. The samples or particles are randomly chosen using the sequential importance sampling (SIS) algorithm. SIS is a Monte Carlo (MC) method that forms the basis for most sequential MC filters.

Let $\{\mathtt{x}^i_{0:k}, i = 0, \ldots, N_s\}$ be a set of support points (samples or particles) and $\{w^i_k, i = 1, \ldots, N_s\}$ be the set of associated weights. The weights are normalized such that $\sum_i w^i_k = 1$. We note $\mathtt{x}_{0:k} = \{\mathtt{x}_j, j = 0, \ldots, k\}$ the set of all states up to time k. Then, the posterior pdf at time k can be approximated as

$$p(\mathtt{x}_{0:k}|\mathtt{z}_{1:k}) \approx \sum_{i=1}^{N_s} w^i_k \delta(\mathtt{x}_{0:k} - \mathtt{x}^i_{0:k}). \tag{4.44}$$

This gives us a discrete weighted approximation of the true posterior pdf. As the number of samples increases, it can be shown that the discrete approximation converges (almost surely) to the true pdf.

Given such an approximation, it is also easy to approximate the moments of the distribution; e.g., the expectation is approximated by

$$\mathbb{E}[g(\mathbf{x}_{0:k})] = \int g(\mathbf{x}_{0:k}) p(\mathbf{x}_{0:k}|\mathbf{z}_{1:k}) d\mathbf{x}_{0:k} \approx \sum_{i=1}^{N_s} w_k^i g(\mathbf{x}_{0:k}^i).$$

Unfortunately, it is often not possible to sample directly from the posterior distribution. So we draw the samples form another, easy to sample, proposal distribution $\mathbf{q}(\mathbf{x}_{0:k}|\mathbf{z}_{1:k})$, called the *importance density*. We can then approximate the expectation by

$$\begin{aligned}
\mathbb{E}[g(\mathbf{x}_{0:k})] &= \int g(\mathbf{x}_{0:k}) \frac{p(\mathbf{x}_{0:k}|\mathbf{z}_{1:k})}{q(\mathbf{x}_{0:k}|\mathbf{z}_{1:k})} q(\mathbf{x}_{0:k}|\mathbf{z}_{1:k}) d\mathbf{x}_{0:k}, \\
&= \int g(\mathbf{x}_{0:k}) \frac{p(\mathbf{z}_{1:k}|\mathbf{x}_{0:k}) p(\mathbf{x}_{0:k})}{p(\mathbf{z}_{1:k}) q(\mathbf{x}_{0:k}|\mathbf{z}_{1:k})} q(\mathbf{x}_{0:k}|\mathbf{z}_{1:k}) d\mathbf{x}_{0:k}, \\
&= \int g(\mathbf{x}_{0:k}) \frac{w_k(\mathbf{x}_{0:k})}{p(\mathbf{z}_{1:k})} q(\mathbf{x}_{0:k}|\mathbf{z}_{1:k}) d\mathbf{x}_{0:k},
\end{aligned}$$

where $w_k(\mathbf{x}_{0:k})$ are the unnormalized importance weights:

$$w_k(\mathbf{x}_{0:k}) = \frac{p(\mathbf{z}_{1:k}|\mathbf{x}_{0:k}) p(\mathbf{x}_{0:k})}{q(\mathbf{x}_{0:k}|\mathbf{z}_{1:k})}. \tag{4.45}$$

Note that the weights can also be writen as

$$w_k^i = w_k(\mathbf{x}_{0:k}^i) \propto \frac{p(\mathbf{x}_{0:k}^i|\mathbf{z}_{1:k})}{q(\mathbf{x}_{0:k}^i|\mathbf{z}_{1:k})}. \tag{4.46}$$

If the importance density is chosen to factorize such that

$$q(\mathbf{x}_{0:k}|\mathbf{z}_{1:k}) = q(\mathbf{x}_k|\mathbf{x}_{0:k-1}, \mathbf{z}_{1:k}) q(\mathbf{x}_{0:k-1}|\mathbf{z}_{1:k-1}),$$

then one can augment old particles $\mathbf{x}_{0:k-1}^i \sim q(\mathbf{x}_{0:k-1}|\mathbf{z}_{1:k-1})$ by the new state $\mathbf{x}_k^i \sim q(\mathbf{x}_k|\mathbf{x}_{0:k-1}, \mathbf{z}_{1:k})$ to obtain the new particles $\mathbf{x}_{0:k}^i \sim q(\mathbf{x}_{0:k}|\mathbf{z}_{1:k})$.

The weights can also be updated as

$$w_k^i = w_{k-1}^i \frac{p(\mathbf{z}_k|\mathbf{x}_k^i) p(\mathbf{x}_k^i|\mathbf{x}_{k-1}^i)}{q(\mathbf{x}_k^i|\mathbf{x}_{0:k-1}^i, \mathbf{z}_{1:k})}. \tag{4.47}$$

Furthermore, if $q(\mathbf{x}_k|\mathbf{x}_{0:k-1}, \mathbf{z}_{1:k}) = q(\mathbf{x}_k|\mathbf{x}_{k-1}, \mathbf{z}_{1:k})$, then the importance density becomes only dependent on the last state $\mathbf{x}_{k-1}$ and the observation $\mathbf{z}_k$. In such a case, we need only to store $\mathbf{x}_k^i$ and not all the trajectory $\mathbf{x}_{0:k-1}^i$ or all the history of observations $\mathbf{z}_{1:k-1}$. In this case, the weights are given by

$$w_k^i \propto w_{k-1}^i \frac{p(\mathbf{z}_k|\mathbf{x}_k^i) p(\mathbf{x}_k^i|\mathbf{x}_{k-1}^i)}{q(\mathbf{x}_k^i|\mathbf{x}_{k-1}^i, \mathbf{z}_k)}, \tag{4.48}$$

and the posterior pdf can be approximated by

$$p(\mathbf{x}_k|\mathbf{z}_{1:k}) \approx \sum_{i=1}^{N_s} w_k^i \delta(\mathbf{x}_k - \mathbf{x}_k^i). \tag{4.49}$$

The SIS algorithm thus consists in recursively propagating the weights and support points as each measurement is received sequentially.

A pseudo-code is given bellow.

ALGORITHM 1: SIS Particle Filter

$[\{\mathbf{x}_k^i, w_k^i\}_{i=1}^{N_s}] = \text{SIS } [\{\mathbf{x}_{k-1}^i, w_{k-1}^i\}_{i=1}^{N_s}, \mathbf{z}_k]$

- FOR $i = 1 : N_s$
  - Draw $\mathbf{x}_k^i \sim q(\mathbf{x}_k|\mathbf{x}_{k-1}^i, \mathbf{z}_k)$
  - Update weights according to Eq. (4.48)
- END FOR
- Normalize weights to $\sum_{i=1}^{N_s} w_k^i = 1$

A common problem with the SIS algorithm is knwon as the *degeneracy problem*. That is, after a few iterations, most particles have negligible weights; the weights are concentrated on a few particles only!

A real measure of degeneracy is hard to compute, but a good estimate can be obtained by

$$\widehat{N_{eff}} = \frac{1}{\sum_{i=1}^{N_s}(w_k^i)^2}. \tag{4.50}$$

Notice that $\widehat{N_{eff}} \leq N_s$, and small value of $\widehat{N_{eff}}$ indicates severe degeneracy.

To overcome the degeneracy problem, one can adopt different strategies:

1. **Brute force:** many, many samples! This is often impractical.

2. **Good choice of importance density** $q(.)$:
   It can be shown, see for example [DGK00], that the optimal importance density is given by

   $$q(\mathbf{x}_k|\mathbf{x}_{k-1}, \mathbf{z}_k)_{opt} = p(\mathbf{x}_k|\mathbf{x}_{k-1}, \mathbf{z}_k).$$

   The weights are then given by

   $$w_k^i = w_{k-1}^i \int p(\mathbf{z}_k|\mathbf{x}_k')p(\mathbf{x}_k'|\mathbf{x}_{k-1}^i)d\mathbf{x}_k'.$$

   This optimal importance density requires to sample from $p(\mathbf{x}_k|\mathbf{x}_{k-1}^i, \mathbf{z}_k)$ and to evaluate an integral over the entire state. Both operations can rarely be done.

   An alternative choice which is often convenient is to take

   $$q(\mathbf{x}_k|\mathbf{x}_{k-1}^i, \mathbf{z}_k) = p(\mathbf{x}_k|\mathbf{x}_{k-1}^i).$$

   This second choice is easy to implement, but does not take measurements into account.

3. **Resampling**
   The basic idea of resampling is to eliminate particles with small associated weights and to concentrate on particles with high weights. This is done by generating a new set of samples $\{\mathbf{x}_k^{*i}, i = 1, \ldots, N_s\}$, by resampling N_s times from an approximate discrete representation of $p(\mathbf{x}_k|\mathbf{z}_{1:k})$ given by

$$p(\mathbf{x}_k|\mathbf{z}_{1:k}) \approx \sum_{i=1}^{N_s} w_k^i \delta(\mathbf{x}_k - \mathbf{x}_k^i),$$

   so that $Pr(\mathbf{x}_k^{*i} = \mathbf{x}_k^i) = w_k^i$.

   If resampling reduces degeneracy, it also introduces other problems. Particularly, samples with high weights are selected more and more often, while others are never selected. This is known as the *sample impoverishment* problem. Methods exist to solve this problem, which we do not discuss here.

4.4.2 Generic Particle Filter

Here we give a pseudo-code of a generic PF algorithm.

ALGORITHM 2: Generic Particle Filter

$[\{\mathbf{x}_k^i, w_k^i\}_{i=1}^{N_s}] = \text{PF } [\{\mathbf{x}_{k-1}^i, w_{k-1}^i\}_{i=1}^{N_s}, \mathbf{z}_k]$

- FOR $i = 1 : N_s$
  - Draw $\mathbf{x}_k^i \sim q(\mathbf{x}_k|\mathbf{x}_{k-1}^i, \mathbf{z}_k)$
  - Update weights according to Eq. (4.48)
- END FOR
- Normalize weights to $\sum_{i=1}^{N_s} w_k^i = 1$
- Calculate degeneracy measure $\widehat{N_{eff}}$ using Eq. (4.50)
- IF $\widehat{N_{eff}} < N_T$
  - Resample
- END IF

Note that there are various versions of particle filters that have been developped, and they can be viewed as special cases of the general SIS algorithm presented in previous subsection [AMGC02]. These include:

- sampling importance resampling (SIR) filter
- auxiliary sampling importance resampling (ASIR) filter
- regularized particle filter (RPF)

Appendix A

Notes about linear regression

In this section, we review different approaches for solving the linear regression problem.

A.1 Introduction

Say we consider a group of N students, and each student is described by p independent variables. For example, a student may be described by his age (x_1), his gender (x_2), his weight (x_3), etc. Let $y_1, \ldots, y_N$ be the test results of the students.

We call the the data set of N pairs $\{x_i, y_i\}_{i=1,\ldots,N}$ the *training set*.

Note that each x_i is a p-dimensional vector: $x_i = [x_{i1}, \cdots, x_{ip}]^T$.

A.1.1 The problem

What we would like to do, is to express each y_i as a linear combination of the p independent variables, with minimum approximation error. That is, we would like to predict the test score y_i of a student given its description by the set of p variables $x_i = [x_{i1}, \cdots, x_{ip}]^T$.

More formally, we want to express each y_i as:

$$\begin{aligned} y_1 &= \phi_0 + \phi_1 x_{11} + \phi_2 x_{12} + \cdots + \phi_p x_{1p} + e_1 \\ y_2 &= \phi_0 + \phi_1 x_{21} + \phi_2 x_{22} + \cdots + \phi_p x_{2p} + e_2 \\ \vdots &= \vdots \\ y_N &= \phi_0 + \phi_1 x_{N1} + \phi_2 x_{N2} + \cdots + \phi_p x_{Np} + e_N \end{aligned}$$

where the values ϕ_i, $i = 1, \ldots, p$, are the coefficients of the linear approximation, and the variables e_i represent the approximation error, or residual.

We can reformulate the problem in terms of matrices as follows:

- Form a matrix X that contains in each row the components of the x_i vectors (including the constant 1 in the first column to account for the

bias ϕ_0):

$$X = \begin{bmatrix} 1 & x_{11} & x_{12} & \cdots & x_{1p} \\ 1 & x_{21} & x_{22} & \cdots & x_{2p} \\ \vdots & \vdots & \ddots & \vdots & \\ 1 & x_{N1} & x_{N2} & \cdots & x_{Np} \end{bmatrix},$$

and a vector $y^T = [y_1, \ldots, y_N]$.

- Form a vector $\Phi^T = [\phi_1, \ldots, \phi_p]$ containing the coefficients of the linear approximation, and a vector $e^T = [e_1, \ldots, e_N]$ containing the individual approximation errors.
- Then, we want to find the vector of coefficients Φ such that

$$y = X\Phi + e. \tag{A.1}$$

Since, we want the approximation to be minimum, we are looking for Φ such that the sum of squared errors (or sum of squared residuals) is minimum, i.e.

$$\begin{aligned} \Phi^* &= \min_{\Phi} E = e^T e && \text{(A.2)} \\ &= \min_{\Phi} (y - X\Phi)^T (y - X\Phi) && \text{(A.3)} \\ &= \min_{\Phi} \sum_{i=1}^{N} (y_i - \phi_0 + \phi_1 x_{i1} - \cdots - \phi_p x_{ip})^2 && \text{(A.4)} \end{aligned}$$

NOTE
Linear regression is therefore a MMSE (minimum mean squared error) estimation problem.

A.2 Different approaches

As mentionned above, there are several ways for solving the linear regression problem. Here, we review some of them.

A.2.1 Solution by partial derivatives

A direct approach is to compute all partial derivatives of E w.r.t. $\phi_0, \phi_1, \ldots, \phi_p$, and set the results to zero. This gives

$$\begin{aligned} \frac{\partial E}{\partial \phi_0} &= -2 \sum_{i=1}^{N} (y_i - \phi_0 - \phi_1 x_{i1} - \cdots - \phi_p x_{ip}) = 0 \\ \frac{\partial E}{\partial \phi_1} &= -2 \sum_{i=1}^{N} (y_i - \phi_0 - \phi_1 x_{i1} - \cdots - \phi_p x_{ip}) x_{i1} = 0 \\ \vdots &= \vdots \\ \frac{\partial E}{\partial \phi_p} &= -2 \sum_{i=1}^{N} (y_i - \phi_0 - \phi_1 x_{i1} - \cdots - \phi_p x_{ip}) x_{ip} = 0 \end{aligned}$$

Re-arranging this, we obtain a set of $p+1$ linear equations for $p+1$ variables:

$$\begin{array}{rl} \sum y_i & = \phi_0 \sum 1 + \phi_1 \sum x_{i1} + \cdots + \phi_p \sum x_{ip} \\ \sum y_i x_{i1} & = \phi_0 \sum x_{i1} + \phi_1 \sum x_{i1}^2 + \cdots + \phi_p \sum x_{ip} x_{i1} \\ \vdots & = \quad \vdots \\ \sum y_i x_{ip} & = \phi_0 \sum x_{ip} + \phi_1 \sum x_{i1} x_{ip} + \cdots + \phi_p \sum x_{ip}^2 \end{array}$$

These $p+1$ equations can be written in terms of matrix as follows:

$$X^T y = X^T X \Phi. \tag{A.5}$$

If the inverse $(X^T X)^{-1}$ exists, then the solution for Φ is given by

$$\Phi = (X^T X)^{-1} X^T \Phi. \tag{A.6}$$

A.2.2 Solution by vector derivatives

If we know the derivation rules for vectors (see appendix B), we can obtain the above result in a much faster way.

Indeed, we have

$$\begin{array}{rcl} E & = & (y - X\Phi)^T (y - X\Phi) \\ & = & y^T y - y^T X \Phi - \Phi^T X^T y + \Phi^T X^T X \Phi \end{array}$$

Taking the derivative of E w.r.t. vector Φ, we get

$$\frac{dE}{d\Phi} = 0 - 2X^T y + 2X^T X \Phi = 0,$$

where we have used the fact that $y^T y$ is a constant, and $y^T X \Phi = \Phi^T X^T y$.

Therefore, we have

$$X^T y = X^T X \Phi,$$

and, if the inverse $(X^T X)^{-1}$ exists, we get

$$\Phi = (X^T X)^{-1} X^T \Phi.$$

A.2.3 Solution by normal projection

Let consider the matrix formulation

$$y = X\Phi + e.$$

We want to write the vector y as linear combination of columns of the matrix X (the coefficients of the combination being given by Φ), such that the error e is minimal. The error is minimum when the distance to the linear subspace defined by $X\Phi$ is minimum, which is the case when considering the normal projection onto the subspace spanned by the columns of X.

Note that we have $e = y - X\Phi$. And, since e is orthogaonal to the column space $C(X)$ of matrix X, we know $e \in N(X^T)$ (see the fundamental theorem of linear algebra in Section 1.4.1). Therefore, we have

$$X^T e = 0 \Rightarrow \; X^T (y - X\Phi) = 0 \Rightarrow \; \Phi = (X^T X)^{-1} X^T y.$$

This projection approach is reprensented by the diagram of figure A.1.

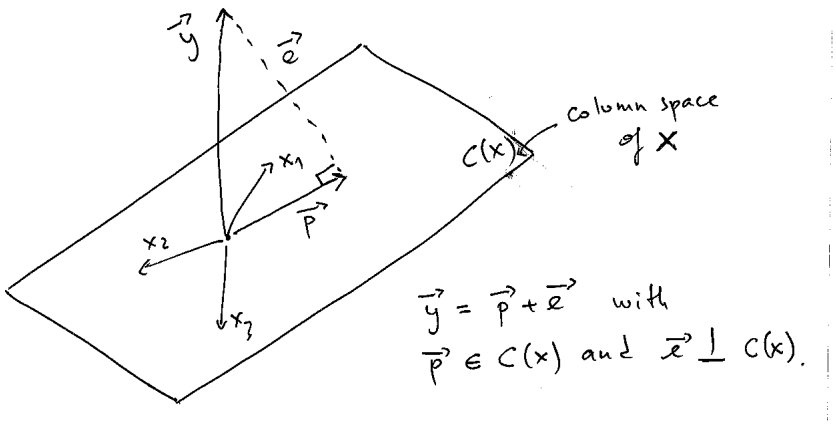

Figure A.1: Projection onto column space

A.2.4 Solution using pseudoinverse

Remember that we want to find Φ sucht that:

$$y = X\Phi + e.$$

If the matrix X was invertible, then we would easily get a solution for Φ. But, in general X is an $N \times (p+1)$ matrix. Therefore, X is not square and does not have an inverse. However, any matrix has a pseudoinverse (see Section 1.7.6).

If we call $X^{\mp}$ the pseudoinverse of X, the solution to the linear approximation problem is

$$\Phi = X^{\pm} y. \tag{A.7}$$

Note that if the SVD of X is given by $X = U\Sigma V^T$, then its pseudoinverse is given by

$$X^{\mp} = V\Sigma^{\mp}U^T, \qquad \text{where} \qquad \Sigma^{\mp} = \begin{bmatrix} \Sigma^{-1} & 0 \\ 0 & 0 \end{bmatrix}.$$

Σ^{-1} is the inverse of the diagonal submatrix formed by the nonzeros singular values of X.

NOTE

- If $(X^TX)^{-1}$ exists, then it is easy to showt that $X^{\pm} = (X^TX)^{-1}X^T$. In that case, we recognize that $\Phi = (X^TX)^{-1}X^Ty$ (same solution as before).
- However, the soltution $\Phi = X^{\pm}y$ is more general, since even if $(X^TX)^{-1}$ does not exist, the pesuedo inverse always does!

A.2.5 Completing the square

We see that the error function E is quadratic:

$$\begin{aligned} E &= (y - X\Phi)^T(y - X\Phi) \\ &= \Phi^TX^TX\Phi - (y^TX\Phi + \Phi^TX^Ty) + y^Ty \end{aligned}$$

Now assume we could write E in the form $(\Phi - s)^T X^T X(\Phi - s)$ plus some constant. Then, if we take $\Phi = s$ the quadratic term equals zero and E is minimized. Therefore, s would be the value of Φ we want.

Let expand the expression $(\Phi - s)^T X^T X(\Phi - s)$. We get

$$E = \Phi^T X^T X\Phi - (\underbrace{s^T X^T X\Phi}_{y^T X\Phi} + \underbrace{\Phi^T X^T Xs}_{\Phi^T X^T y}) + s^T X^T Xs + \underbrace{c}_{y^T y - s^T X^T Xs},$$

where c is a constant vector.

This looks very similar to second expansion of E above, and the underbraces show the term to term equalization to make in order to have equivalent expressions for E.

If we identify the term $s^T X^T X\Phi$ with the term $y^T X\Phi$, we see that we need to have

$$s^T X^T X\Phi = y^T X\Phi \Rightarrow s^T = y^T X(X^T X)^{-1}.$$

So, we can write E as

$$E = (\Phi - s)^T X^T X(\Phi - s) + y^T y - s^T X^T Xs.$$

The last two terms in the above equation are constant, so the minimum of the quadratic function is obtained for $\Phi = s$, i.e.

$$\Phi = (X^T X)^{-1} X^T y.$$

NOTE

This method is a 'usual' approach for solving quadratic equations, called *completing the square.*

Appendix B

Notes about differentiability

In this section, we review the notion of differentiabilty of functions and matrices. These are useful concepts to know with applications in different areas such as optimization and numerical analysis.

B.1 Differentiable functions

Let $f : \mathbb{R}^n \to \mathbb{R}$ be a function, and let a vector $\mathbf{x} \in \mathbb{R}^n$ and a direction $\mathbf{p} \in \mathbb{R}^n$ be given.

- f is said to be *differentiable* at $\mathbf{x}$ when $f'(\mathbf{x}; \mathbf{p})$, called *the directional derivative* of f along direction $\mathbf{p}$ at point $\mathbf{x}$, defined by the limit

$$f'(\mathbf{x}; \mathbf{p}) = \lim_{x \downarrow 0} \frac{f(\mathbf{x} + \lambda \mathbf{p}) - f(\mathbf{x})}{\lambda},$$

  exists for all directions $\mathbf{p} \in \mathbb{R}^n$.

- If the directional derivative $f'(\mathbf{x}; \mathbf{p}) : \mathbb{R}^n \to \mathbb{R}$ is linear, we can write

$$f'(\mathbf{x}; \mathbf{p}) = \nabla f(\mathbf{x})^T \mathbf{p} \text{ for all } \mathbf{p} \in \mathbb{R}^n.$$

- $\nabla f(\mathbf{x})$ is the *gradient* of f at $\mathbf{x}$ and is given by

$$\nabla f(\mathbf{x}) = \begin{bmatrix} \frac{\partial f(\mathbf{x})}{\partial x_1} \\ \vdots \\ \frac{\partial f(\mathbf{x})}{\partial x_n} \end{bmatrix}, \tag{B.1}$$

  where $\frac{\partial f(\mathbf{x})}{\partial x_i}$ for $i = 1, \ldots, n$ is a *partial derivative* of f at $\mathbf{x}$ given by

$$\frac{\partial f(\mathbf{x})}{\partial x_i} = \lim_{\lambda \to 0} \frac{f(\mathbf{x} + \lambda e_i) - f(\mathbf{x})}{\lambda}.$$

When f is differentiable at $\mathbf{x}$ for all $\mathbf{x}$ in some (open) set X, we say that f is *differentiable* over X. When f is differentiable over $\mathbb{R}^n$, we just say that f is differentiable.

If $f : \mathbb{R}^n \to \mathbb{R}$ has second partial derivatives $\frac{\partial^2 f(\mathbf{x})}{\partial x_i \partial x_j}$ for all i, j at a given point $\mathbf{x}$, we say that f is *twice differentiable* at $\mathbf{x}$.

- The matrix $\nabla^2 f(\mathbf{x})$ is called the *Hessian* and is given by

$$\nabla^2 f(\mathbf{x}) = \begin{bmatrix} \frac{\partial^2 f(\mathbf{x})}{\partial x_1 \partial x_1} & \cdots & \frac{\partial^2 f(\mathbf{x})}{\partial x_1 \partial x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial^2 f(\mathbf{x})}{\partial x_n \partial x_1} & \cdots & \frac{\partial^2 f(\mathbf{x})}{\partial x_n \partial x_n} \end{bmatrix}. \tag{B.2}$$

- Since, second partial derivatives are symmetric, $\frac{\partial^2 f(\mathbf{x})}{\partial x_i \partial x_j} = \frac{\partial^2 f(\mathbf{x})}{\partial x_j \partial x_i}$, the *Hessian is a symmetric matrix.*

Theorem B.1.1. (Taylor expansions).
Let $X \subseteq \mathbb{R}^n$ be a open set and let $\mathbf{x}, \mathbf{y} \in X$. Let $f : X \to \mathbb{R}$ be a function.

1. *If f is continuously differentiable over X, then*

$$f(\mathbf{y}) = f(\mathbf{x}) + \nabla f(\mathbf{x})^T(\mathbf{y} - \mathbf{x}) + o(\|\mathbf{y} - \mathbf{x}\|). \tag{B.3}$$

2. *If f is twice continuously differentiable over X, then*

$$f(\mathbf{y}) = f(\mathbf{x}) + \nabla f(\mathbf{x})^T(\mathbf{y} - \mathbf{x}) + \frac{1}{2}(\mathbf{y} - \mathbf{x})^T \nabla^2 f(\mathbf{x})(\mathbf{y} - \mathbf{x}) + o(\|\mathbf{y}\|), \tag{B.4}$$

where $o(h)$ is a continuous function such that $\lim_{\alpha \to 0} \frac{o(\alpha)}{\alpha} = 0$.

Differentiable maps
Let $F : \mathbb{R}^n \to \mathbb{R}^m$ be a mapping given by

$$F(\mathbf{x}) = \begin{bmatrix} f_1(\mathbf{x}) \\ \vdots \\ f_m(\mathbf{x}) \end{bmatrix},$$

where $f_i : \mathbb{R}^n \to \mathbb{R}$.

- When each F_i is differentiable at a given $\mathbf{x}$, then the mapping F is differentiable at $\mathbf{x}$.
- The matrix whose rows are $\nabla f_1(\mathbf{x})^T, \ldots, \nabla f_m(\mathbf{x})^T$ is called the *Jacobian* of F at $\mathbf{x}$, and denoted $JF(\mathbf{x})$.
- $JF(\mathbf{x})$ is an $m \times n$ matrix given by

$$JF(\mathbf{x}) = \begin{bmatrix} \nabla f_1(\mathbf{x})^T \\ \vdots \\ \nabla f_m(\mathbf{x})^T \end{bmatrix}. \tag{B.5}$$

Chain rules
Let $G : \mathbb{R}^p \to \mathbb{R}^n$ be a mapping differentiable at x, and let $F : \mathbb{R}^n \to \mathbb{R}^m$ be another mapping differentiable at $G(\mathbf{x})$. Then, the composite mapping FoG is differentiable at $\mathbf{x}$ and the following *chain rule* holds for the Jacobian $J(FoG)(\mathbf{x})$:

$$J(FoG)(\mathbf{x}) = JF(G(\mathbf{x})JG(\mathbf{x}). \tag{B.6}$$

When G is a linear mapping given by $G(\mathbf{x}) = A\mathbf{x}$, then we have

$$J(FoG)(\mathbf{x}) = JF(A\mathbf{x})A. \tag{B.7}$$

NOTE
The chain rule also holds for the gradient and the Hessian, i.e. if $G : \mathbb{R}^p \to \mathbb{R}^n$ be a linear mapping ($G(\mathbf{x}) = A\mathbf{x}$) differentiable at x and, if $f : \mathbb{R}^n \to \mathbb{R}$ is a differentiable function at $\mathbf{x}$, then we have

$$\nabla(foG)(\mathbf{x}) = A^T \nabla f(A\mathbf{x}). \tag{B.8}$$

$$\nabla^2(foG)(\mathbf{x}) = A^T \nabla^2 f(A\mathbf{x})A. \tag{B.9}$$

B.2 Matrix differentiation

In the following, $\mathbf{x}$ and $\mathbf{y}$ are vectors while x and y are scalars.

Derivative of a scalar w.r.t. vector
If y is a scalar,

$$\frac{\partial y}{\partial \mathbf{x}} \triangleq \begin{bmatrix} \frac{\partial y}{\partial x_1} \\ \vdots \\ \frac{\partial y}{\partial x_n} \end{bmatrix}. \tag{B.10}$$

Derivative of vector w.r.t. scalar
If x is a scalar,

$$\frac{\partial \mathbf{y}}{\partial x} \triangleq \begin{bmatrix} \frac{\partial y_1}{\partial x} & \cdots & \frac{\partial y_m}{\partial x} \end{bmatrix}. \tag{B.11}$$

Derivative of vector w.r.t. vector
The derivative of vector $\mathbf{y}$ w.r.t. vector $\mathbf{x}$ is the $n \times m$ matrix

$$\frac{\partial \mathbf{y}}{\partial \mathbf{x}} \triangleq \begin{bmatrix} \frac{\partial y_1}{\partial x_1} & \cdots & \frac{\partial y_m}{\partial x_1} \\ \frac{\partial y_1}{\partial x_2} & \cdots & \frac{\partial y_m}{\partial x_2} \\ \vdots & \ddots & \vdots \\ \frac{\partial y_1}{\partial x_n} & \cdots & \frac{\partial y_m}{\partial x_n} \end{bmatrix}. \tag{B.12}$$

Example: Given $\mathbf{y} = [y_1\ y_2]^T$ and $\mathbf{x} = [x_1\ x_2\ x_3]^T$, and

$$\begin{array}{rcl} y_1 & = & x_1^2 - x_2 \\ y_2 & = & x_3^2 + 3x_2 \end{array}.$$

The partial derivative matrix $\partial\mathbf{y}/\partial\mathbf{x}$ is computed as follows:

$$\frac{\partial \mathbf{y}}{\partial \mathbf{x}} = \begin{bmatrix} \frac{\partial y_1}{\partial x_1} & \frac{\partial y_2}{\partial x_1} \\ \frac{\partial y_1}{\partial x_2} & \frac{\partial y_2}{\partial x_2} \\ \frac{\partial y_1}{\partial x_3} & \frac{\partial y_2}{\partial x_3} \end{bmatrix} = \begin{bmatrix} 2x_1 & 0 \\ -1 & 3 \\ 0 & 2x_3 \end{bmatrix}.$$

$\mathbf{y}(\mathbf{x})$	$\frac{\partial \mathbf{y}}{\partial \mathbf{x}}$
$A\mathbf{x}$	A^T
$\mathbf{x}^T A$	A
$\mathbf{x}^t\mathbf{x}$	$2\mathbf{x}$
$\mathbf{x}^T A\mathbf{x}$	$A\mathbf{x} + A^T\mathbf{x}$

Table B.1: Some useful derivatives formulas.

Derivative formulas
Some frequently used formulas are given in the table B.1.

Cahin rule
Let $\mathbf{x}$, $\mathbf{y}$ and $\mathbf{z}$ be vectors such as $\mathbf{z}$ is a function of $\mathbf{y}$ which is in turn a function of $\mathbf{x}$. We have the following rule

$$\frac{\partial \mathbf{z}}{\partial \mathbf{x}} = \frac{\partial \mathbf{y}}{\partial \mathbf{x}} \frac{\partial \mathbf{z}}{\partial \mathbf{y}}. \tag{B.13}$$

Derivative of scalar functions of a matrix
Let $\mathbf{X}$ be an $m \times n$ matrix, and let $y = f(\mathbf{X})$ be a scalar function.

The derivative of y w.r.t. $\mathbf{X}$ is defined as the following $m \times n$ matrix

$$\mathbf{G} = \frac{\partial y}{\partial \mathbf{X}} = \begin{bmatrix} \frac{\partial y}{\partial x_{11}} & \cdots & \frac{\partial y}{\partial x_{1n}} \\ \vdots & \ddots & \vdots \\ \frac{\partial y}{\partial x_{m1}} & \cdots & \frac{\partial y}{\partial x_{mn}} \end{bmatrix} = \left[\frac{\partial y}{\partial x_{ij}}\right] = \sum_{i,j} \mathbf{E}_{ij} \frac{\partial y}{\partial x_{ij}}, \tag{B.14}$$

where $\mathbf{E}_{ij}$ denotes the elementary $m \times n$ matrix (i.e. a matrix with all entries equal to zero except for the (i, j) entry which is one).

B.3 Examples

Compute the gradients and Hessians of the following vector input functions:

1. $g(\mathbf{w}) = \frac{1}{2}\mathbf{w}^T\mathbf{Q}\mathbf{w} + \mathbf{r}^T\mathbf{w} + d$, where $\mathbf{Q}$ is a $N \times N$ symmetric matrix, $\mathbf{r}$ is a $N \times 1$ vector, and d is a scalar.
2. $g(\mathbf{w}) = \sum_{p=1}^{P} \log(1 + e^{-\mathbf{a}_p^T\mathbf{w}})$, where $\mathbf{a}_i$, $i = 1, \ldots, P$ are $N \times 1$ vectors.

You should find the following results:

1. $$\nabla g(\mathbf{w}) = \mathbf{Q}\mathbf{w} + \mathbf{r};\ \nabla^2 g(\mathbf{w}) = \mathbf{Q}$$
2. $$\nabla g(\mathbf{w}) = -\sum_{p=1}^{P} \frac{\mathbf{a}_p}{1 + e^{\mathbf{a}_p^T\mathbf{w}}};\ \nabla^2 g(\mathbf{w}) = \sum_{p=1}^{P} \frac{e^{\mathbf{a}_p^T\mathbf{w}}}{(1 + e^{\mathbf{a}_p^T\mathbf{w}})^2} \mathbf{a}_p\mathbf{a}_p^T$$

Appendix C

Notes on Some Important Distributions

A summary of some important probabilty distributions.

C.1 Bernoulli trials

Let us perform an experiment, or trial, whose outcome can be classified as either a success or a failure. We define a r.v. X such as:

$$X = \begin{cases} 1 & \text{if the outcome is a success} \\ 0 & \text{if the outcome is a failure} \end{cases}$$

For example, consider the case of flipping a coin and define $X = 1$ if one gets a head, and $X = 0$ if one gets a tail.

If p is the probability of a success, the the pmf of X is

$$P(X = 1) = p \text{ and } P(X = 0) = 1 - p. \tag{C.1}$$

X is called a Bernoulli r.v. with parameter $p \in [0, 1]$.

Expected value and variance

$$\mathbb{E}[X] = p. \tag{C.2}$$

$$\mathbb{V}\text{ar}\,[X] = p(1 - p). \tag{C.3}$$

C.2 Binomial random variables

Now consider that we perform n independent Bernoulli trials, each of them having probability p of success and probability $1 - p$ of failure.

Let $X =$ number of success in the n trials.

Then X is a called a Binomial r.v. with parameters n and p, denoted $\mathcal{B}(n, p)$. The pmf of X is given by

$$P(X = k) = \left(\begin{array}{c} n \\ k \end{array} \right) p^k (1 - p)^{n-k}, \text{ for } k = 0, 1, \ldots, n. \tag{C.4}$$

Expected value and variance

$$\mathbb{E}[X] = np. \tag{C.5}$$

$$\mathbb{V}\text{ar}\,[X] = np(1-p). \tag{C.6}$$

C.3 Geometric random variables

Consider that we perform n independent Bernoulli trials, each of them having probability p of success and probability $1-p$ of failure.

Let X = number of trials until we get the first success.

X is a r.v. that can take on values of $1, 2, 3, \ldots$. X is a called a Geometric r.v. with parameter p, denoted $\mathcal{G}(p)$. The pmf of X is given by

$$P(X = k) = p(1-p)^{k-1} \text{ for } k \geq 0. \tag{C.7}$$

Expected value and variance

$$\mathbb{E}[X] = \frac{1}{p}. \tag{C.8}$$

$$\mathbb{V}\text{ar}\,[X] = \frac{1-p}{p^2}. \tag{C.9}$$

C.4 Poisson random variables

A r.v. X, taking values $0, 1, 2, \ldots$, is said to be a Poisson r.v. with parameter λ, denoted $\mathcal{P}(\lambda)$, if for some $\lambda > 0$

$$P(X = k) = \frac{\lambda^k}{k!} e^{-\lambda} \text{ for } k \geq 0. \tag{C.10}$$

The Poisson r.v. has a large number of applications. A major reason of this being that a Poisson r.v. can be used as an approximation for a Binomial r.v. with parameters (n, p) when n is large and p small (rare events).

Expected value and variance

$$\mathbb{E}[X] = \lambda. \tag{C.11}$$

$$\mathbb{V}\text{ar}\,[X] = \lambda. \tag{C.12}$$

C.5 Poisson process

A Poisson process is a model for counting occurrences, or events, over an interval of time. For example, we may be interested in the number of customers entering a shop in a period of t hours.

A Poisson process having rate λ means that the number of events occurring in any fixed interval of lenght t units, is a Poisson random variable with mean λt. The value λ is the rate per unit time at which events occur and must be empirically determined.

If we write that the number of occurrences during t time units is $N(t)$, then $N(t) \rightsquigarrow \mathcal{P}(\lambda t)$.

C.6 Uniform random variable

A continuous r.v. X is said to be uniformly distributed over the interval (a, b) if its pdf is given by

$$f_X(x) = \begin{cases} \frac{1}{b-a}, \text{ if } a < x < b \\ 0, \text{ otherwise.} \end{cases} \tag{C.13}$$

Expected value and variance

$$\mathbb{E}[X] = \frac{a+b}{2}. \tag{C.14}$$

$$\mathbb{V}\text{ar}\,[X] = \frac{(b-a)^2}{12}. \tag{C.15}$$

C.7 Exponential random variable

A continuous r.v. X is said to be an exponential r.v. with parameters λ ($\lambda > 0$) if its pdf is given by

$$f_X(x) = \begin{cases} \lambda e^{-\lambda x}, \text{ if } x \geq 0 \\ 0, \text{ otherwise.} \end{cases} \tag{C.16}$$

Expected value and variance

$$\mathbb{E}[X] = \frac{1}{\lambda}. \tag{C.17}$$

$$\mathbb{V}\text{ar}\,[X] = \frac{1}{\lambda^2}. \tag{C.18}$$

The exponential r.v. can be used to model lifetimes or time between unlikely events. For instance, they are used to model the queuing problem, i.e. the problem of the customers waiting in a queue for service.

Memoryless property

An important property of exponential r.v. is the memoryless property:

$$P(X > s + t | X > t) = P(X > s), \ \forall s, t > 0. \tag{C.19}$$

The interpretation of this property is that if X is, for example, the lifetime of a product, then the residual lifetime $X - t$ of that product, if it is still operating after t time units, has the same distribution as that of a new product.

C.8 Gamma random variable

A r.v. X is a Gamma r.v. with parameters (α, λ), $(\alpha, \lambda > 0)$, if its pdf is given by

$$f_X(x) = \begin{cases} \frac{\lambda e^{-\lambda x}(\lambda x)^{\alpha-1}}{\Gamma(\alpha)}, \text{ if } x \leq 0 \\ 0, \text{ otherwise,} \end{cases} \tag{C.20}$$

where the Gamma function is defined by

$$\Gamma(\alpha) = \int_0^\infty e^{-y} y^{\alpha-1} dy.$$

The Gamma function has different nice properties. It can be shown that

$$\Gamma(1) = 1 \text{ and } \Gamma(1/2) = \sqrt{\pi}$$

$$\Gamma(\alpha) = (\alpha - 1)\Gamma(\alpha - 1)$$

Thus, for integers $(\alpha = n)$, we have

$$\Gamma(n) = (n - 1)!.$$

Expected value and variance

$$\mathbb{E}[X] = \frac{\alpha}{\lambda}. \tag{C.21}$$

$$\mathbb{V}\text{ar}\,[X] = \frac{\alpha}{\lambda^2}. \tag{C.22}$$

Moreover, many r.v.s can be seen as special cases of the Gamma r.v.

- If X is a Gamma r.v. with $\alpha = 1$, then X is an exponential r.v.
- If X is a Gamma r.v. with $\lambda = 1/2$ and $\alpha = n/2$, then X is χ^2 with n degrees of freedom.

C.9 Normal random variable

A r.v. X is a normal (Gaussian) r.v. with parameters μ and σ^2 if its pdf is given by

$$f_X(x) = \frac{1}{\sqrt{2\pi\sigma^2}} exp\{-\frac{(x-\mu)^2}{2\sigma^2}\}, \text{ for } x \in \mathbb{R}. \tag{C.23}$$

This is the density function of the so-called Bell curve. The normal distribution shows up frequently in practice due, in large part, to the central limit theorem.

Expected value and variance

$$\mathbb{E}[X] = \mu. \tag{C.24}$$

$$\mathbb{V}\text{ar}\,[X] = \sigma^2. \tag{C.25}$$

Some important facts about normal r.v.s

- If X is normally distributed with parameters μ and σ^2, then $Y = aX + b$ is also normally distributed with parameters $a\mu + b$ and $a^2\sigma^2$.
- If X and Y are independent normal r.v.s with parameters (μ_1, σ_1^2) and (μ_2, σ_2^2), then $X + Y$ is also normally distributed with mean $\mu_1 + \mu_2$ and variance $\sigma_1^2 + \sigma_2^2$.

C.10 Useful inequalities

Inequalities are often useful to estimate some expected values. Here are a few particularly useful ones.

- **Exponential bound:**
$$1 + x \leq exp(x). \tag{C.26}$$
- **Chebychev:**
$$P(X \leq a) \leq \mathbb{E}(X^2)/a^2. \tag{C.27}$$
- **Markov inequality:** If $f(.)$ is nonnegative and nondecreasing on $[a, +\infty)$, then
$$P(X \leq a) \leq \{\mathbb{E}(f(X))\}/f(a). \tag{C.28}$$
- **Jensen inequality:** If $f(.)$ is convex, then
$$\mathbb{E}(f(X)) \geq f(\mathbb{E}(X)). \tag{C.29}$$

Appendix D

Discrete Kalman Filter

An example of discrete Kalman filter.

D.1 Background

Kalman Filter (KF) is an optimal linear filter that recursively estimates parameters of a system. It is based on the state-space representation of the system:

$$x_k = Ax_{k-1} + Bu_{k-1} + w_{k-1}$$

$$z_k = Cx_k + v_k$$

KF uses prediction-correction cycles:

- Prediction step

$$\hat{x}_{k+1}^- = A\hat{x}_k + Bu_k$$

$$P_{k+1}^- = AP_kA^T + Q_k$$

- Correction step

$$K_{k+1} = P_{k+1}^- C^T [CP_{k+1}^- C^T + R_{k+1}]^{-1}$$

$$\hat{x}_{k+1} = \hat{x}_{k+1}^- + K_{k+1}(z_{k+1} - C\hat{x}_{k+1}^-)$$

$$P_{k+1} = (I - K_{k+1}C)P_{k+1}^-$$

D.2 Example: Estimation of a scalar constant

We want to estimate a scalar random constant x. Let's assume that we can get a measure of x, but the measurements are corrupted by white Gaussian noise of standard deviation 0.1.

1. Write the state and measurement equations for this system.

2. Use matlab to simulate $N = 100$ distinct measurements $z_k, k = 1 \ldots, N$, assuming a small process noise of standard deviation 10^{-3} and a constant value equal to 0.50235.

3. Write a function `discrete_kalman_filter.m` that takes as input, the state-space representation of a system, the observations $z_k, k = 1 \ldots, N$, the initial state vector x_0 and initial state covariance matrix P_0, and return the estimate of the state $\hat{x}_k, k = 1, \ldots, N$.

4. Estimate the value of x, for $x_0 = 0$ and $P_0 = 1$. How does the covariance matrice P_k vary over time (here P_k is scalar)?

5. What happen if the value of R, the measurement variance, is increased or decreased by a factor of 100 respectively?

Solution

The state-space representation of this system is just

$$x_k = x_{k-1} + w_{k-1}$$

$$z_k = x_k + v_k$$

We can use the following Matlab code to simulate the system.

```
% the state-space equations
A = 1; % transition matrix
B = 0; % input matrix: no control input
C = 1; % measurement matrix: noisy measurement is the state
% calculation of noise covariance matrices
R = Meas_Noise^2; % measurement noise cov
Q = Process_Noise^2;  % process noise cov
% initialization
x = -0.51234;  % true state
% simulation
x_out = zeros(duration,1);
y_out = zeros(duration,1);
Process_Noise = Process_Noise * randn(duration,1);
Meas_Noise = Meas_Noise * randn(duration,1);
for t=1:duration,
    x_out(t) = A*x + Process_Noise(t);
    y_out(t) = C*x + Meas_Noise(t);
end
```

Finally a discrete Kalman filter can be written like this

```
function [x_out]=discrete_kalman_filter(y, u, x0, P0, A,B,C,Q,R)
T = size(y, 1); %number of observations
[n,m] = size(P0); I = eye(n,m); % form identity matrix
xhat = x0; P = P0; %initialization
x_out = [];
for k=1:T,
   % compute Kalman Gain
   K = P*C'* inv(C*P*C' + R);
   % estimate state
   innovation = y(k) - C*xhat; % innovation vector
   xhat = xhat + K*innovation;
   x_out = [x_out; xhat'];
   % update covariance matrice
   P = (I - K*C)*P;
   % predict next state and covariance
   xhat = A*xhat + B*u;
   P = A*P*A' + Q;
end
```

Given that $Q_k = 1e-5$ and $R_k = 0.01$, and initializing the filter with $\hat{x}_0 = 0$ and $P_0 = 1$, we obtain the estimation result in Fig. D.1.

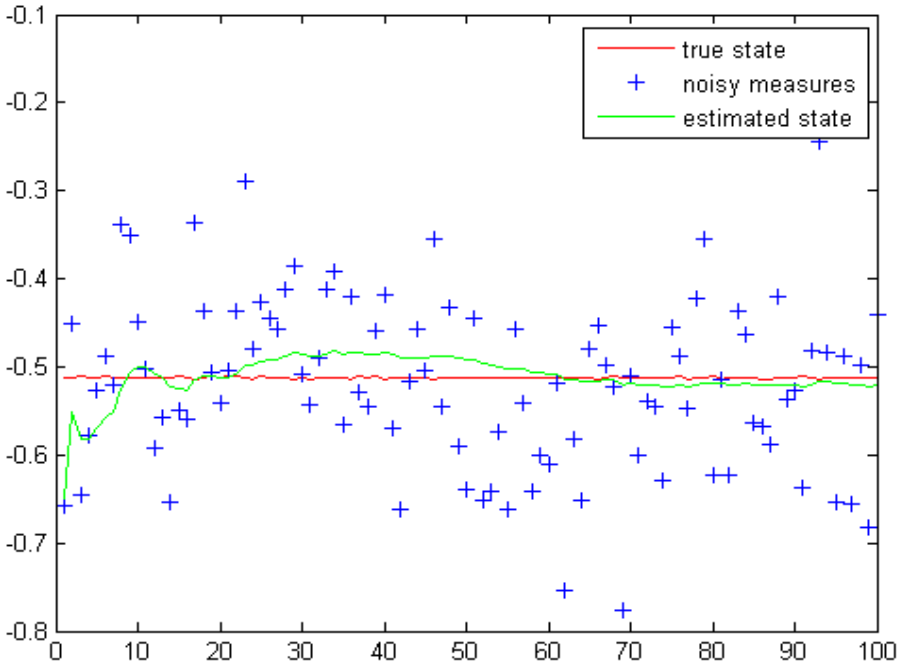

Figure D.1: Random constant estimation with a Kalman Filter.

Appendix E

Applications

Some computer vision applications.

E.1 SVD and Image Compression

The SVD of an $m \times n$ matrix A of rank k is given by $A = U\Sigma V^T$, which can also be written as $A = \sum_{i=1}^{k} \sigma_i u_i v_i^T$.

So, the matrix A is a sum of rank one matrices that are orthogonal with respect to the matrix inner product. Truncating the sum at p terms defines a rank p matrix $A_p = \sum_{i=1}^{p} \sigma_i u_i v_i^T$. If we approximate A with A_p, then we make an error equals to $E_p = A - A_p = \sum_{i=p+1}^{k} \sigma_i u_i v_i^T$. It can be shown that A_p is the best rank p approximation to A.

These are the basic ideas behind SVD: taking a high dimensional, highly variable set of data points and reducing it to a lower dimensional space that exposes the substructure of the original data more clearly and orders it from most variation to the least. Thus, SVD can be used for image compression.

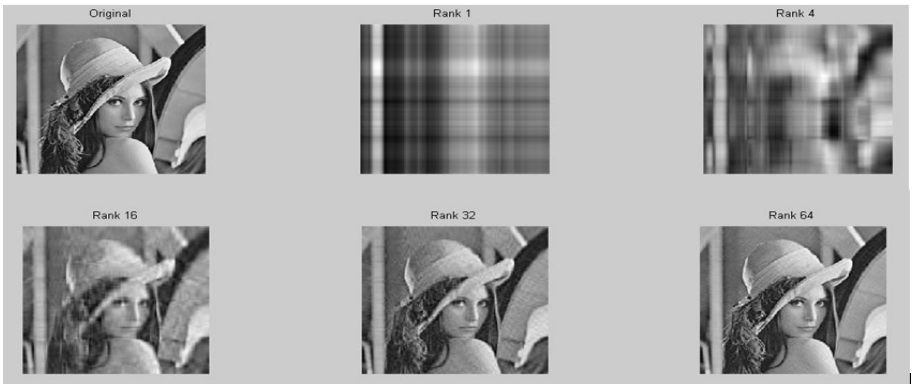

Figure E.1: Image compression using SVD.

E.2 PCA based Face Recognition

The basic idea of any FR system is to extract a set of interesting and discriminative features from the face images with the goal of reducing the number of variables. The Eigenfaces approach is a PCA based method, in which a small set of characteristic pictures is used to describe the variation between face images. The goal is to find the eigenvectors (eigenfaces) of the covariance matrix of the distribution, spanned by a training set of face images. Later, every face image is represented by a linear combination of these eigenvectors. Recognition is performed by projecting a new image onto the subspace spanned by the eigenfaces and then classifying the face by comparing its position in the face space with the positions of known individuals. PCA is fast and needs lesser amount of memory, since it basically performs dimensionality reduction.

Figure E.2: Eigenfaces.

Reference: M. Turk and A. Pentland, *Face recognition using eigenfaces.* Proc. of CVPR, pp. 586-591, 1991.

E.3 Active Shape and Appearance Models

Statistical models of shape and appearance are powerful tools for interpreting and modeling shapes and images. Given a training set of images in which corresponding 'landmark' points have been marked on every image, we can compute a a statistical model of the shape variation, a model of the texture variation and a model of the correlations between shape and texture. With enough training examples such models should be able to synthesize any image of normal anatomy. By Finding the parameters which optimize the match between a synthesized model image and a target image we can locate all the structures represented by the model. The Active Shape Model essentially matches a model to boundaries in an image. The Active Appearance Model finds model parameters which synthesize a complete image which is as similar as possible to the target image. These models were proposed by Cootes and Taylor and have been widely used in computer vision.

Reference: T. F. Cootes, G. J. Edwards, and C. J. Taylor. *Active appearance models.* IEEE TPAMI, 23(6):681-685, 2001.

Figure E.3: Active Models.

E.4 Fundamental Matrix Estimation

The first step for most computer vision algorithms is almost always the computation of the Fundamental matrix, which is of great theoretical and practical importance.

Before estimating the fundamental matrix, we first need to introduce the very important concept of *epipolar geometry*.

Recall on epipolar geometry

Epipolar geometry is the intrinsic projective geometry between two views. It is independent of scene structure, and only depends on the cameras' internal parameters and relative pose.

Epipolar geometry gives a constraint between corresponding points, i.e. if a 3D point $\mathbf{X}$ of the scene is projected onto $\mathbf{x}$ and $\mathbf{x}'$ in the two views as depicted in Fig. E.4, then the image points $\mathbf{x}$ and $\mathbf{x}'$ must satisfy the epipolar constraint:

$$\mathbf{x}^T F \mathbf{x}' = 0, \tag{E.1}$$

where $\mathbf{x} = \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$, $\mathbf{x}' = \begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix}$ and $F = \begin{bmatrix} f_{11} & f_{12} & f_{13} \\ f_{21} & f_{22} & f_{23} \\ f_{31} & f_{32} & f_{33} \end{bmatrix}$.

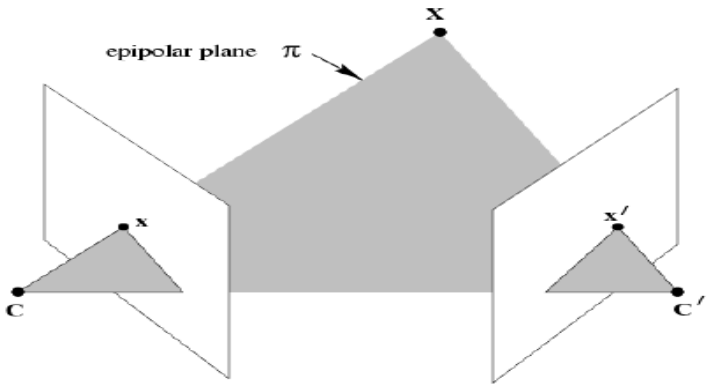

Figure E.4: Epipolar geometry.

Computing the fundamental matrix

Each corresponding couple of points $(\mathbf{x}, \mathbf{x}')$ yields one equation E.1. Therefore, with

a sufficient number of correspondences in general position it is possible to determine F. No knowledge about the cameras or scene structure is necessary.

The epipolar constraint equation is linear in the entries of F and it can be rewritten as: $\mathbf{U}^T\mathbf{f} = 0$, where

$$\mathbf{U} = [xx', xy', x, yx', yy', y, x', y', 1]^T$$

,

$$\mathbf{f} = [f_{11}, f_{12}, f_{13}, f_{21}, f_{22}, f_{23}, f_{31}, f_{32}, f_{33}]^T$$

.

With N correspondences, we obtain a linear system of the form $\mathbf{Af} = 0$ and solve for $\mathbf{f}$. Therefore, some knowledge of linear algebra is necessary. We can use LLS or SVD to find F.

How many points?

The first answer to this question would be $N \geq 9$ (you should be able to guess why!).

But, the fundamental matrix F is defined up to a scale factor (this is a consequence of projective geometry). Moreover, the fact that the epipole in the second image belongs to all the epipolar lines implies that F is a singular matrix, i.e. $det(F) = 0$. Therefore, F only depends on seven parameters.

Using seven points, it is possible to compute F using the rank constraint $det(F) = 0$. However, there is no unique solution. With eight correspondences, there is a unique solution which can be obtained linearly. In practice, we will use at least eight correspondences.

Enforcing the rank 2 constraint

In practice, the fundamental matrix obtained as the solution of the linear system does not satisfy the constraint $det(F) = 0$. So, we have to find the best rank two matrix from the obtained F. Best in the sense that the obtained matrix F' minimizes the Frobenius norm $\|F - F'\|$.

If the SVD of F is $F = USV^T$, with $S = \begin{bmatrix} \lambda_1 & 0 & 0 \\ 0 & \lambda_2 & 0 \\ 0 & 0 & \lambda_3 \end{bmatrix}$, then we obtain $F' = US'V^T$ taking $S' = \begin{bmatrix} \lambda_1 & 0 & 0 \\ 0 & \lambda_2 & 0 \\ 0 & 0 & 0 \end{bmatrix}$.

Reference: D. Sidibé. *Lecture Notes on Computer Vision.* University of Burgundy, 2010.

E.5 Particle Filtering based Object Tracking

Tracking moving objects is an important task in many computer vision applications including video surveillance, smart rooms, mobile robotics, augmented reality and video compression. Despite many effort, it is still a challenging problem due to the presence of noise, changes of illumination, cluttered background and occlusions that introduce uncertainty in the estimation of the object's state.

The main objective of tracking is to roughly predict and estimate the location of a target object in each frame of a sequence. When using a probabilistic method, the tracking problem can then be viewed as a Bayesian inference problem. In the case of a linear dynamic model with Gaussian noise, Kalman filter provides an optimal solution. However, for non-linear and non-Gaussian cases, it is impossible to find analytic solutions. Over the last decade, particles filters, also known as condensation or sequential Monte Carlo methods, have proved to be very efficient for object tracking.

To implement the particle filter, one has to define the state vector and the dynamic model of the system. We can define the state as $\mathbf{x} = [x, y, s_x, s_y]^T$, where (x, y) is the location of the target object, s_x and s_y are the scales in the x and y directions. In the prediction stage of the particle filter, the samples are propagated through a dynamic model. We can use a first order auto-regressive (AR) process for simplicity:

$$\mathbf{x}_t = A\mathbf{x}_{t-1} + \mathbf{v}_{t-1}, \tag{E.2}$$

where $\mathbf{v}_{t-1}$ is a multivariate Gaussian random noise and A defines the deterministic system model. A constant velocity model is usually used for the dynamic model.

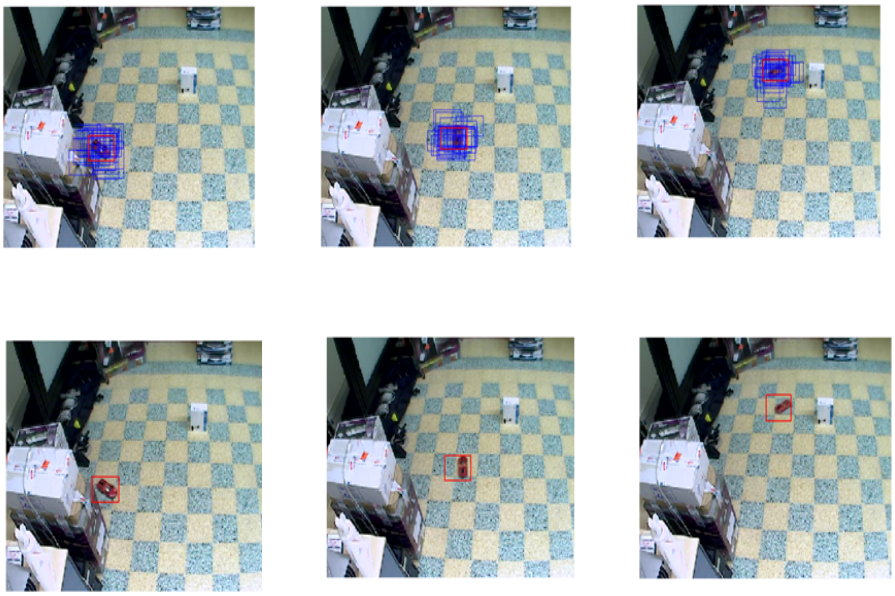

Figure E.5: Particle filter based tracking: for every frame, the mean value of the particles give an estimate of the object's current location.

Reference: P. Perez, J. Vermaak and A. Blake, *Data Fusion for Visual Tracking with Particles*. Proceedings of the IEEE, pages 495-513, 2004.

E.6 SLAM

The Simultaneous Localisation and Mapping (SLAM) is a process by which a mobile robot can build a map of an environment and at the same time use this map to deduce it's location. In SLAM both the trajectory of the platform and the location of all landmarks are estimated on-line without the need for any a priori knowledge of location.

Consider a mobile robot moving through an environment taking relative observations of a number of unknown landmarks using a sensor located on the robot as shown in Fig. E.6. At a time instant k, the following quantities are defined:

- $\mathbf{x}_k$: The state vector describing the location and orientation of the vehicle.
- $\mathbf{u}_k$: The control vector, applied at time $k-1$ to drive the vehicle to a state $\mathbf{x}_k$ at time k.

- $\mathbf{m}_i$: A vector describing the location of the ith landmark whose true location is assumed time invariant.
- $\mathbf{z}_{ik}$: An observation taken from the vehicle of the location of the ith landmark at time k. When there are multiple landmark observations at any one time or when the specific landmark is not relevant to the discussion, the observation will be written simply as $\mathbf{z}_k$.

In addition, the following sets are also definned:

- $\mathbf{X}_{0:k} = \{\mathbf{x}_0; \mathbf{x}_1; \ldots; \mathbf{x}_k\} = \{\mathbf{X}_{0:k-1}; \mathbf{x}_k\}$: The history of vehicle locations.
- $\mathbf{U}_{0:k} = \{\mathbf{u}_1; \mathbf{u}_2; \ldots; \mathbf{u}_k\} = \{\mathbf{U}_{0:k-1}; \mathbf{u}_k\}$: The history of control inputs.
- $\mathbf{m} = \{\mathbf{m}_1; \mathbf{m}_2; \ldots; \mathbf{m}_n\}$: The set of all landmarks.
- $\mathbf{Z}_{0:k} = \{\mathbf{z}_1; \mathbf{z}_2; \ldots; \mathbf{z}_k\} = \{\mathbf{Z}_{0:k-1}; \mathbf{z}_k\}$: The set of all landmark observations.

In probabilistic form, the Simultaneous Localisation and Map Building (SLAM) problem requires that the probability distribution $P(\mathbf{x}_k; \mathbf{m}|\mathbf{Z}_{0:k}; \mathbf{U}_{0:k}; \mathbf{x}_0)$ be computed for all times k.

This is exactly the estimation problem we have seen in chapter 4. Solutions to the SLAM problem include Extended Kalman Filter (EKF-SLAM) and particle filters (Fast-SLAM).

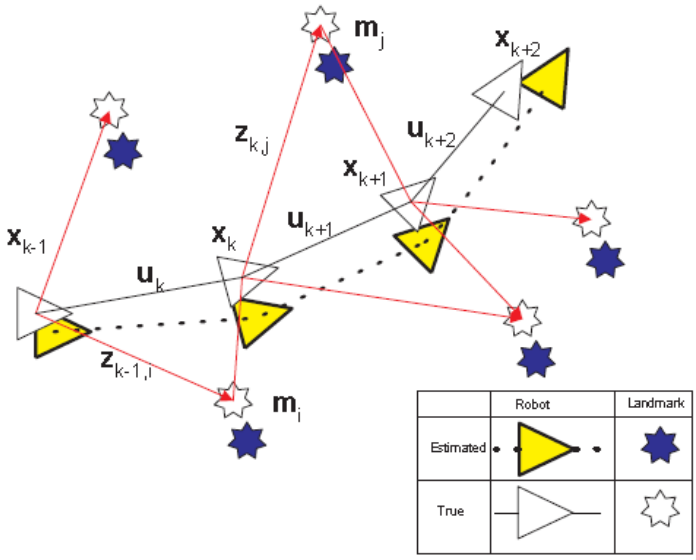

Figure E.6: The essential SLAM problem.

Reference: H. Durrant-Whyte and Tim Bailey, *Simultaneous Localisation and Mapping (SLAM): Part I The Essential Algorithms*, Robotics and Automation Magazine (13):99-110, 2006.

E.7 Ghost Detection in HDRI

If you have ever taken a photograph in very dark or bright conditions, then you surely faced the limited dynamic range problem os most digigtal cameras. Indeed,

a photograph taken with a conventional camera cannot capture the whole dynamic range of real scenes which varies over several orders of magnitude. As a consequence, some regions of the scene will be under- or over-exposed and appear saturated in the image.

To enlarge the dynamic range spanned by conventional cameras a very interesting and powerful technique has been developed in the last few years: high dynamic range imaging. The obtained images are called high dynamic range (HDR) images and represent the scene more faithfully than conventional low dynamic range (LDR) images.

A classic approach for obtaining an HDRI with a conventional camera is to take a sequence of images of the same scene with different exposure times, and combine them to a single radiance map. This multiple exposures technique suffers from two main problems:

- *Misalignment*: if the camera moves during the time of capture, the images will be misaligned and the combined HDRI will look blurry.
- *Ghosting*: if there are moving objects while capturing the sequence of images, these objects will appear in different locations in the combined HDRI, creating what is called ghost or ghosting artifacts.

The first problem can be solved easily by placing the camera on a tripod or by using an image registration method to align the differently exposed images. On the contrary, the second problem is a severe limitation of the multiple exposures technique since motions are hardly avoilable in outdoor environments.

We (Abhilash, a former Master Computer Vision student and I) have proposed a very simple and elegant method based on SVD to solve this problem.

Figure E.7: An example of ghost detection and removal in HDRI. Left: HDR image with ghost artefacts. Right: HDR image with removed ghost.

Reference: A. Srikantha and D. Sidibé, *An SVD-Based Approach for Ghost Detection and Removal in High Dynamic Range Images*, Proc. ICPR, 2012.

Bibliography

[AMGC02] M. S. Arulampalam, S. Maskell, N. Gordon, and T. Clapp. A tutorial on particle filters for online nonlinear/non-gaussian bayesian tracking. *IEEE Transactions on Signal Processing*, 50:174–188, 2002.

[BT02] D. P. Bertsekas and J. N. Tsitsiklis. *Introduction to Probability.* Athena Scientific, Belmont, MA, 2002.

[DGK00] A. Doucet, N. Gordon, and V. Krishnamurthy. On sequential Monte Carlo sampling methods for bayesian filtering. *Statistics and Computing*, 10:197–208, 2000.

[GS97] Charles M. Grinstead and J. Laurie Snell. *Introduction to Probability.* American Mathematical Society, 1997.

[JU97] S. J. Julier and J. K. Uhlmann. A new extension of the kalman filter to nonlinear systems. In *Int. Symp. Aerospace/Defense Sensing, Simulation and Controls*, 1997.

[Kal60] R.E. Kalman. A new approach to linear filtering and prediction problems. *Transaction of the MSME - Journal of Basic Engineering*, 82:35–45, 1960.

[Lay99] David-C Lay. *Linear Algebra and Its Application.* Addison-Wesley/Helix Books, second edition, 1999.

[MSKS06] Yi Ma, Stefano Soatto, Jana Kosecka, and S. Shankar Sastry. *An invitation to 3D Vision: From Images to Geometric Models.* Springer, 2006.

[Str09] Gilbert Strang. *Introduction to Linear Algebra.* Wesley-Cambrigbe Press, fourth edition, 2009.