Thyagaraju Damarla

Battlefield Acoustics

Springer

Battlefield Acoustics

Thyagaraju Damarla

Battlefield Acoustics

Springer

Thyagaraju Damarla
U.S. Army Research Laboratory
Laurel, MD
USA

ISBN 978-3-319-16035-1 ISBN 978-3-319-16036-8 (eBook)
DOI 10.1007/978-3-319-16036-8

Library of Congress Control Number: 2015935214

Springer Cham Heidelberg New York Dordrecht London

Printed on acid-free paper

Springer International Publishing AG Switzerland is part of Springer Science+Business Media (www.springer.com)

To my mother, Ravamma Damarla, who inculcated the importance of education in me at an early age and my father, Ramakantarao Damarla, whose image with a book in hand is etched in my mind forever.

Preface

Situational awareness in the battlefield is an age-old quest. The advent of modern sensors and advances in digital signal processing is making the art of inference from sensor data far more feasible. Acoustic sensors are the ears in the field. One should strive to understand the situation based on what is heard in the area. Moreover, acoustic sensors are omnidirectional and consume less power, so they last a longer once deployed, unlike other sensors, which require frequent change of batteries. As a result, there is a lot of interest in acoustics and its signal processing.

When I first started working in the Acoustics Branch, there were no books that dealt with battlefield acoustics. For the majority of cases, one is forced to look for articles in various journals. Although this contributed to deeper learning and understanding of the subject, a book on the battlefield acoustics would have been a good starting point to help in focusing the search. Situational awareness for intelligence, surveillance, and reconnaissance (ISR) requires detection, classification, and tracking of targets, which could be ground or airborne vehicles, people, hostile gunfire, etc. While performing acoustic signal processing, one should also understand the dynamics of acoustic waves. For example, just as with light, acoustic waves undergo reflection and refraction, which, in turn, have a dramatic effect on the acoustic signals. Thus, while processing acoustic signals, the effects of reflection and refraction should be taken into account.

The organization of the book is as follows. We begin with an introduction to various types of microphones in Chap. 1. The next three chapters present some of the concepts in probability, detection, and estimation theory that are essential for acoustic signal processing. Those with some knowledge of these theories can skim or skip Chaps. 2 through 4.

Chapter 5 presents some concepts in physical acoustics, including the properties of reflection and refraction of acoustic waves. The concept of ground impedance, which plays a significant role in the ground reflection of acoustic signals, is also explored.

Chapter 6 focuses on the theory of microphone arrays. Here, several configurations of microphones, namely linear, circular, and grids of microphones are considered. The beam widths of such arrays are also estimated and the concept of

spatial aliasing is presented. This theory is fundamental in constructing an array for estimating the direction of arrival (DOA) of sound sources depending on the mission. DOA estimation of signals to determine from where they are emanating is vital for finding their location and tracking them—a fundamental aspect of situational awareness. Further, DOA estimation depends on the type of waves that the source is emitting. For example, vehicles emit continuous signals, whereas gunfire, mortar launchings, and detonations emit transient sounds, that is, short bursts of sound.

Chapter 7 details the estimation of DOA from continuous sound sources, covering several methods, namely adaptive beamforming and eigenvector-based techniques. The multiple signal classification (MUSIC) and minimum variance distortionless response (MVDR) techniques are discussed at length. To assist in experimentation, the corresponding MATLAB code is also presented as a jumping off point on the subject. The chapter explains how in order to track a target, several arrays must be deployed so that the estimated DOAs at each array can be used to triangulate to find the location of the target.

Chapter 8 deals with a very important related topic: the fact that often such DOAs are noisy, which means the estimates of the target locations are also noisy at best. To address this issue, the chapter outlines the theoretical concepts of various filters that can be used. Specifically, sequences of target estimations are used to track targets using Kalman, extended Kalman, unscented Kalman, and particle filters. Again, to assist in experimentation the MATLAB code for each filter is also presented.

Chapter 9 considers the nature of transient sound signals that occur mainly due to gunfire or mortars, focusing on the specific problem of determining a sniper's location. Given that supersonic guns emit both shockwave and muzzle blast signals, the speed of a supersonic bullet can be determined by the N-wave due to shockwave. The relevant theory is presented with this chapter. Also, localization of gunfire can be determined by finding the time difference of arrival of the muzzle blast signals at distributed microphones; this theory is also presented in the chapter.

Chapter 10 covers several commonly and most widely used classifiers used to classify signals from various targets, civilian and military vehicles, people, etc. The chapter presents the principles of some of the most popular classifiers, namely multivariate Gaussian classifier, Gaussian mixture model, support vector machines, and neural networks.

Of course, situational awareness requires the knowledge and identity of the targets, and to classify/identify targets, one needs to extract target features that represent the physics-based phenomenology. Chapter 11 presents some of these features and their extraction for vehicles and people. This chapter also details some of the high fidelity features generated by ultrasonic signals to distinguish people and animals.

Quite often, multiple multimodal sensors are used to detect, identify, and track targets in order to improve the detection statistics. The theory of fusion of multiple detections is considered in Chap. 12.

It is my intent that this book will provide the basics of battlefield acoustics and the issues involved, and in so doing, pave the way for engineering solutions. I hope this book will be useful to the practicing engineering students who aspire to be knowledgeable on the subject of battlefield acoustics.

Laurel, March 2015 Thyagaraju Damarla

Acknowledgments

I would like to take this opportunity to thank the U.S. Research Laboratory for giving me the opportunity and resources to explore and understand, and the freedom to work on all aspects of battlefield acoustics for situational awareness. In particular, I would like to thank Mr. Nassy Srour and Dr. Tien Pham for their unwavering support and their confidence in me. I would also like to acknowledge the support provided me by the Signal and Image Processing Division.

I have a lot to be thankful for. I thank the Chinthakrindi Kanakayya Higher Secondary School, Mangalagiri; the Indian Institute of Technology, Kharagpur; and Boston University, for providing me free education and laying the foundation for my life in the field of beautiful science and engineering, and sparing me the grunt work of routine life. I thank all the teachers who touched my life with a magic wand and made me see the brighter future in education. I particularly thank Prof. Mark Karpovsky, Boston University, who was instrumental in my coming to the United States to pursue higher education.

I cannot thank enough Ms. Carol Johnson for taking on the burden of editing the whole book within a short time and doing a wonderful job. Because of her dedication and hard work, the book is in a decent form and easy to read.

I would also like to thank my brother Mr. Umamaheswara Rao Damarla who inculcated the discipline in me to read at an early age.

Finally, I thank my wife Mrs. Komala Bai Damarla profoundly for her patience and understanding all these years while I was writing the book. Her friendship, comfort, and persuasion were instrumental in writing this book.

Contents

Chapter 1
Introduction to Acoustics

All living creatures rely on their senses—sight, sound, touch, taste and smell-to interpret and interact with the world around them. A major function of our sensory organs is to provide situational awareness. We use our various sensory organs—eyes, ears, skin, tongue and noses—to identify and avoid dangers; detect and acquire sustenance; as well as to experience pleasure and social connectedness. Over time, humans and other living creatures have adapted in many ways to refine the way they use their sensory organs to best fit their circumstances and environments. For instance, humans use mainly their eyes, which are highly developed allowing for an acute field of vision, to observe the world around them, whereas bats, who have poor eyesight and operate mainly at night, rely on echolocation using specially developed auditory organs that allow them to use sound waves to navigate with great accuracy and locate their tiny prey (mosquitoes). The human brain has developed over time to process the signals from various sensory organs efficiently and effectively, and as such, various segments of the human brain are apportioned to each of our biologically based sensors, as shown in Fig. 1.1, which depicts the cortical sensory homunculus. While the image in Fig. 1.1 appears distorted, that is because the image reflect the amount of sensory nerve tissue within each region, which is not all equal, thus, the lips, tongue and finger tips, which are highly sensitive, appear the largest. While we use all of senses to process and relate to the world around us, the most prominent and most important sensory organs for humans are eyes and ears. This book is focused on the latter, particularly as it related to how this sense in uses on the battlefield.

1.1 Sound/Acoustics

Sound plays a vital role as the main source of communication for both humans and other living creatures. Sounds allow us to quickly locate and assess danger. We used sounds to alert ourselves and others about dangers to avoid them. Mechanically speaking, sounds (also referred to as acoustics) are pressure waves that disturb the tranquility of the medium in which they propagate. Imagine a still pond: when a

T. Damarla, *Battlefield Acoustics*, DOI 10.1007/978-3-319-16036-8_1

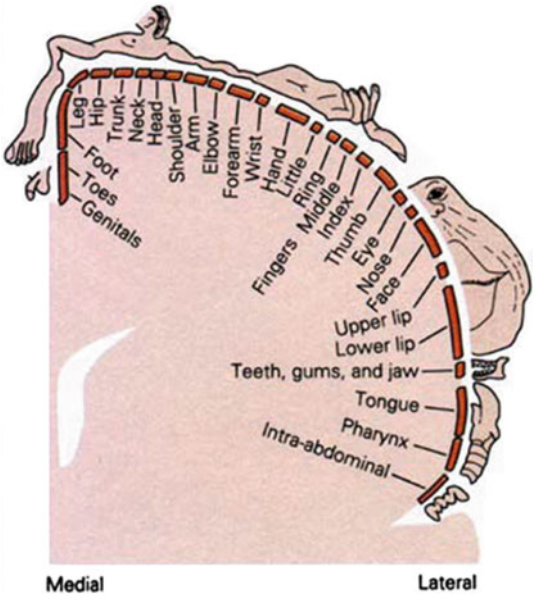


Fig. 1.1 Sensory homunculus depicting the functional space in the brain

pebble is dropped, ripples radiate from the location where the pebble hits the water. In much the same way, a sound creates waves in the surrounding medium, be it air, water, etc. These sound waves radiate outward from the source to the far corners of the available space. Mathematically, as these waves travel, the intensity S of the waves reduces proportional to the inverse of the distance d from the source:

$$S \propto 1/d. \tag{1.1}$$

Depending on the frequency, the sound waves can be categorized as infrasonic (low), audible or ultrasonic (high). For instance, dogs can hear sounds at higher frequencies (ultrasonic) than humans, as well as from farther distances. This capability allows dogs to be more alert compared to the humans, a trait humans value in dogs. In humans, sound is collected and processed by special sensory organs called ears. Their special shape is ideally suited for collecting sound waves, which impinge on a diaphragm that vibrates in response to sound wave pressure. These vibrations of the diaphragm are converted into electrical signals, which are processed by cochlea in the inner ear. These signals are then carried on to the nervous system, where the meanings of the sounds are interpreted. Acoustics is the study of vibration and sound, and is also a term often used interchangeably with "sound" to mean sound-based signals. In this book, we use the term acoustics in both senses of the word.

1.2 Sound and Its Applications

Acoustics have been a vital aspect in battlefield planning and execution. First reference to sound as an instrument of war to bring down the walls of Jericho using trumpets date back to biblical times [69]. In World War II the French used acoustics to detect the incoming enemy airplanes [69]. Sound is also used to communicate signals over long distances to alert troops about enemy locations and approach, etc. It can also be used to detect enemy soldiers walking in underground tunnels or entering forts in order to attack. In recent years, acoustics have been successfully used on the battlefield for various applications. Unlike other sensors such as radar and imaging sensors, which require line of sight, many of the transducers use to sense sound do not. This capability is exploited in the battlefield to determine the direction of the enemy fire and reposition other transducers such as video cameras in the right direction. The following are some of the battlefield applications of acoustics:

(a) **Hostile fire detection and localization:** Acoustics can be used to detect the direction of hostile fire and localize the source of the fire and, in the case of mortar or rocket launching, they can also be used to localize the impact. Localization of source helps to eliminate or mitigate the source, whereas localization of impact helps in sending the humanitarian help to the right location and assessing the damage.
(b) **Target detection:** In a battlefield, targets are often identified by the sound they make, whether intentional or unintentional. In majority of the cases, detection of sound implies detection of targets. Some typical targets are personnel, military vehicles (such as humvee, tanks, helicopters, airplanes), gun fire, etc.
(c) **Target Classification:** Classification is a process of analyzing sound waves and identifying the type of target. For example, some humans can tell the type of vehicle from the sound the vehicle is emitting; this is usually based on experience, but is also a result of the acuity of a person's hearing. Similarly, one can process the acoustic signals emitted by vehicles, tanks, airplanes, and helicopters to identify them as such.
(d) **Acoustics as a weapon:** It is possible to emit a focused high-pitched sound to disperse the hostile crowds. This method is non-lethal and often is a preferred way of handling crowds.
(e) **Monitoring sounding rockets or missiles:** When rockets or missiles are launched, they emit low frequency (infra-sound at approximately 1–20 Hz) signals, which can be detected from distances several hundreds of kilometers from the launching sites. As such, one can use these sonic emission to monitor rocket launch activities in countries that do not cooperate with international community.
(f) **underwater guided torpedos:** One can transmit sound signals and use their echoes as they bounce of submarines to locate them. The process of capturing and analyzing such signals is called sonar. Using sonar, one can locate the position of the submarine or torpedo, etc., much like the radar and track the submarine.
(g) **Robotics navigation system:** Ultrasonic systems can be used for navigational purposes in robots.

(h) **Ranging system:** Ultrasonic systems can be used to determine range of an object accurately.

1.3 Microphones

While we use our ears to sense and interpret the vibrations created by sound in the air, we can gain greater accuracy by turning to mechanical devices to mimic this function. Thus, we have developed several devices to aid us in the detection and processing of sound, foremost among them, the microphone. A microphone is a device that converts sound pressure into electrical signals in a way designed to mimic human ear. Depending on the construction and design of microphones, such devices can be classified into several categories [89], namely, (a) condenser microphones, (b) piezoelectric microphones, (c) dynamic microphones, (d) carbon microphones, (e) magnetic microphones and (f) microelectromechanical system (MEMS) microphones. Each type of microphone possesses different frequency characteristics. For example, a condenser or dynamic microphone's frequency response may vary from 3 to 20,000 Hz depending on the construction, while piezoelectric microphone may have frequency response between 300–340 Hz. We now provide a description of each type of microphone.

(a) **Condenser Microphone:** The design of the condenser microphone is shown in Fig. 1.2. Sound pressure moves the diaphragm of the microphone changing the capacitance between the two electrodes (back-plate and the diaphragm), which is captured by different configurations of the electrical circuit. One such configuration is shown in Fig. 1.2b. Here the electret is a permanently charged dielectric usually made of a 0.5–1.0 mil polymer film such as Teflon. The charge of the electret is chosen bias of the electric circuit that amplifies the signals received by the microphone.

(b) **Piezoelectric Microphone:** These microphones come in a variety of designs [89]. The piezoelectric microphone comprises one or two electrodes and a piezoelectric foil or crystal. Variations in design involve how these elements are arranged. One of the structure is shown in Fig. 1.3. When the sound pressure is applied to the structure in Fig. 1.3, the piezoelectric foil generates electricity that is carried out by the wires and applied to a preamplifier.

(c) **Dynamic Microphone:** This is one of the simplest type of microphones and can be easily built. The design is shown in Fig. 1.4. The sound pressure impinging on the diaphragm moves a coil in the magnetic flux which induces voltage in the coil that is proportional to the sound vibrations.

(d) **Carbon Microphone:** The carbon microphone's diaphragm is connected to movable electrodes (see Fig. 1.5). The movable electrodes and stationary electrodes form a cup in which carbon granules are stored and a polarizing current is passed through the granules. When the diaphragm moves, the resistance of the carbon granules changes and hence current transmitted. The sound vibrations are then converted into acoustic signals.

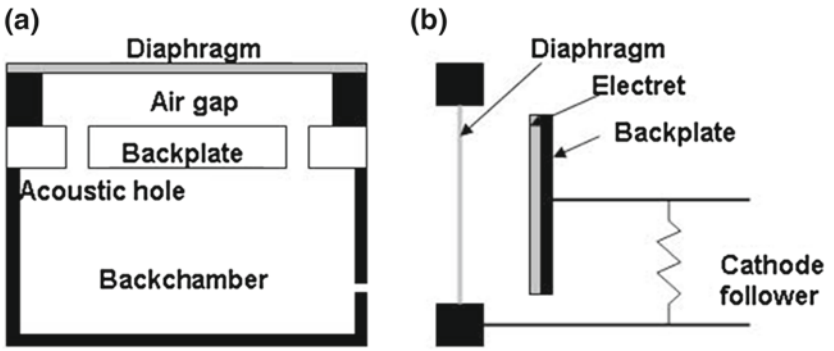


Fig. 1.2 Design of condenser microphone: **a** structure, **b** electrical structure

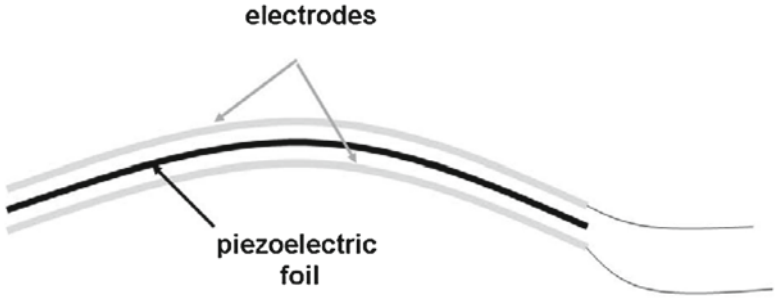


Fig. 1.3 Piezoelectric microphone components

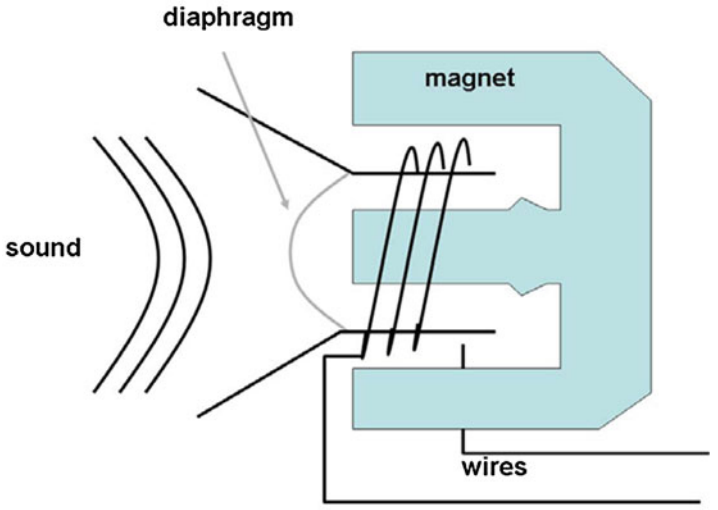


Fig. 1.4 Dynamic microphone design

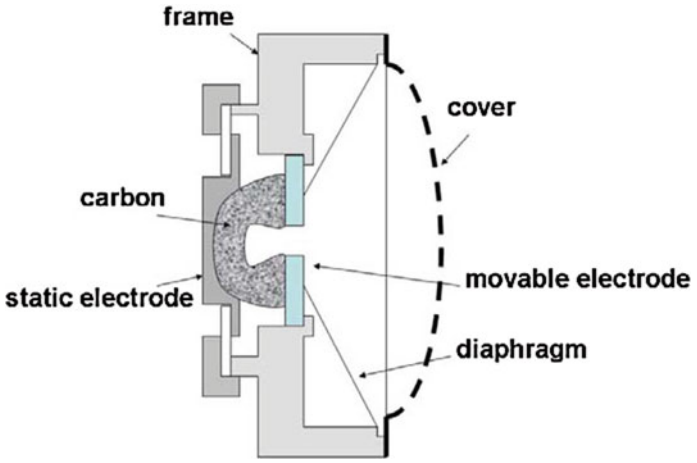
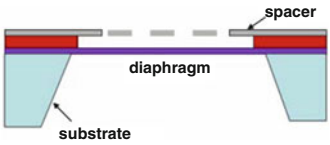


Fig. 1.5 Carbon microphone

Fig. 1.6 MEMS microphone



(e) **Magnetic Microphones:** Magnetic microphones are similar to dynamic microphones. In a magnetic microphone, the diaphragm moves an armature, which, in turn, varies the reluctance in the magnetic field. The variation of reluctance in the magnetic field changes the magnetic field, which leads to variation in the induced voltage.

(f) **MEMS Microphone:** Due to the advent of very large scale integrated (VLSI) circuit design, MEMS microphones have been developed by various vendors on silicon substrate. These microphones are low power consuming, and rugged and are useful in numerous applications. MEMS microphones are used in cell phones, toys, etc. The design of microphone comprises an etching diaphragm and back-plate on a silicon substrate, as shown in Fig. 1.6. The diaphragm and back-plate act as the plates of a capacitor and the capacitance changes with the diaphragm's motion due to sound pressure. The change in capacitance, in turn, changes the voltage, which is amplified using onboard circuitry.

(g) **Acoustic Vector Sensor:** A new class of microphones called acoustic vector sensors have been developed and are used to capture the vector to determine the direction of a target. These sensors measure the particle velocity of the air passing over them simultaneously in all three directions. Such a device has a small form factor (size) and reduced noise due to a small aperture. A company in Holland has successfully used such devices in tracking targets and localizing sound sources, such as gunshots.

1.4 Selecting the Right Microphone

Using microphones to listen for sounds in an area of interest are on rise. Along with video sensors, audio sensors (microphones) are widely used for better situational awareness. Depending on the application, microphones with different sensitivity and frequency characteristics need to be used. The sensitivity of the microphone determines how well the microphone responds to even the smallest sound level. The lowest amplitude a human ear can hear is 20 millionth of a Pascal (20 μPa.) Pascal is a unit of pressure equal to one Newton per square meter. Some of the reference sound levels are presented in Table 1.1.

Often sound pressure is measured in the decibel scale. The sound pressure level (SPL) or L_p is defined as

$$L_p = 20 \log_{10}\left(\frac{p}{p_{ref}}\right) \text{dB} \tag{1.2}$$

where p is the root mean square sound pressure and $p_{ref} = 20\,\mu\text{Pa}$ is the reference level. Clearly from Table 1.1, different events have different signal levels and as such the microphone recording the events should be selected so that it does not become saturated with sound and, yet at the same time it should be able to record the expected lowest signal. For recording music, the frequency response of the microphone should go well beyond 20 kHz. For recording gunfire data, the frequency response of the microphone should be higher than 50 kHz. A supersonic rifle generates both shock wave and muzzle blast signals. The shock wave signal levels exceed more than 140 dB while the muzzle blast signal changes with distance given by Eq. (1.1) and can be anywhere between 60 and 120 dB. Thus, a special microphone with a good dynamic range and frequency response greater than 50 kHz should be used for recording signatures of various gunshots.

In this chapter, we presented importance of sound in situational awareness in the battlefield. We presented some historical uses of sound in battlefield. We also presented different types of microphones in use and their characteristics. We showed the type of microphone to be used depends on the application at hand. There are several reference books [39, 89] that deal with various aspects of sound and interested reader should consult them.

Table 1.1 Sample references sound levels

Reference levels	
0 dB = 0.00002 Pa	Threshold of hearing
60 dB = 0.02 Pa	Business office
80 dB = 0.2 Pa	Shop noise
90 dB = 1 Pa	Large truck
100 dB = 24 Pa	Jack hammer
120 dB = 20 Pa	Airplane takeoff
140 dB = 200 Pa	Threshold of pain

Chapter 2
Basic Concepts in Probability

In this chapter, some basic concepts required for the detection and estimation of signals are presented from the theory of probability, statistics and random processes. The reader is assumed to have some basic knowledge of the theory of probability. Some concepts on multivariate statistics and random vectors are also presented. For a more thorough treatise of the subject, readers should consult the noted reference books [73, 76].

2.1 Probability Distributions and Densities of Random Variables

Papoulis [73] defines a random variable x as a real function whose domain is the probability space S and such that the following is true:

- The set $\{x \leq X\}$ is an event for any real number X.
- The probability of the events $\{x = +\infty\}$ and $\{x = -\infty\}$ equals zero.

A random variable is called either a continuous random variable or a discrete random variable depending on whether it takes real values or discrete values, respectively. The *cumulative distribution function* (*cdf*) F_x of a random variable x at point $x = X_0$ is defined as the probability that $x \leq X_0$:

$$F_x(X_0) = P(x \leq X_0). \tag{2.1}$$

Changing the value of X_0 from $-\infty$ to ∞ in (2.1) gives the whole *cdf* for all values of x. From the definition of the random variable and *cdf*, it follows directly that $F_x(-\infty) = 0$ and $F_x(\infty) = 1$. A more widely used function is the *probability density function* (*pdf*), which is defined for the continuous random variable x as the derivative of the cumulative distribution function:

T. Damarla, *Battlefield Acoustics*, DOI 10.1007/978-3-319-16036-8_2

$$p_x(x_0) = \frac{dF_x(x)}{dx}\Big|_{x=x_0} \tag{2.2}$$

The *pdf* is the probability that the variable has the value x. Often the probability density function is available or estimated in real applications. The cumulative distribution function can be computed from the *pdf* by the inverse relationship

$$F_x(X_0) = \int_{-\infty}^{X_0} p_x(\xi) d\xi. \tag{2.3}$$

One of the most widely used *pdf*s in signal processing is the normal distribution. It is characterized by two parameters, mean and variance. The normal distribution is given by

$$p_x(x) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right) \triangleq \mathcal{N}_x\left(\mu, \sigma^2\right) \tag{2.4}$$

where μ and σ^2 denotes the mean and the variance, respectively. Figure 2.1a shows the plot of the normal distribution for $\mu = 0$ and $\sigma = 1$ and the corresponding *cdf* is given in Fig. 2.1b. From the *cdf*, the probabilities of a random variable taking a value or a set of values can be obtained. For example, the probability of the random variable x taking the value less than or equal to 1.0 is found to be 0.84 from Fig. 2.1b, that is, $p(x \leq 1) = 0.84$.

There are several other distributions that are widely used in practice depending on the data collected and its statistical nature, including Bernoulli, Poisson, Rayleigh, uniform, Weibull distributions, to name few.

The mean (μ) and variance (σ^2) in (2.4) are also called the first and second moments and, in general, the kth moments are defined for continuous random variable as

$$\mu_k = E\{x^k\} = \int_{-\infty}^{+\infty} x^k p(x) dx \tag{2.5}$$

In the case of discrete random variable, the mean and the variances are computed as

$$\mu = \sum_{i=1}^{N} P(x_i)\, x_i \tag{2.6}$$

and

$$\sigma^2 = \sum_{i=1}^{N} P(x_i)(x-\mu)^2 \tag{2.7}$$

where N is the number of samples x_i, $i = \{1, 2, \ldots, N\}$. If the random variable x_i is equally likely, then $P(x_i) = \frac{1}{N}$, then (2.6) becomes

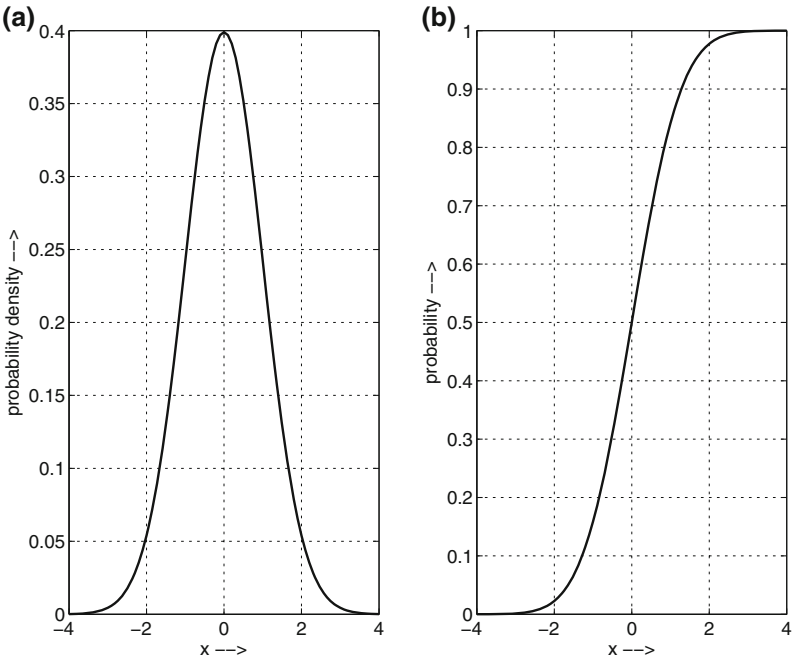


Fig. 2.1 **a** Normal distribution, **b** cumulative distribution

$$\mu = \frac{1}{N}\sum_{i=1}^{N} x_i \tag{2.8}$$

When the variance exists, then the standard deviation is given by σ. For a normal distribution with mean $\mu = 0$ and $\sigma = 1$, as shown in Fig. 2.1a, the probability that the random variable x lies between ± 1, that is, $p(-1 \leq x \leq 1)$ is given by the area under the curve in Fig. 2.1a lying between $x = -1$ and $x = 1$ and it can be shown to be 68.2 % of the area under the curve and $p\,(-2 \leq x \leq 2)$ is 95.4 % of the area under the curve. This fact is often used in signal processing while selecting thresholds for various parameters.

Let $g(x)$ is a function of analog random variable x, then the expectation of $g(x)$ is denoted by $E\,[g(x)]$ and is defined by

$$E\,[g(x)] = \int_{-\infty}^{+\infty} g(x)\,p(x)dx \tag{2.9}$$

2.2 Bernoulli Distribution

Suppose an experiment is conducted with its outcome determined as "success" or "failure". Let the probability of success is denoted by p, then the probability of failure is $(1-p)$. An example of such an experiment is tossing of a coin, where if the coin toss results in "heads" it is called success and if the toss results in "tails" it is called failure. Other experiments could be the outcome of surgery, throwing a ball through hoops, the transmission of a disease, etc. Thus, the Bernoulli distribution is defined as

$$f(x) = p^x (1-p)^{1-x}, \quad \text{for } x = 0, 1. \tag{2.10}$$

If "n" tosses of a coin are done, then the probability that "k" successive tosses result in heads is given by the probability mass function

$$P(X = k) = \binom{n}{k} p^k (1-p)^{n-k}. \tag{2.11}$$

2.3 Random Vectors

In general, more useful cases arise when there are multiple random variables. For example, in order to classify a target, one may consider several features that are pertinent to the target. These features can be denoted as a random vector

$$\mathbf{X} = (x_1, x_2, \ldots, x_n)^T \tag{2.12}$$

where T denotes the transpose and n is the number of elements in X. Just as in the case of a single variable, the *cdf* of a vector $\mathbf{X}$ is defined by

$$F_{\mathbf{X}}(\mathbf{X}_0) = P(\mathbf{X} \leq \mathbf{X}_0) \tag{2.13}$$

where $P(.)$ denotes the probability and $\mathbf{X}_0$ is some constant vector. The multivariate probability density function $p_{\mathbf{X}}(\mathbf{X})$ of $\mathbf{X}$ is defined as the derivative of the cumulative distribution function $F_{\mathbf{X}}(\mathbf{X}_0)$ with respect to the components of $\mathbf{X}$:

$$p_{\mathbf{X}}(\mathbf{X}_0) = \frac{\partial}{\partial x_1}\frac{\partial}{\partial x_2}\cdots\frac{\partial}{\partial x_n} F_{\mathbf{X}}(\mathbf{X})\Big|_{\mathbf{X}=\mathbf{X}_0} \tag{2.14}$$

2.3.1 Joint and Marginal Distributions

Let us assume that there are two different sensors, say, acoustic and seismic sensors. Let the feature vector for the acoustic sensor be denoted by the random vector

$\mathbf{X} = (x_1, x_2, \ldots, x_n)^T$ of length 'n' and the seismic sensor be denoted by the random vector $\mathbf{Y} = (y_1, y_2, \ldots, y_m)^T$ of length 'm', then the joint distribution function of two random vectors $\mathbf{X}$ and $\mathbf{Y}$ is given by

$$F_{\mathbf{X},\mathbf{Y}}(\mathbf{X}_0, \mathbf{Y}_0) = P(\mathbf{X} \leq \mathbf{X}_0, \mathbf{Y} \leq \mathbf{Y}_0) \tag{2.15}$$

for some fixed vectors $\mathbf{X}_0$ and $\mathbf{Y}_0$. Let $\mathbf{Z} = \left(\mathbf{X}^T, \mathbf{Y}^T\right)$ be the super vector, then the joint density function $p_{\mathbf{Z}}(\mathbf{Z}) = p_{\mathbf{X},\mathbf{Y}}(\mathbf{X}, \mathbf{Y})$ of $\mathbf{X}$ and $\mathbf{Y}$ is obtained by differentiating the joint distribution function $F_{\mathbf{X},\mathbf{Y}}(\mathbf{X}, \mathbf{Y})$ as in (2.14) with respect to all the components of $\mathbf{X}$ and $\mathbf{Y}$. Hence,

$$F_{\mathbf{X},\mathbf{Y}}(\mathbf{X}_0, \mathbf{Y}_0) = \int_{-\infty}^{\mathbf{X}_0} \int_{-\infty}^{\mathbf{Y}_0} p_{\mathbf{X},\mathbf{Y}}(\eta, \xi)\, d\eta d\xi \tag{2.16}$$

The density functions with respect to a subset of $\mathbf{Z}$ are called the marginal density and are denoted by $p_{\mathbf{X}}(\mathbf{X})$ for $\mathbf{X}$ and $p_{\mathbf{Y}}(\mathbf{Y})$ for $\mathbf{Y}$ and given by

$$p_{\mathbf{X}}(\mathbf{X}) = \int_{-\infty}^{+\infty} p_{\mathbf{X},\mathbf{Y}}(\mathbf{X}, \eta)\, d\eta \tag{2.17}$$

$$p_{\mathbf{Y}}(\mathbf{Y}) = \int_{-\infty}^{+\infty} p_{\mathbf{X},\mathbf{Y}}(\xi, \mathbf{Y})\, d\xi \tag{2.18}$$

2.3.2 Mean Vector and Correlation Matrix

Some of the general features of the target used for classification are the mean values of the individual features. Using (2.5)

$$\mathbf{m}_{\mathbf{X}} = \mu_{\mathbf{X}} = E[X] = \int_{-\infty}^{+\infty} \mathbf{X}\, p_{\mathbf{X}}(\mathbf{X})\, d\mathbf{X} \tag{2.19}$$

The mean vector $\mathbf{m}_{\mathbf{X}} = \left(m_{x_1}, m_{x_2}, \ldots, m_{x_n}\right)^T$ and the individual components are given by

$$m_{x_i} = E[x_i] = \int_{-\infty}^{+\infty} x_i\; p_{\mathbf{X}}(\mathbf{X})\, d\mathbf{X} = \int_{-\infty}^{+\infty} x_i\; p_{x_i}(x_i)\, dx_i \tag{2.20}$$

where $p_{x_i}(x_i)$ is the marginal density of the ith component x_i of $\mathbf{X}$. Equation (2.20) is true due to the fact that the integrals over the other components of $\mathbf{X}$ reduce to unity by the definition of the marginal density given by (2.17) and (2.18).

As shown in Fig. 2.1, the variance gives a measure of the spread of the data or the random variable. Similarly, covariance of two random variables give the measure how two variables vary together. The covariance of two random variables is defined as expectation

$$C_{x_i,x_j} = cov(x_i, x_j) = E\left[(x_i - \mu_i)(x_j - \mu_j)\right] \tag{2.21}$$

where μ_i and μ_j denotes the mean values of random variable x_i and x_j, respectively. The covariance matrix $cov(x_i, x_j)$ is a 2×2 matrix whose ijth element is $E\left[(x_i - \mu_i)\left(x_j - \mu_j\right)\right]$. Note that the diagonal elements are the variances of the individual random variables $E\left[(x_i - \mu_i)^2\right]$. The off-diagonal elements $E\left[(x_i - \mu_i)\ \left(x_j - \mu_j\right)\right]$, $i \neq j$, are called the covariance of x_i and x_j.

If the random vector Y is a linear transform of another vector X

$$Y = AX$$

then

$$E[Y] = AE[X] \tag{2.22}$$

and

$$cov(Y) = A\ cov(X)\ A^T \tag{2.23}$$

Another statistical parameter that is closely related to the covariance is the correlation and it is given by the second moment

$$r_{ij} = E\left[x_i x_j\right] = \int_{-\infty}^{+\infty} x_i x_j\ p_{\mathbf{X}}(\mathbf{X}) = \int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty} x_i x_j\ p_{x_i,x_j}\left(x_i, x_j\right) dx_j dx_i \tag{2.24}$$

If $\mathbf{X} = (x_1, x_2, \ldots, x_n)^T$, then the *correlation matrix* is given by

$$R_{\mathbf{X}} = E\left[\mathbf{X}\mathbf{X}^T\right] \tag{2.25}$$

and the component r_{ij} in ith row and jth column of matrix $R_{\mathbf{X}}$ is the correlation between x_i and x_j. When $\mathbf{m}_{\mathbf{X}} = 0$ the correlation and covariance matrices become identical, that is, $R_{\mathbf{X}} = C_{\mathbf{X}}$.

***Uncorrelatedness and Whiteness*:** The random vectors X_1 and X_2 are said to be uncorrelated if the cross-correlation matrix C_{X_1,X_2} is a zero matrix:

$$C_{X_1,X_2} = cov(X_1, X_2) = E[(X_1 - m_1)(X_2 - m_2)] = 0 \tag{2.26}$$

It can be shown that for uncorrelated random vectors, the correlation matrix is

$$R_{X_1,X_2} = E\left[X_1 X_2^T\right] = E[X_1]\, E\left[X_2^T\right] = m_{X_1} m_{X_2}^T \tag{2.27}$$

In particular if the random vectors have zero mean and unit covariance then they are said to be white, that is,

$$C_{X_1,X_2} = R_{X_1,X_2} = I; \qquad m_{X_1} = m_{X_2} = 0. \tag{2.28}$$

2.3.3 Independence of Variables

The independence of random variables is a very important concept that is often invoked in signal processing to simplify matters. Two random variables x and y are said to be independent if

$$P\left(x \leq x_0,\ y \leq y_0\right) = P\left(x \leq x_0\right)\ P\left(y \leq y_0\right) \tag{2.29}$$

Similarly, multiple random variables are said to be independent if

$$P\left(x \leq x_0,\ y \leq y_0, \cdots, z \leq z_0\right) = P\left(x \leq x_0\right) P\left(y \leq y_0\right) \cdots P\left(z \leq z_0\right) \tag{2.30}$$

2.3.4 Conditional Probabilities and Bayes Theorem

Let X and Y be two random events with probabilities $P(X) > 0$ and $P(Y) > 0$, then the probability of X assuming Y, is defined as

$$P(X|Y) = \frac{P(XY)}{P(Y)} \tag{2.31}$$

Rewriting the above equation, we get the product rule

$$P\left(A, B\right) = P\left(A|B\right) P\left(B\right). \tag{2.32}$$

We can extend this for three variables:

$$P\left(A, B, C\right) = P\left(A|B, C\right) P\left(B|C\right) P\left(C\right). \tag{2.33}$$

In general, the above equation can be extended to n variables to obtain the chain rule

$$P\left(A_1, A_2, \cdots, A_n\right) = P\left(A_1|A_2, \cdots, A_n\right) P\left(A_2|A_3, \cdots, A_n\right) \cdots P\left(A_{n-1}|A_n\right) P\left(A_n\right) \tag{2.34}$$

2.4 Cross Correlation

Cross correlation is a widely used concept in signal processing to determine the time difference between two signals recorded by two microphones in an array of microphones—often used to estimate the direction of arrival angle of the signal. The correlation between two signals $f(t)$ and $g(t)$ is defined as

$$r_{fg} = (f \star g)(t) \triangleq \int_{-\infty}^{\infty} f^*(\tau) g(t+\tau)\, d\tau, \tag{2.35}$$

where $f^*(t)$ is the complex conjugate of $f(t)$. In signal processing, the cross correlation is a measure of similarity between two waveforms (or signals) as a function of time lag. For discrete signals the cross correlation is given by

$$r_{fg} = (f \star g)[n] \triangleq \sum_{m=-\infty}^{m=\infty} f^*[m]\, g[n+m]. \tag{2.36}$$

The following example shows cross correlation of two signals and estimation of time delay between the two signals.

Example: Let $f[n]$ is the original signal and $g[n]$ is the delayed and attenuated signal without any noise, as shown in Fig. 2.2a, b, respectively. Each signal is of $N_s = 2048$ samples long. Their cross correlation r_{fg} is shown in (c) with cross correlation lag. The signals are sampled at a sampling rate of $f_s = 20{,}000$ samples/s. In order to compute the time delay between the two signals find the index k at which r_{fg} is

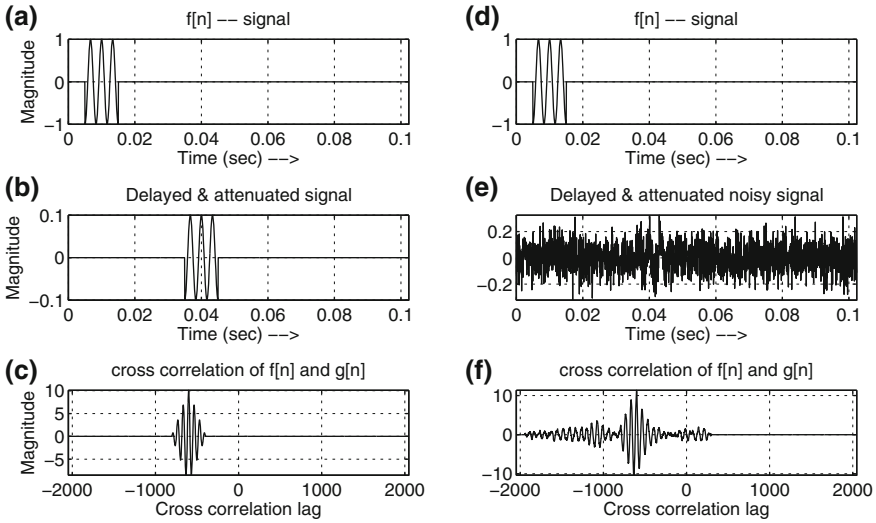


Fig. 2.2 Cross correlation of signals

maximum, then $(N_s - k)/f_s$ gives the time delay between the signals f and g. For the signals shown in Fig. 2.2a, b, the delay is estimated to be 0.03 s. A new signal $z(t) = g(t) + n(t)$ is generated by adding noise $n(t)$ to the signal $g(t)$ and is shown in Fig. 2.2e. The cross correlation r_{fz} is shown in Fig. 2.2f and the time delay is computed. The time delay is found to be 0.03 s. In spite of the signal $z(t)$ being very noisy, we are able to correctly estimate the time delay between the two signals using the cross correlation technique.

Chapter 3
Detection Theory

Target detection, that is, the determination of a target's presence or lack of presence is fundamental to battlefield acoustics. As such, target detection is a process of detecting the signals emitted by the target. The signals travel through a medium such as air or through water and reach the sensors. The sensor captures the signals and makes a determination whether there is signal belonging to a target or not. When the signal travels through the medium, it gets attenuated depending on the distance it travels and how much turbulence in the air affects it, as shown in Fig. 3.1. As a result, the signal-to-noise ratio (SNR) decreases, making detection of the signal difficult.

Consider the case where data from an acoustic sensor are available. The data are modeled as a random process because all aspects of the data are not describable with certainty. For example, the acoustic sensor data will be different depending on whether there is a target or not and it is not known with certainty that the target is present. Moreover, the data are corrupted with noise, which is unpredictable. So, the problem is to determine which of a number of situations gave rise to the data that the sensor observed. That is, the signal detection problem can be cast as *M-ary hypothesis testing*, in which the observed data are determined to have been generated due to one of the M possible statistical situations [76]. In particular, the case $M = 2$ is called the binary hypothesis testing. Let the two possible hypotheses, H_0 and H_1, belong to two probability distributions, P_0 and P_1, respectively, on the observation space (Γ, G). Thus the problem is to determine

$$\begin{aligned} H_0 &: Y \sim P_0 \\ H_1 &: Y \sim P_1, \end{aligned} \tag{3.1}$$

where the notation "$Y \sim P$" denotes the condition that "Y" has distribution "P". The hypothesis H_0 is referred to as *null* hypothesis and H_1 as the alternative hypothesis. A decision rule (or hypothesis test) δ partitions the observation space G in to two subspaces Γ_0 and Γ_1, as shown in Fig. 3.2, one corresponding to H_0 and another to H_1, respectively, such that

T. Damarla, *Battlefield Acoustics*, DOI 10.1007/978-3-319-16036-8_3

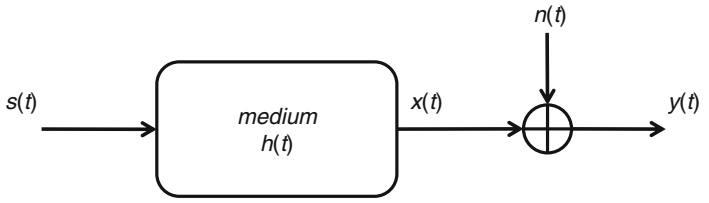


Fig. 3.1 Signal transformation due to medium

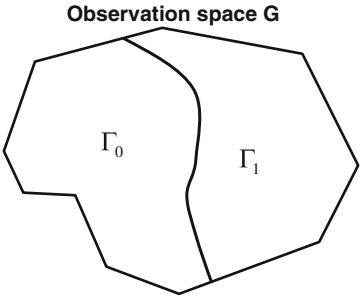


Fig. 3.2 Partitioning of observation space

$$\delta(y) = \begin{cases} 1, \text{ if } y \in \Gamma_1 \\ 0, \text{ if } y \in \Gamma_0. \end{cases} \tag{3.2}$$

Moreover, selection of Γ_1 is done in some optimal way to minimize the number of missed detections. Optimization of the decision process becomes minimization of the cost associated with each decision. Let, C_{ij}, for $i = 0, 1$ and $j = 0, 1$, is a non-negative cost associated with selection of hypothesis H_i when H_j is true. Then the conditional risk $R_j(\delta)$ for each hypothesis is the average or expected cost incurred by decision rule δ when the hypothesis is true, that is,

$$\begin{aligned} R_0(\delta) &= C_{10}P(D_1|H_0) + C_{00}P(D_0|H_0) \text{ when } H_0 \text{ true}, \\ R_1(\delta) &= C_{11}P(D_1|H_1) + C_{01}P(D_0|H_1) \text{ when } H_1 \text{ true}. \end{aligned} \tag{3.3}$$

Clearly, R_j is the cost of choosing H_1 when H_j is true times the probability of doing so, that is, the probability of making the decision D_1, plus the cost of choosing H_0 when H_j is true times the probability of making the decision D_0. Decision D_j is made when observed data $y \in \Gamma_j$. Further, if the *a prior* probabilities for the occurrences of H_0 and H_1 are π_0 and $\pi_1 = 1 - \pi_0$, then the average or *Bayes risk* for overall average cost incurred by the decision rule δ is defined as

$$r(\delta) = \pi_0 R_0(\delta) + \pi_1 R_1(\delta), \tag{3.4}$$

and the priors are assumed to be known. The optimum decision rule for H_0 versus H_1 is the one that minimizes the Bayes risk and such a rule is called *Bayes rule*. Combining (3.3) and (3.4) gives

$$r(\delta) = \pi_0 \left[C_{10} P(D_1|H_0) + C_{00} P(D_0|H_0)\right] + \pi_1 \left[C_{11} P(D_1|H_1) + C_{01} P(D_0|H_1)\right] \tag{3.5}$$

Since the decision D_j is made whenever data y is in the region Γ_j, the probabilities

$$\begin{aligned} P\left(D_1|H_0\right) &= \int_{\Gamma_1} p_0(y) dy, \\ P\left(D_1|H_1\right) &= \int_{\Gamma_1} p_1(y) dy. \end{aligned} \tag{3.6}$$

Using the fact that $P(D_0|H_0) = 1 - P(D_1|H_0)$ and $P(D_0|H_1) = 1 - P(D_1|H_1)$ in (3.5) and substituting (3.6), the average cost is

$$r(\delta) = \pi_0 C_{00} + \pi_1 C_{01} + \pi_0 P(D_1|H_0)\left(C_{10} - C_{00}\right) - \pi_1 P(D_1|H_1)\left(C_{01} - C_{11}\right) \tag{3.7}$$

or

$$r(\delta) = \pi_0 C_{00} + \pi_1 C_{01} + \int_{\Gamma_1} \left[\pi_0 \left(C_{10} - C_{00}\right) p_0(y) - \pi_1 \left(C_{01} - C_{11}\right) p_1(y)\right] dy \tag{3.8}$$

The region Γ_1 is selected so as to minimize the average risk, that is, select the data points y that will be included in the region of integration Γ_1. In general, the cost of making wrong decision is higher than making correct decision, that is,

$$C_{i \neq j, j} - C_{j,j} > 0, \quad \text{all } i, j. \tag{3.9}$$

This implies that the second term in the integrand should be larger than the first term, then only the overall $r(\delta)$ will be small. In other words, include in Γ_1 all the data points y that make the second part of integrand larger than the first part. That is, put in Γ_1 all points y for which

$$\lambda(y) = \frac{p_1(y)}{p_0(y)} > \frac{\pi_0 \left(C_{10} - C_{00}\right)}{\pi_1 \left(C_{01} - C_{11}\right)} = \lambda_t. \tag{3.10}$$

In the binary decision case, if the decision is made in favor of H_1 when in fact H_0 is true. This type of error is called *Type I error, or error of the first kind*. The probability $P_f = P(D_1|H_0)$, also called probability of false alarm since there is no target present. On the other hand, if the decision is made in favor of H_0 when in fact H_1 is true, it is called *Type II error, or error of second kind*. Type II error is made with probability $P_m = P(D_0|H_1)$. Type II error also called *missed detection*. The probability of detecting a target when it is $P_d = P(D_1|H_1) = 1 - P(D_0|H_1)$.

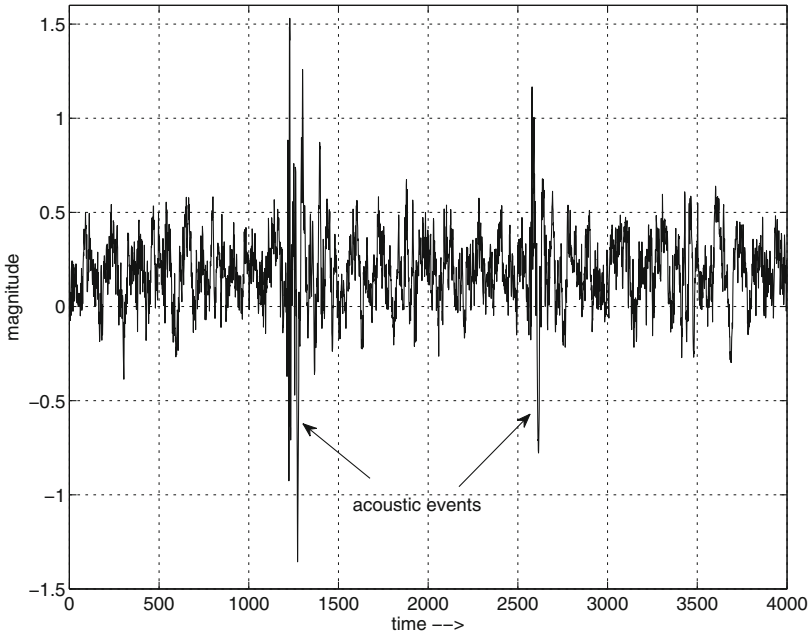


Fig. 3.3 Acoustic Event

The quantity

$$L(y) = \lambda(y) = \frac{p_1(y)}{p_0(y)}, \quad y \in \Gamma_1, \tag{3.11}$$

is known as the *likelihood ratio* between H_0 and H_1. The Bayes rule for (3.1) is

$$\delta_B(y) = \begin{cases} 1 \text{ if } L(y) \geq \lambda_t \\ 0 \text{ if } L(y) < \lambda_t. \end{cases} \tag{3.12}$$

Example : Detection of Signal in Gaussian Noise

Consider the case where a microphone is used to detect an acoustic event such as gun fire, or some other event where there is a short loud sound burst. One such a signal captured by a microphone is shown in Fig. 3.3. When there is no acoustic event, the microphone records the noise due to wind, the rustle of leaves from nearby trees, etc. Let us assume the noise is Gaussian with an average peak to peak amplitude of μ_0 and variance σ_0^2. Whenever the acoustic event occurs, it occurs with a mean peak to peak amplitude of μ_1 and variance σ_1^2. It is assumed that the random variable y takes the values μ_0, μ_1 with probabilities (priors) P_0 and P_1, respectively. Then the hypothesis pair can be written as

$$\begin{aligned} H_0 &: Y \sim \mathcal{N}\left(\mu_0, \sigma_0^2\right) \\ H_1 &: Y \sim \mathcal{N}\left(\mu_1, \sigma_1^2\right) \end{aligned} \tag{3.13}$$

where $\mathcal{N}\left(\mu, \sigma^2\right)$ denotes the Gaussian (or normal) distribution with mean μ and variance σ^2 and the Gaussian probability density function is given by

$$p(x|\mu) = \frac{1}{\sqrt{2\pi}\,\sigma} \exp -\frac{(x-\mu)^2}{2\sigma^2}, \quad x \in \mathbb{R}$$

The detection strategy is to choose μ_0 or μ_1 depending on which has the higher probability given the received value y, that is, choose μ_1 if

$$p\left(\mu_1|y\right) > p\left(\mu_0|y\right). \tag{3.14}$$

This is called the *maximum a posteriori probability* (MAP) criterion. Using the prior probabilities P_0, P_1 of the noise and acoustic event, the Bayes' rule gives

$$\begin{aligned} p(\mu_0|y) &= \tfrac{p(y|\mu_0)P_0}{p(y)} \\ p(\mu_1|y) &= \tfrac{p(y|\mu_1)P_1}{p(y)} \end{aligned} \tag{3.15}$$

The decision rule (3.14) or the likelihood ratio for (3.13) is given by

$$L(y) = \frac{p(\mu_1|y)}{p(\mu_0|y)} = \frac{p(y|\mu_1)P_1}{p(y|\mu_0)P_0} > 1. \tag{3.16}$$

$$\begin{aligned} L(y) &= \frac{\frac{P_1}{\sqrt{2\pi}\,\sigma_1} \exp -\frac{(y-\mu_1)^2}{2\sigma_1^2}}{\frac{P_0}{\sqrt{2\pi}\,\sigma_0} \exp -\frac{(y-\mu_0)^2}{2\sigma_0^2}} \\ &= \tfrac{P_1\sigma_0}{P_0\sigma_1} \exp\left\{-\tfrac{1}{2\sigma_0^2\sigma_1^2}\left[\sigma_0^2\left(y-\mu_1\right)^2 - \sigma_1^2\left(y-\mu_0\right)^2\right]\right\} \end{aligned} \tag{3.17}$$

Equation (3.17) can be further simplified if the variances are equal, that is, $\sigma_0^2 = \sigma_1^2$. In general, the variances are equal because it depends on the noise, which is random in nature. However, if they are not equal, one can make them equal by dividing the data corresponding to the noise by σ_0 and the data corresponding to the acoustic event by σ_1. Dividing the data by σ makes it unit variance, that is, $\sigma = 1$. Now, with $\sigma = \sigma_0 = \sigma_1$, the likelihood ratio becomes

$$L(y) = \frac{P_1}{P_0} \exp\left\{\frac{\mu_1 - \mu_0}{\sigma^2}\left(y - \frac{\mu_0 + \mu_1}{2}\right)\right\}. \tag{3.18}$$

The Bayes test for (3.13) for the case $\sigma_0 = \sigma_1 = \sigma$ becomes

$$\delta(y) = \begin{cases} 1, \text{ if } \frac{P_1}{P_0} \exp\left\{\frac{\mu_1-\mu_0}{\sigma^2}\left(y - \frac{\mu_0+\mu_1}{2}\right)\right\} \geq 1 \\ 0, \text{ otherwise} \end{cases} \tag{3.19}$$

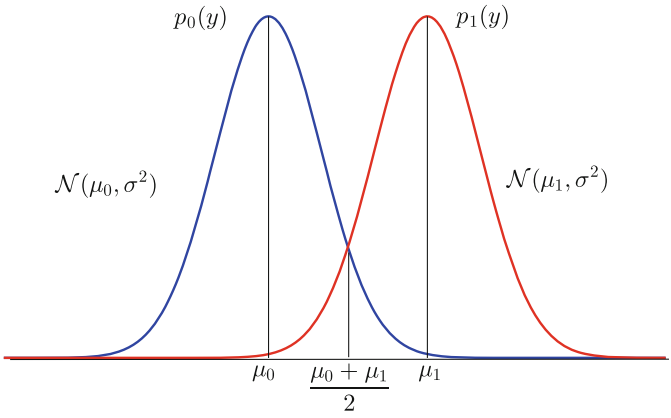


Fig. 3.4 Detection example

Taking the logarithms of the inequality in (3.19)

$$\frac{\mu_1 - \mu_0}{\sigma^2}\left(y - \frac{\mu_1 + \mu_0}{2}\right) > \ln\left(\frac{P_0}{P_1}\right)$$

or

$$y > \ln\left(\frac{P_0}{P_1}\right)\ \frac{\sigma^2}{\mu_1 - \mu_0} + \frac{\mu_1 + \mu_0}{2}. \tag{3.20}$$

results in a new Bayes test

$$\delta(y) = \begin{cases} 1, \text{ if } y \geq \tau' \\ 0, \text{ otherwise} \end{cases} \tag{3.21}$$

where $\tau' = \ln\left(\frac{P_0}{P_1}\right)\ \frac{\sigma^2}{\mu_1 - \mu_0} + \frac{\mu_1 + \mu_0}{2}$. If the priors P_0 and P_1 are equal, then $\tau' = \frac{\mu_1 + \mu_0}{2}$. Then the Bayes test checks the average of μ_0 and μ_1 and decides in favor of H_1 if the value of y is greater or equal to average; otherwise, it will decide in favor of H_0, as shown in Fig. 3.4. The Bayes risk can be computed from (3.8) and noting that $\Gamma_1 = \left\{y \in \mathbb{R} | y \geq \tau'\right\}$,

$$\begin{aligned} P\left(D_j | H_1\right) &= \int_{\tau'}^{\infty} p_j(y) dy = 1 - \Phi\left(\frac{\tau' - \mu_j}{\sigma}\right) \\ &= \begin{cases} 1 - \Phi\left(\frac{1}{d}\ln\left(\frac{P_0}{P_1}\right) + \frac{d}{2}\right), & j = 0 \\ 1 - \Phi\left(\frac{1}{d}\ln\left(\frac{P_0}{P_1}\right) - \frac{d}{2}\right), & j = 1, \end{cases} \end{aligned} \tag{3.22}$$

where Φ denotes the *cdf* of the $\mathcal{N}(0, 1)$ random variable, and $d = (\mu_1 - \mu_0)/\sigma$. For the case $\tau = 1$ the Bayes risk is

$$r(\delta) = 1 - \Phi(d/2). \tag{3.23}$$

In other words, the probability of false alarm is

$$\begin{aligned} P_f = P(D_1|H_0) = P(y > \tau'|H_0) &= \int_{\tau'}^{\infty} \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(y-\mu_0)^2}{2\sigma^2}\right) dy \\ &= \int_{(1/d)\ \ln(P_0/P_1)+d/2}^{\infty} \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{u^2}{2}\right) du \end{aligned} \tag{3.24}$$

where $u = (y - \mu_0)/\sigma$ and the probability of missed detection is

$$\begin{aligned} P_M = P(D_0|H_1) &= P(y \le \tau'|H_1) \\ &= \int_{-\infty}^{\tau'} \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(y-\mu_1)^2}{2\sigma^2}\right) dy \\ &= \int_{-\infty}^{(1/d)\ \ln(P_0/P_1)-d/2} \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{u^2}{2}\right) du \end{aligned} \tag{3.25}$$

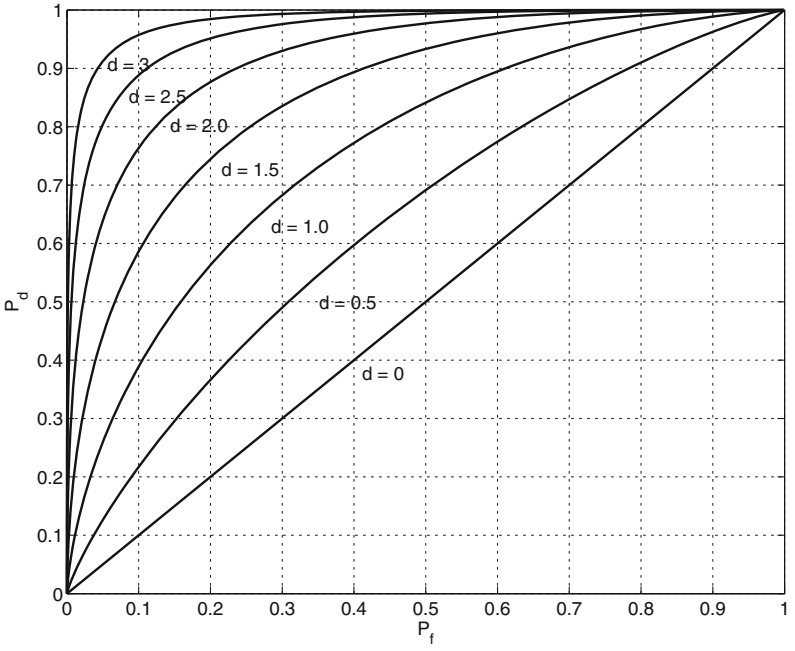


Fig. 3.5 ROC curve for detection problem as in Eqs. (3.24) and (3.25) with $P_0 = P_1$

Note that the probability of detection $P_D = 1 - P_M$. Both the Eqs. (3.24) and (3.25) depend only on d for given priors. The parameter $d = (\mu_1 - \mu_0)/\sigma$ can be taken as the SNR for this detection problem. The receiver operating characteristic (ROC) curves (Eqs. (3.24) and (3.25)) for different SNRs are plotted in Fig. 3.5.

3.1 Neyman-Pearson Criterion

In the previous section, concepts on probability of detection, probability of missed detection and probability of false alarm are presented. The concept of cost function and minimization of cost function to optimize the correct probability of detection are also presented. However, to minimize the cost function, one would often require the priors p_0 and p_1. In majority of the cases, the priors are not readily available. The Neyman-Pearson Criterion is an alternative approach used for optimizing the detection rule δ, where a bound is placed on the false-alarm probability:

$$\max_{\delta} P_d(\delta) \quad \text{subject to} \quad P_f(\delta) \leq \alpha \tag{3.26}$$

where α is the bound on the probability of false alarm. We use the Neyman-Pearson Criterion in the Chap. 12 to fuse the information from multiple detectors to achieve required false alarm rate.

Chapter 4
Estimation Theory

This chapter introduces some of the fundamental mathematical concepts essential for parameter estimation that are widely used in the field of battlefield acoustics. For example, in order to track acoustic targets that generate continuous signals, one may need to estimate the bearing angles of the target using several sensors and triangulate to estimate the position of the target. Usually, bearing estimation involves determining (estimating) the phase angles at various frequencies. In the case of transient events such as gunshots, localization of the source involves estimation of the time of arrival of the acoustic event at various microphones. The treatment presented here is only a fraction of the theory that is needed for the battlefield acoustics. However, interested people should consult some of the books [58, 76, 87] presented in the references on the subject. Estimation theory has its roots in astronomy. Several astronomers observing the celestial body movements found errors in their observations and wanted to determine which were the correct values out of a variety of measurements. The estimation problem deals with inferring the values of the parameters from the measurement data. Often measurements are prone to errors due either to human or instrument errors. Human errors are often random and independent, while the instrument errors are fixed and result in bias in the estimation.

Earlier analysis of the measurement data resulted in the formulation of the arithmetic mean [75] as the arithmetic average $\bar{x}$ of the independent observations x_i given by

$$\bar{x} = \frac{1}{N} \sum_{i=1}^{N} x_i \,. \tag{4.1}$$

where N is the number of measurements. The estimation problem is divided into two broad classes depending on whether the parameters to be estimated are deterministic constants or random. We discuss this issue more in later chapters. Moreover, estimators can be categorized as batch or on-line type. For example, if all the measurements are available for processing, then it is a matter of applying the processing technique

T. Damarla, *Battlefield Acoustics*, DOI 10.1007/978-3-319-16036-8_4

Fig. 4.1 Picture of Reverend Thomas Bayes

to all the data to get the estimation of the parameter. For example, summing all the measurements and dividing the sum by the number of measurements gives the estimate of the mean for the variable as in (4.1).
In the case of on-line estimation, also called adaptive estimation, all measurements are not available. The estimation of the parameters are updated as the measurements become available. The mean of the variable can be estimated using [51]

$$\bar{x} = \frac{j-1}{j}\bar{x}_{j-1} + \frac{1}{j}x_j \tag{4.2}$$

as the jth measurement available.
The seminal paper that laid foundation to the modern day estimation theory is the posthumous publication of the paper on "An Essay Towards Solving a Problem in the Doctrine of Chances" by Thomas Bayes[1] (see Fig. 4.1). In that famous paper, Bayes poses the problem as

> Given the number of times in which an unknown event has happened and failed: Required the chance that the probability of its happening in a single trail lies some where between any two degrees of probability that can be named.

This is the inversion problem, which was not considered until his publication. To put it in a mathematical framework: given the measurement data $\mathbf{Z}^T = (z_1, z_2, \ldots, z_m)$

[1] http://www.stat.ucla.edu/history/essay.pdf or http://canoe.ens.fr/~ebrian/s1h-dhsrb/1764-bayes.pdf.

determine the a posteriori density function of the unknown parameters $\mathbf{X}$ which characterizes the probability distribution from which the data were obtained. From Bayes rule, the a posteriori density function $p\,(\mathbf{X}|\mathbf{Z})$ is given by

$$p\,(\mathbf{X}|\mathbf{Z}) = \frac{p\,(\mathbf{Z}|\mathbf{X})\;\; p(\mathbf{X})}{p(\mathbf{Z})} \tag{4.3}$$

where $p(\mathbf{X})$ and $p\,(\mathbf{Z}|\mathbf{X})$ are the a priori density functions.
Bayes rule also provided the framework for maximum likelihood estimators. For example, consider the likelihood density function $p\,(\mathbf{Z}|\mathbf{X})$, to optimize it with respect to the parameters $\mathbf{X}$, consider its derivative and set it to zero

$$\frac{\partial p\,(\mathbf{Z}|\mathbf{X})}{\partial \mathbf{X}} = 0 \tag{4.4}$$

and solve for $\mathbf{X}$, which would be $\hat{\mathbf{X}}_{ML}$. This is discussed further in later sections.

Another important estimator is the method of *least-squares* that was developed by Gauss [41] and independently by Legendre. We present the least square estimator developed by Gauss.

4.1 Least Squares Estimator

Let $\mathbf{Z}^T = (z_1, z_2, \ldots, z_m)$ be the measurements made to determine the parameter x. Let us assume that the measurement z_i of parameter x differs by an amount e_i, then

$$z_i = x + e_i, \quad i = 1, 2, \ldots, m. \tag{4.5}$$

The least squares estimate $\hat{x}_{LS}$ is the estimate of x that minimizes

$$\mathcal{E}_{LS} \triangleq \frac{1}{2}\sum_{i=1}^{m} e_i^2 = \frac{1}{2}\mathbf{e}^T\mathbf{e} \tag{4.6}$$

Generation of the least squares estimator problem for multiple parameters is $\mathbf{X}^T = (x_1, x_2, \ldots, x_n)$ where $\mathbf{Z}$ and $\mathbf{X}$ are related by

$$\mathbf{Z} = H\mathbf{X} + \mathbf{e} \tag{4.7}$$

where H is an $(m \times n)$ matrix with a rank of n and $\mathbf{e}^T = (e_1, e_2, \ldots, e_m)$. In general, $m \geq n$, that is, measurements are more than the number of parameters to be estimated. It is difficult to estimate $\mathbf{X}$ uniquely; however, it is possible to estimate $\mathbf{X}$ in some

sense that minimizes the effect of the errors $\mathbf{e}$. In the least squares estimator, $\hat{\mathbf{X}}_{LS}$ is determined to minimize

$$\begin{aligned}\mathcal{E}_{LS} &= \tfrac{1}{2}\sum_{i=1}^{m} e_i^2 = \mathbf{e}^T\mathbf{e} \\ &= \tfrac{1}{2}\left(\mathbf{Z} - H\mathbf{X}\right)^T\left(\mathbf{Z} - H\mathbf{X}\right)\end{aligned} \tag{4.8}$$

Minimization with respect to $\mathbf{X}$ can be obtained by equating the derivative of $\mathcal{E}_{LS}$ with respect to $\mathbf{X}$ to zero

$$\frac{\partial \mathcal{E}_{LS}}{\partial X} = -\left(\mathbf{Z} - H\mathbf{X}\right)^T H = 0$$

or

$$H^T\left(\mathbf{Z} - H\mathbf{X}\right) = 0 \tag{4.9}$$

Equation (4.9) states that the residual $\mathbf{r} = \left(\mathbf{Z} - H\mathbf{X}\right)^T$is orthonormal to the observation matrix H. Rewriting (4.9) gives

$$H^T H\mathbf{X} = H^T\mathbf{Z} \tag{4.10}$$

Equation (4.10) is called the *normal equation*. The rank of matrix H is n, hence, the inverse of $H^T H$ exists and the least square estimator is found to be

$$\hat{\mathbf{X}}_{LS} = \left(H^T H\right)^{-1} H^T\mathbf{Z} \tag{4.11}$$

The error in the estimator is defined as

$$\tilde{\mathbf{X}}_{LS} \triangleq \mathbf{X} - \hat{\mathbf{X}}_{LS} = \mathbf{X} - \left(H^T H\right)^{-1} H^T\left[H\mathbf{X} + \mathbf{e}\right]$$

Or

$$\tilde{\mathbf{X}}_{LS} = -\left(H^T H\right)^{-1} H^T\mathbf{e} \tag{4.12}$$

If the average of errors $E(\mathbf{e}) = 0$, then the error in the estimate

$$\begin{aligned}E\left(\mathbf{X} - \hat{\mathbf{X}}_{LS}\right) &= -E\left[\left(H^T H\right)^{-1} H^T\mathbf{e}\right] \\ &= -\left(H^T H\right)^{-1} H^T E(\mathbf{e}) = 0\end{aligned}$$

or

$$E\left(\tilde{\mathbf{X}}_{LS}\right) = 0. \tag{4.13}$$

The estimator that satisfies (4.13) is said to be unbiased. The covariance of the error in the estimator is given by

$$E\left(\tilde{\mathbf{X}}_{LS}\tilde{\mathbf{X}}_{LS}^{T}\right) = (H^T H)^{-1} H^T R\, H (H^T H)^{-1} \tag{4.14}$$

If the errors $\mathbf{e}$ are uncorrelated and have identical variance σ^2, that is,

$$R = \sigma^2 \mathbf{I}$$

where $\mathbf{I}$ is the identity matrix, then (4.14)

$$P = \sigma^2 (H^T H)^{-1}. \tag{4.15}$$

4.2 Generalized Least Squares Estimator

This is the weighted least squares estimator, which achieved by introducing a weighting function W in (4.8)

$$\mathcal{E}_{WLS} = (\mathbf{Z} - H\mathbf{X})^T\, W\, (\mathbf{Z} - H\mathbf{X}) \tag{4.16}$$

Weighting function is used to emphasize or de-emphasize some of the measurements. The optimal choice for the weighting matrix W is the covariance matrix [87] of the measurement errors (noise) $W = C_{\mathbf{e}}^{-1}$. Then the generalized least squares estimator

$$\begin{aligned} \hat{\mathbf{X}}_{WLS} &= \left(H^T W H\right)^{-1} H^T W \mathbf{Z} \\ \hat{\mathbf{X}}_{WLS} &= \left(H^T C_{\mathbf{e}}^{-1} H\right)^{-1} H^T C_{\mathbf{e}}^{-1} \mathbf{Z} \end{aligned} \tag{4.17}$$

also minimizes the mean-squared estimator error

$$\mathcal{E}_{MSE} = E\left[\left(\mathbf{X} - \hat{\mathbf{X}}_{LS}\right)^T \left(\mathbf{X} - \hat{\mathbf{X}}_{LS}\right)\right]$$

The estimator (4.17) is referred to as the *best linear unbiased estimator* (BLUE) or Gauss-Markov estimator.

4.3 Maximum Likelihood Estimator

Given a set of observations $\mathbf{Z}$ and probability density function $p(\mathbf{Z}|\theta)$, the maximum likelihood estimator is the one that maximizes the probability distribution that makes the observed data most likely. For example, Fig. 4.2 shows the values of observed output versus the fraction of time each observation occurred. Based on the shape of the data distribution, it is assumed that it is a normal distribution function with mean μ and variance σ^2, which are the parameters $\theta = \{\mu, \sigma^2\}$ that need to be estimated. Figure 4.3 shows the normal density functions for two different sets of parameters. So by changing values of the parameters, one can get different density functions. The estimation of these parameters that matches the distribution of the data observed is the aim of the maximum likelihood estimator. Assuming the measurements are independent, the likelihood function is given by

$$L\left(z_1, z_2, \ldots, z_m|\theta_1, \theta_2, \ldots, \theta_k\right) = \prod_{i=1}^{m} p\left(z_i|\theta_1, \theta_2, \ldots, \theta_k\right) \tag{4.18}$$

The logarithmic likelihood function is given by

$$\Lambda = \ln L = \sum_{i=1}^{m} \ln\, p\left(z_i|\theta_1, \theta_2, \ldots, \theta_k\right) \tag{4.19}$$

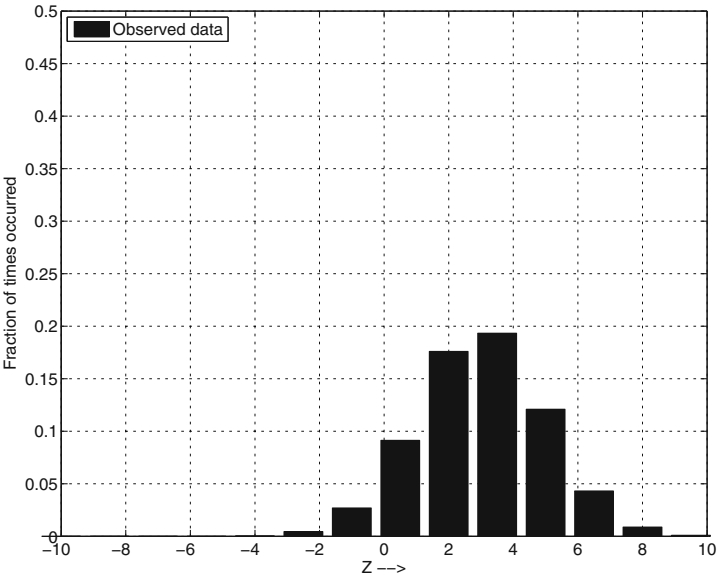


Fig. 4.2 Observed data distribution

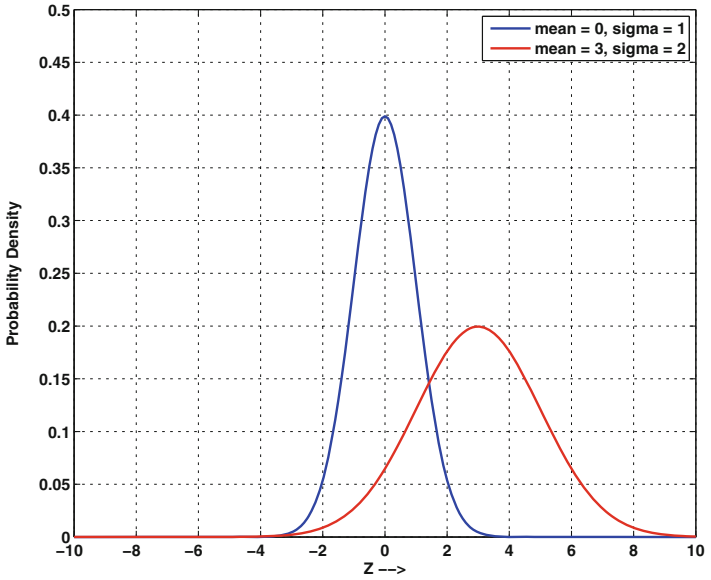


Fig. 4.3 Normal density functions

Assuming the probability density function $p(\mathbf{Z}|\theta)$ is continuous and differentiable, a necessary condition that $\hat{\theta}_{ML}$ maximizes the likelihood function $p(\mathbf{Z}|\theta)$ is

$$\frac{\partial}{\partial\theta}\, p\left(\mathbf{Z}|\hat{\theta}_{ML}\right) = 0 \tag{4.20}$$

The maximum likelihood estimations of $\theta_1, \theta_2, \ldots, \theta_k$ are obtained by maximizing Λ or L. This is achieved by (4.20), or by solving the simultaneous equations given by

$$\frac{\partial \Lambda}{\partial \theta_j} = 0, \qquad j = 1, 2, \ldots, k \tag{4.21}$$

The following example illustrates the maximum likelihood method using a normal distribution.

Example: Let $z_1, z_2, \ldots, z_m$ be the independent samples from a Gaussian (Normal) distribution $\mathcal{N}\left(\mu, \sigma^2\right)$ where μ and σ^2 are unknown. We use the maximum likelihood estimation method to determine their values.
The likelihood function (4.18) is given by

$$\begin{aligned} L\left(z_1, z_2, \ldots, z_m|\mu, \sigma^2\right) &= \prod_{i=1}^{m} \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left\{-\frac{1}{2\sigma^2}\left(z_i - \mu\right)^2\right\} \\ &= \frac{1}{\left(2\pi\sigma^2\right)^{m/2}} \exp\left\{-\frac{1}{2\sigma^2}\sum_{i=1}^{m}\left(z_i - \mu\right)^2\right\} \end{aligned} \tag{4.22}$$

The logarithmic likelihood function is

$$\Lambda = \ln L = -\frac{m}{2}\ln(2\pi) - m\ln(\sigma) - \frac{1}{2\sigma^2}\sum_{i=1}^{m}(z_i - \mu)^2 \tag{4.23}$$

Estimation of μ_{ML}:

$$\frac{\partial \Lambda}{\partial \mu} = \frac{1}{\sigma^2}\sum_{i=1}^{m}(z_i - \mu) = 0$$

or

$$\mu_{ML} = \frac{1}{m}\sum_{i=1}^{m} z_i$$

Estimation of σ_{ML}:

$$\frac{\partial \Lambda}{\partial \sigma} = -\frac{m}{\sigma} + \frac{1}{\sigma^3}\sum_{i=1}^{m}(z_i - \mu)^2 = 0$$

Or

$$\sigma^2_{ML} = \frac{1}{m}\sum_{i=1}^{m}(z_i - \mu)^2$$

The maximum likelihood estimator has several desirable characteristics needed in an estimator. For example,

- If there exists an estimator that achieves a Cramer-Rao lower bound as an equality, it can be determined using the maximum likelihood method.
- The maximum likelihood estimator is consistent.
- The maximum likelihood estimator is asymptotically efficient, that is, the maximum likelihood estimator achieves an asymptotically Cramer-Rao lower bound for the estimation error.

A Cramer-Rao bound is a bound that is often used as a measure of estimator error and is given in the following theorem:

Theorem: If $\hat{\theta}$ is any unbiased estimator of θ based on the measurement data $\mathbf{Z}$, then the covariance matrix of error in the estimator is bounded below by the inverse of the *Fisher information matrix* $\mathbf{J}$

$$E\left\{\left(\theta - \hat{\theta}\right)\left(\theta - \hat{\theta}\right)^T \mid \theta\right\} \geq \mathbf{J}^{-1} \tag{4.24}$$

where

$$\mathbf{J} = E\left\{\left[\frac{\partial}{\partial\theta}\ln p\left(\mathbf{Z}|\theta\right)\right]\left[\frac{\partial}{\partial\theta}\ln p\left(\mathbf{Z}|\theta\right)\right]^{T} \mid \theta\right\} \tag{4.25}$$

Further details on the characteristics of the maximum likelihood estimator can be found in [51, 68, 87].
Although for exponential distributions it is possible to estimate the parameters, in general the maximum likelihood estimation is difficult to compute. One of the algorithm that computes maximum likelihood called the "expectation-maximization" algorithm [67], which is presented in Sect. 4.5.

4.4 Maximum a Posteriori Estimators

From the Baye's rule (4.3), the a posteriori density $p\left(\theta|\mathbf{Z}\right)$ and the a priori density $p(\theta)$ are related

$$p\left(\theta|\mathbf{Z}\right) = \frac{p\left(\mathbf{Z}|\theta\right)p\left(\theta\right)}{p\left(\mathbf{Z}\right)}. \tag{4.26}$$

The denominator $p(\mathbf{Z})$ in (4.26) does not depend on θ and hence the maximum a posteriori estimator involves maximizing the numerator. As in the case of the maximum likelihood estimator, the maximization of the numerator involves solving

$$\frac{\partial}{\partial\theta}\ln p(\theta,\mathbf{Z}) = \frac{\partial}{\partial\theta}\ln p(\mathbf{Z}|\theta) + \frac{\partial}{\partial\theta}\ln p(\theta) = 0 \tag{4.27}$$

This is similar to the maximum likelihood estimator but includes an additional term $\frac{\partial}{\partial\theta}\ln p(\theta)$, which takes into account the prior information of the parameter θ.
The following example illustrates the maximum a posteriori estimation technique.

Example: Let $z_1, z_2, \ldots, z_m$ be independent samples of a Gaussian random variable $N\left(\mu, \sigma_z^2\right)$ where μ is a random variable with a priori distribution $N\left(0, \sigma_\mu^2\right)$. It is assumed that the variances σ_z^2 and σ_μ^2 are known.

$$\begin{aligned} p(\mu,\mathbf{Z}) &= p(\mathbf{Z}|\mu)\,p(\mu) \\ &= \left[\frac{1}{\left(2\pi\sigma_z^2\right)^{m/2}}\exp\left\{-\frac{1}{2\sigma_z^2}\sum_{i=1}^{m}\left(z_i-\mu\right)^2\right\}\right]\left[\frac{1}{\left(2\pi\sigma_\mu^2\right)^{1/2}}\exp\left\{-\frac{\mu^2}{2\sigma_\mu^2}\right\}\right] \end{aligned}$$

The likelihood equation for the maximum a posteriori estimation $\hat{\mu}_{MAP}$ is

$$\frac{\partial}{\partial\mu}\ln p(\mathbf{Z}|\mu) + \frac{\partial}{\partial\mu}\ln p(\mu) = \frac{1}{\sigma_z^2}\sum_{i=1}^{m}\left(z_i-\mu_{MAP}\right) - \frac{\mu_{MAP}}{\sigma_\mu^2} = 0$$

Solving for μ_{MAP} we get

$$\mu_{MAP} = \frac{\sigma_\mu^2}{\sigma_z^2 + m\,\sigma_\mu^2} \sum_{i=1}^{m} z_i$$

The case when no a priori information about μ is available, μ is obtained by letting σ_μ^2 become infinite, then

$$\lim_{\sigma_\mu^2 \to \infty} \frac{\sigma_\mu^2}{\sigma_z^2 + m\,\sigma_\mu^2} = \frac{1}{\frac{\sigma_z^2}{\sigma_\mu^2} + m} = \frac{1}{m}$$

Then

$$\mu_{MAP} = \frac{1}{m} \sum_{i=1}^{m} z_i,$$

which is the sample mean.

4.5 Expectation Maximization Algorithm

The expectation maximization is an iterative method used to estimate the parameters θ of a probability distribution. Often it is the maximum likelihood estimation of θ. As an example, consider an experiment where two coins (A, B) are tossed one at a time. The probability of "heads" for coin 'A' is θ_A and similarly θ_B for coin 'B' and $\theta_A \neq \theta_B$. Clearly, the coins are biased. We'd like to estimate the parameters $\theta = (\theta_A, \theta_B)$ from the observations. The obvious choice is to take coin 'A' and flip it several times and estimate the θ_A as

$$\theta_A = \frac{\#\text{ of heads using coin A}}{\text{total }\#\text{ of flips using coin A}} \tag{4.28}$$

Similarly, θ_B can be estimated. We can denote the outcome of each coin toss by $x = (x_1, x_2, \ldots, x_n)$ and the identity of the coin tossed by another variable $z = (z_1, z_2, \ldots, z_m)$, where $x_i \in \{1, 2, \ldots, n\}$ denotes the number of heads for ith trail and $z_i \in \{A, B\}$ the identity of the coin used for the ith trail. Then $P(x, z; \theta)$ is the joint probability of x and z and Eq. (4.28) solves parameters $\theta = (\theta_A, \theta_B)$, which maximizes the log-likelihood $\log P(x, z; \theta)$.

Now, suppose only partial data are given, that is, the head counts x but not the identity of the coin used for each trail, so it is not possible to estimate the parameters $\theta = (\theta_A, \theta_B)$. This is the case of incomplete data. The coin identity variables z are referred to as hidden variables or latent factors. If somehow, we guess the coin used correctly for each trail, then the parameter estimation for the incomplete data can be

reduced to the maximum likelihood estimation for the complete data. The expectation maximization (EM) algorithm makes a guess on the identity of the coin and estimates the parameters. It improves the estimation of the parameters by maximizing the log probability $\log P(x;\theta)$ of the observed data on each iteration [30, 32, 67]. We now present the EM algorithm.

Let Y be the observation space and $\mathbf{y} \in R^m$ is an observation from Y. Similarly, let the actual space be denoted by $\mathcal{X}$ with $\mathbf{x} \in R^n,\ m < n$. $\mathbf{x}$ is referred to as the complete data and is not observed directly but only through $\mathbf{y}$, where $\mathbf{y} = T(\mathbf{x})$. Hence, $T(\mathbf{x})$ is a many to one mapping and an observation $\mathbf{y}$ determines a subset of $\mathcal{X}$, which is denoted by $\mathcal{X}(\mathbf{y})$. The *pdf* of the complete data is $p_{\mathbf{x}}(\mathbf{x}|\theta) = p(\mathbf{x}|\theta)$, where $\theta \in \Theta \subset R^r$ is a set of parameters of the density. The *pdf* is assumed to be differentiable with respect to θ. The *pdf* of incomplete data is

$$p(\mathbf{y}|\theta) = \int_{\mathcal{X}(\mathbf{y})} p(\mathbf{x}|\theta)\, d\mathbf{x} \tag{4.29}$$

Let

$$l_y = g(\mathbf{y}|\theta) \tag{4.30}$$

denote the likelihood function and the log-likelihood function is

$$L_y = \log\ g(\mathbf{y}|\theta) \tag{4.31}$$

The EM algorithm tries to find the parameters θ that maximizes the log-likelihood $\log p(\mathbf{x}|\theta)$, but we do not have the data $\mathbf{x}$ to compute the log-likelihood. So, this is achieved indirectly by maximizing the expectation of $\log p(\mathbf{x}|\theta)$ given the data $\mathbf{y}$ and the current estimate of θ. This can be expressed in two steps, (1) the expectation step and (2) the maximization step. Let $\theta^{(k)}$ be the estimate of the parameters at the kth iteration.

For the E-step, compute

$$Q\left(\theta|\theta^{(k)}\right) = E\left[\log p(\mathbf{x}|\theta)\,|\mathbf{y}, \theta^{(k)}\right] \tag{4.32}$$

Note that while the first argument $\mathbf{y}$ conditions the likelihood of the complete data, the second argument $\theta^{(k)}$ conditions the expectation and it is fixed and known at every E-step.

For the M-step, let $\theta^{(k+1)}$ be the value of θ that maximizes $Q\left(\theta|\theta^{(k)}\right)$

$$\theta^{(k+1)} = \arg\max_{\theta}\ Q\left(\theta|\theta^{(k)}\right) \tag{4.33}$$

Note in Eq. (4.33) the maximization is with respect to θ of the Q function, the conditioner of the complete data likelihood.

In order to implement the EM algorithm, first choose an initial value of $\theta^{(k)}$, k=1, then perform E and M steps until the convergence, that is, the difference between two successive log-likelihood values is less than some pre-determined value. Note that the EM algorithm converges to the local maxima not the global maxima. Hence, one may have to run the EM algorithm with several initial values of $\theta^{(k)}$, $k = 1$, and select the one that results in the maximum log-likelihood value.

From Eq. (4.32),

$$Q\left(\theta|\theta^{(k)}\right) = E\left[\log p\left(\mathbf{x}|\theta\right)|\mathbf{y}, \theta^{(k)}\right] = \int_{\mathcal{X}(y)} \log p\left(\mathbf{x}|\theta\right) p\left(\mathbf{x}|y, \theta^{(k)}\right) \tag{4.34}$$

where the integration is over the support of $\mathcal{X}$ given $\mathbf{y}$. Note that from Baye's rule we have

$$p\left(x|y, \theta\right) = \frac{p\left(x, y|\theta\right)}{p\left(y|\theta\right)}. \tag{4.35}$$

Since $y = T\left(x\right)$ is a deterministic function, the above Eq. (4.35) can be written as

$$p\left(x|y, \theta\right) = \frac{p\left(x|\theta\right)}{p\left(y|\theta\right)}. \tag{4.36}$$

Hence, Eq. (4.34) becomes

$$Q\left(\theta|\theta^{(k)}\right) = \int_{\mathcal{X}(y)} \log p\left(\mathbf{x}|\theta\right) \frac{p\left(\mathbf{x}|\theta^{(k)}\right)}{p\left(\mathbf{y}|\theta^{(k)}\right)} \tag{4.37}$$

We now prove the convergence of EM algorithm; first consider the following theorems:

Definition: Let f be a real valued function defined on an interval $I = [a, b]$. f is said to be convex on I if for all $x_1, x_2 \in I$, $\lambda \in [0, 1]$,

$$f(\lambda x_1 + (1 - \lambda)x_2) \leq \lambda f(x_1) + (1 - \lambda) f(x_2).$$

f is said to be strictly convex if the inequality is strict.

Theorem 1 (Jensen's inequality): Let f be a convex function defined on an interval I. If $x_1, x_2, \ldots, x_n \in I$ and $\lambda_1, \lambda_2, \ldots, \lambda_n \geq 0$ with $\sum_{i=1}^{n} \lambda_i = 1$,

$$f\left(\sum_{i=1}^{n} \lambda_i x_i\right) \leq \sum_{i=1}^{n} \lambda_i f\left(x_i\right). \tag{4.38}$$

Proof: The proof is a straightforward application of the definition given above.

Theorem 2: Let $L(\theta) = \log p\left(y|\theta\right)$ be the log-likelihood function. For $\theta \in \Theta$, if $Q\left(\theta|\theta^{(k)}\right) \geq Q\left(\theta^{(k)}|\theta^{(k)}\right)$, then $L(\theta) \geq L\left(\theta^{(k)}\right)$.

Proof: By definition (we use the proof given in [16])

$$L(\theta) = \log p\,(y|\theta) \tag{4.39}$$

Equation (4.39) can be written as

$$L(\theta) = \log \int_{\mathcal{X}(y)} p\,(x, y|\theta) = \log \int_{\mathcal{X}(y)} p\,(x|\theta) \tag{4.40}$$

where we used the fact that $y = T(x)$ and is deterministic. Multiplying the numerator and denominator of Eq. (4.40) by $p\left(x|y, \theta^{(k)}\right)$, we get

$$L(\theta) = \log \int_{\mathcal{X}(y)} \frac{p\,(x|\theta)}{p\left(x|y, \theta^{(k)}\right)} p\left(x|y, \theta^{(k)}\right) dx \tag{4.41}$$

Writing as an expectation and using Jensen's inequality would give us

$$L(\theta) = \log E_{X|y,\theta^{(k)}}\left[\frac{p\,(X|\theta)}{p\left(X|y, \theta^{(k)}\right)}\right] \geq E_{X|y,\theta^{(k)}}\left[\log \frac{p\,(X|\theta)}{p\left(X|y, \theta^{(k)}\right)}\right] \tag{4.42}$$

We can express the right-hand side as

$$\begin{aligned} &E_{X|y,\theta^{(k)}}\left[\log \tfrac{p(X|\theta)}{p\left(X|y,\theta^{(k)}\right)}\right] = \\ &= E_{X|y,\theta^{(k)}}\left[\log p\,(X|\theta)\right] + E_{X|y,\theta^{(k)}}\left[-\log p\left(X|y, \theta^{(k)}\right)\right] \\ &= Q\left(\theta|\theta^{(k)}\right) + E_{X|y,\theta^{(k)}}\left[-\log p\left(X|y, \theta^{(k)}\right)\right] \end{aligned} \tag{4.43}$$

where we used the definition of Q-function given in Eq. (4.32). Substituting Eqs. (4.43) in (4.42), we get

$$L\,(\theta) \geq Q\left(\theta|\theta^{(k)}\right) + E_{X|y,\theta^{(k)}}\left[-\log p\left(X|y, \theta^{(k)}\right)\right] \tag{4.44}$$

or

$$L\,(\theta) \geq Q\left(\theta|\theta^{(k)}\right) + h\left(X|y, \theta^{(k)}\right) \tag{4.45}$$

where

$$h\left(X|y, \theta^{(k)}\right) = E_{X|y,\theta^{(k)}}\left[-\log p\left(X|y, \theta^{(k)}\right)\right]. \tag{4.46}$$

Equation (4.45) is the lower bound on $L(\theta)$. We note that the only term in Eq. (4.45) that is dependent on θ is $Q\left(\theta|\theta^{(k)}\right)$. Now, consider right-hand side of Eq. (4.45) for the case $\theta = \theta^{(k)}$

$$\begin{aligned} & Q\left(\theta^{(k)}|\theta^{(k)}\right) + h\left(X|y, \theta^{(k)}\right) \\ &= E_{X|y,\theta^{(k)}}\left[\log p\left(X|\theta^{(k)}\right)\right] + E_{X|y,\theta^{(k)}}\left[-\log p\left(X|y, \theta^{(k)}\right)\right] \\ &= \int_{\mathcal{X}(y)} p\left(x|y, \theta^{(k)}\right) \log p\left(x|\theta^{(k)}\right) dx - \int_{\mathcal{X}(y)} p\left(x|y, \theta^{(k)}\right) \log p\left(x|y, \theta^{(k)}\right) dx \\ &= \int_{\mathcal{X}(y)} p\left(x|y, \theta^{(k)}\right) \log \frac{p\left(x|\theta^{(k)}\right)}{p\left(x|y,\theta^{(k)}\right)} dx \end{aligned}$$

Using the fact that

$$p\left(x|\theta^{(k)}\right) = p\left(x, y|\theta^{(k)}\right)$$

and using Baye's rule we get

$$\begin{aligned} & Q\left(\theta^{(k)}|\theta^{(k)}\right) + h\left(X|y, \theta^{(k)}\right) \\ &= \int_{\mathcal{X}(y)} p\left(x|y, \theta^{(k)}\right) \log p\left(y|\theta^{(k)}\right) dx \\ &= \log p\left(y|\theta^{(k)}\right) \\ &\triangleq L\left(\theta^{(k)}\right) \end{aligned} \tag{4.47}$$

The theorem assumes $Q\left(\theta|\theta^{(k)}\right) \geq Q\left(\theta^{(k)}|\theta^{(k)}\right)$ and using Eqs. (4.45) and (4.47) we have

$$\begin{aligned} L(\theta) &\geq Q\left(\theta|\theta^{(k)}\right) + h\left(X|y, \theta^{(k)}\right) \\ &\geq Q\left(\theta^{(k)}|\theta^{(k)}\right) + h\left(X|y, \theta^{(k)}\right) \\ &= L\left(\theta^{(k)}\right) \end{aligned} \tag{4.48}$$

Hence

$$L(\theta) \geq L\left(\theta^{(k)}\right).$$

■

The EM algorithm maximizes the Q-function in the M-step by finding $\theta^{(k=1)}$ such that

$$\theta^{(k+1)} = \arg\max_{\theta} Q\left(\theta|\theta^{(k)}\right)$$

hence

$$Q\left(\theta|\theta^{(k)}\right) \geq Q\left(\theta^{(k)}|\theta^{(k)}\right)$$

which, in turn, implies that

$$L(\theta)^{(k+1)} \geq L\left(\theta^{(k)}\right). \tag{4.49}$$

Hence, the EM guarantees convergence. We now present an example.

Example: In the beginning of this section, we posed a problem where two biased coins are considered for an experiment. One coin is randomly picked from a pool of two coins and tossed 10 times and the head count is reported. The experiment is repeated 5 times, each time with a coin randomly picked. The identity of the coin picked for each experiment is not known. The problem is to estimate the parameters $\theta = (\theta_A, \theta_B)$.

If we know the identities of the coin used for each experiment, then we can estimate the parameters using the Eq. (4.28) using the head counts for each experiment, as shown in Fig. 4.4 resulting in the maximum likelihood estimation of the parameters $\theta = (\theta_A, \theta_B)$.

$$\hat{\theta}_A = \frac{24}{24+6} = 0.8, \qquad \hat{\theta}_B = \frac{9}{9+11} = 0.45$$

If we do not know the identities of the coins used in experiments, the estimation of the parameters using Eq. (4.28) is not possible. This is where the EM algorithm helps. The steps involved in estimating the parameters are shown in Fig. 4.5.
We now derive the equations needed: Let 'x' denote the count of "heads" when a coin is tossed 'n' times and 'z' denote the identity of the coin used. It is well known that the coin tossing follows a Bernoulli distribution:

$$p\left(x|z=A\right) = \binom{n}{x} \theta_A^x \left(1-\theta_A\right)^{(n-x)}. \tag{4.50}$$

The joint distribution is

$$p\left(x|z=(A,B)\right) = p(z=A)\,p\left(x|z=A\right) + p(z=B)\,p\left(x|z=B\right) \tag{4.51}$$

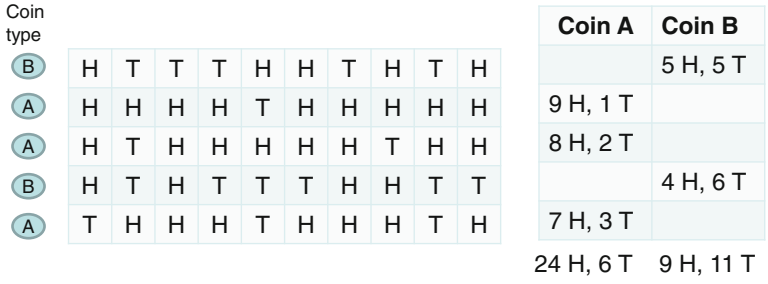


Fig. 4.4 Maximum likelihood estimation of θ

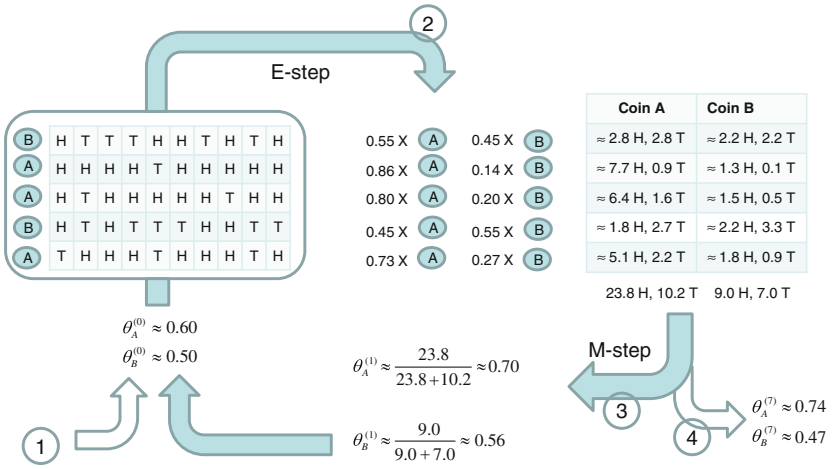


Fig. 4.5 Parameter estimation using the EM algorithm

The log-likelihood for all M trials can be written as

$$\log p\left(x|z=(A,B)\right)=\sum_{i=1}^{M}\log\left[p\left(z_i=A\right)\binom{n}{x_i}\theta_A^{x_i}\left(1-\theta_A\right)^{n-x_i}+p\left(z_i=B\right)\binom{n}{x_i}\theta_B^{x_i}\left(1-\theta_B\right)^{n-x_i}\right] \tag{4.52}$$

Now for the EM algorithm, we use the Q-function defined as

$$Q\left(\theta,\theta^{(k)}\right)=E\left[\log p\left(\mathbf{x},\mathbf{z}|\theta\right)|\mathbf{x},\theta^{(k)}\right] \tag{4.53}$$

or

$$\begin{aligned}Q\left(\theta,\theta^{(k)}\right)&=\sum_{i=1}^{M}\sum_{z_i=\{A,B\}}p\left(z_i|x_i,\theta^{(k)}\right)\log p\left(z_i|\theta\right)p\left(x_i|z_i,\theta\right)\\&=\sum_{i=1}^{M}\left(p\left(z_i=A|x_i,\theta^{(k)}\right)\left[\log p\left(A|\theta\right)+\log\binom{n}{x_i}\theta_A^{x_i}\left(1-\theta_A\right)^{n-x_i}\right]\right.\\&\quad\left.+\,p\left(z_i=B|x_i,\theta^{(k)}\right)\left[\log\left(B|\theta\right)+\log\binom{n}{x_i}\theta_B^{x_i}\left(1-\theta_B\right)^{n-x_i}\right]\right)\end{aligned} \tag{4.54}$$

The E-step:

For the E-step, we estimate the posterior probability that the sample x_i is from $z=A$, that is,

$$p\left(z_i=A|x_i,\theta^{(k)}\right)=\frac{p\left(x_i,z_i=A|\theta^{(k)}\right)}{p\left(x_i|\theta^{(k)}\right)}$$

$$= \frac{p\left(z_i = A|\theta^{(k)}\right)\binom{n}{x_i}\theta_A^{x_i}(1-\theta_A)^{n-x_i}}{p\left(z_i = A|\theta^{(k)}\right)\binom{n}{x_i}\theta_A^{x_i}(1-\theta_A)^{n-x_i} + p\left(z_i = B|\theta^{(k)}\right)\binom{n}{x_i}\theta_B^{x_i}(1-\theta_B)^{n-x_i}}$$

$$\triangleq p_{i,A}^{(k+1)} \tag{4.55}$$

Similarly, we estimate $p\left(z_i = B|x_i, \theta^{(k)}\right) \triangleq p_{i,B}^{(k+1)}$.

The M-step:

For the M-step, we need to maximize the Q-function. There are four variables, θ_A, θ_B, $p\left(z_i = A|\theta_A\right) \triangleq \alpha_i$ and $p\left(z_i = B|\theta_B\right)$. For simplicity, let us assume that $p\left(z_i = B|\theta_B\right) = 1 - \alpha_i$. Then

$$\begin{aligned} Q\left(\theta, \theta^{(k)}\right) = \sum_{i=1}^{M}\Bigg[& p_{i,A}^{(k+1)} \log\alpha_i \binom{n}{x_i}\theta_A^{x_i}(1-\theta_A)^{n-x_i} \\ & + p_{i,B}^{(k+1)} \log(1-\alpha_i)\binom{n}{x_i}\theta_B^{x_i}(1-\theta_B)^{n-x_i}\Bigg] \end{aligned} \tag{4.56}$$

Now for maximization we find the derivatives of Q-function with respect to each variable and set it to zero and find the updated variable value. We get

$$\frac{dQ}{d\theta_A} = \sum_{i=1}^{M} p_{i,A}^{(k+1)}\left[\frac{x_i}{\theta_A} - \frac{(n-x_i)}{(1-\theta_A)}\right] = 0$$

After some manipulation, we get

$$\theta_A^{(k+1)} = \frac{1}{n}\ \frac{\sum_{i=1}^{M} p_{i,A}^{(k+1)} x_i}{\sum_{i=1}^{M} p_{i,A}^{(k+1)}} \tag{4.57}$$

Similarly,

$$\theta_B^{(k+1)} = \frac{1}{n}\ \frac{\sum_{i=1}^{M} p_{i,B}^{(k+1)} x_i}{\sum_{i=1}^{M} p_{i,B}^{(k+1)}} \tag{4.58}$$

Now taking the derivative with respect to α_i we get

$$\frac{dQ}{d\alpha_i} = \frac{1}{\alpha_i}\sum_{i=1}^{M} p_{i,A}^{(k+1)} - \frac{1}{1-\alpha_i}\sum_{i=1}^{M} p_{i,B}^{(k+1)} = 0$$

solving for α_i we get

$$\alpha_i^{(k+1)} = \frac{1}{M} \sum_{i=1}^{M} p_{i,A}^{(k+1)} \tag{4.59}$$

We used Eqs. (4.57–4.59) to estimate the parameters $\theta = (\theta_A, \theta_B) = (0.74, 0.47)$, which are closer to the maximum likelihood values. The EM algorithm is a powerful tool used in many applications.

In this chapter, we only provided few of the estimation techniques that are widely used. However, estimation theory is a subject by itself and there are several good books on it. Readers interested in the subject are encouraged to read [58, 87].

Chapter 5
Atmospheric Acoustics

The science of sound propagation in the atmosphere is called atmospheric acoustics. Sound propagates through a media and undergoes transformation naturally. There are several things that affect the propagation of sound. For example, temperature and wind affect sound propagation. Wind turbulence also affects propagation. Just as light reflects, sound also gets reflected by several barriers such as ground, water, etc. Furthermore, the propagation of sound depends on the type of ground. Hard surfaces, such a concrete floor, reflect all the sound, whereas soft or absorbing ground reflects only part of the sound impinging on the surface. Variations in terrain surface can also result in propagation losses. Sound can also be refracted by the medium in which it is traveling. For example, the propagation velocity of sound changes with temperature which in turn causes refraction of sound waves. The temperature decreases with height above the ground, so as a result, sound undergoes refraction and either refracts upward or downward depending on whether it is day or night. This refraction of the sound waves has a dramatic effect on their propagation and results in zones where the sound becomes inaudible. A study of military history by Ross [78] revealed several instances where the military generals were unable to hear the sounds of guns due to atmospheric refraction, and as a result, failed to position their troops to challenge the advancing enemy and thus lost the battle.

In this chapter, we present some of the techniques to compute the effects of ground and atmosphere on sound propagation. This study helps us understand prevailing terrain conditions and how to deploy acoustic sensors for better coverage in the field for the optimal reception of sound waves. It also helps us to compensate the results for atmospheric effects. Some of the topics covered include,

- Sound waves,
- Atmospheric absorption,
- Reflection of sound waves,
- Acoustic impedance,
- Atmospheric Refraction of sound waves, and
- Compensation for propagation delay.

T. Damarla, *Battlefield Acoustics*, DOI 10.1007/978-3-319-16036-8_5

Atmospheric acoustics is a vast field of study. This chapter is only intended to provide some rudimentary concepts. Interested readers should consult [5, 8, 77, 81].

Sound waves are two types: (a) plane waves and (b) spherical waves. Plane waves travel in one direction in a plane. Sound pressure is a function of position **r** and time t, thus, a plane wave traveling in x-direction is given by

$$p(\mathbf{r}, t) = A\cos(kx - wt) \tag{5.1}$$

where A is the amplitude and $(kx - wt)$ is the phase, which depends on the angular frequency w and the wavenumber k. The wavenumber is given by

$$k = \frac{w}{c} \tag{5.2}$$

where c is the speed of sound. Plane waves seldom occur in the atmosphere, but are useful for understanding wave propagation. The sound's speed increases at ~0.6 m/s for every 1 °C raise in temperature. A spatial variation of sound speed occurs due to atmospheric refraction which bends the sound wave toward the region of lower speed. Figure 5.1 illustrates how the sound wave is refracted with changing temperature.

Spherical Waves

In a majority of cases, we assume sound sources are point or monopole sources. Monopole sources generate spherical waves as shown in Fig. 5.2. The spherical

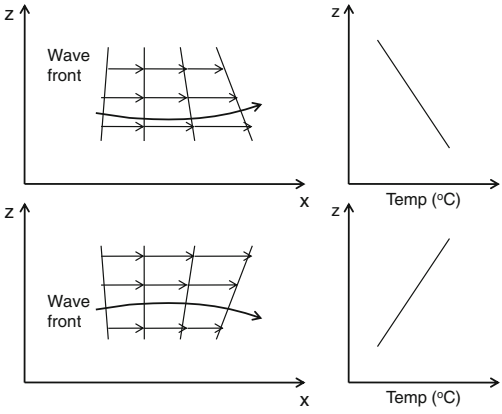


Fig. 5.1 Refraction of sound with temperature

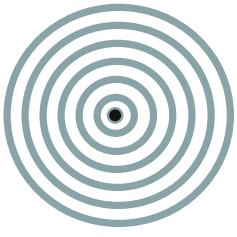


Fig. 5.2 Spherical waves radiating from the monopole at the center

waves spread outward from the center of the source in an unbounded homogeneous atmosphere. Sound pressure in a spherical wave is constant within each spherical surface and is given by

$$p_c(\mathbf{r}) = S\,\frac{\exp(ikr)}{r} \tag{5.3}$$

where r is the radial distance from the source and S is a constant and depends on the generating source and $i = \sqrt{-1}$.

Atmospheric absorption

Sound propagating through the atmosphere experiences dispersion due to atmospheric absorption, that is, waves with different frequencies travel at different speeds. This is accounted for in the wave pressure equation by adding an additional term

$$p_c(\mathbf{r}) = S\,\frac{\exp(ikr)}{r}\,\exp(-k_i r) \tag{5.4}$$

The term $\exp(-k_i r)$ is a factor that indicates that the amplitude decreases exponentially with distance. The phase effects of atmospheric absorptions can be neglected in the majority of cases.

Sound pressure level

The sound pressure of a sound source is measured as the average of the squared pressure p_{av}^2. It is a measure of loudness. The sound pressure level is defined as

$$L_p = 20\log\left(\frac{p_{\text{av}}}{p_{\text{ref}}}\right) \tag{5.5}$$

and the $p_{\text{ref}} = 20\,\mu\text{Pa}$, where Pa denotes Pascal and 1 Pa $= 1\,\text{N/m}^2$. We present spectrograms of three typical vehicles, namely, a tank and car in Fig. 5.3 and a helicopter in Fig. 5.4. From these figures, we notice that the sound pressure levels of the tank and helicopters are high. It is also clear from the figures that the sound consists of several harmonics. On the other hand, the pressure levels for the car are much lower and the energy is distributed in broadband signals.

The sound pressure level of a spherical wave in an unbounded homogeneous atmosphere is given by (5.3); however, the sound sources are close to the ground and hence the pressure levels are affected by the ground reflections. In the next section, we present the techniques used to compute pressure levels affected by ground reflections.

5.1 Reflection of Spherical Waves

It is important to study the effect of the reflection of waves as they are superimposed on the direct waves and significantly alter the sound pressure levels. If the reflecting waves add in phase, the pressure of the total sound wave increases; alternately, it decreases if the direct wave and reflecting waves are in opposite phase.

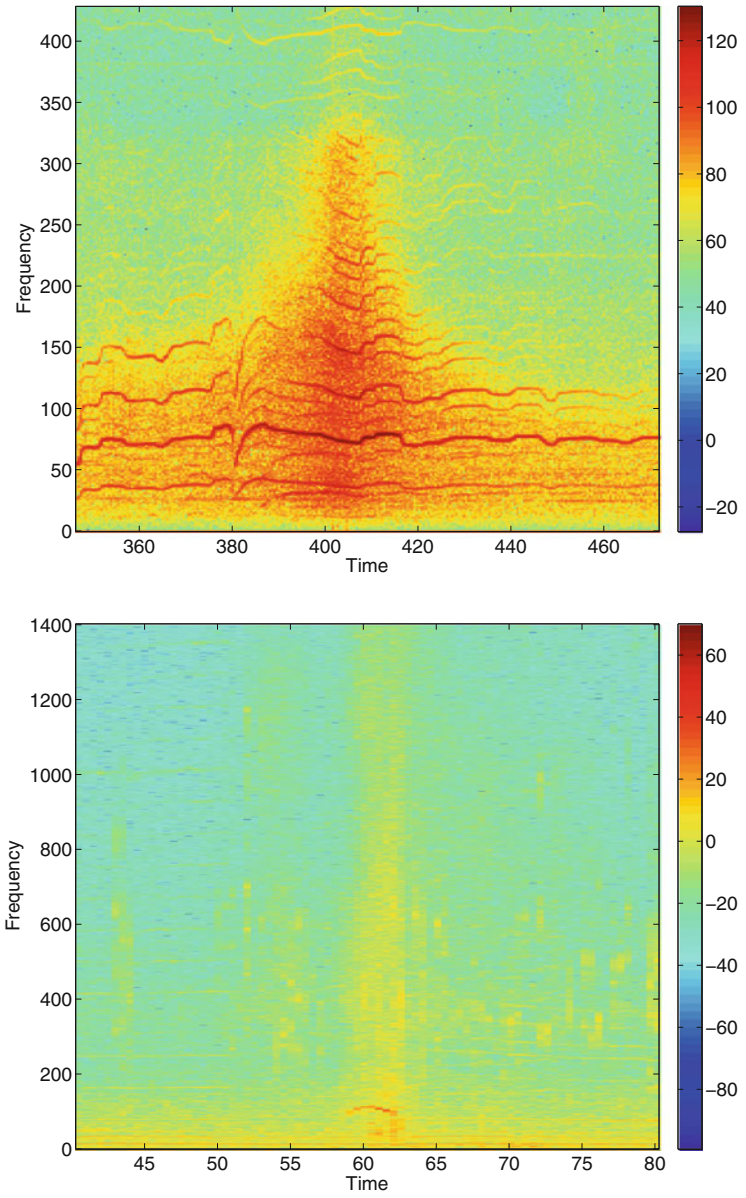


Fig. 5.3 Spectrograms of a **a** tank and **b** car

Suppose the incidence of a harmonic wave is perfectly normal to the boundary as shown in Fig. 5.5. Part of the wave is reflected and part is transmitted. The incident, reflected, and transmitted waves are distinguished by the subscripts i, r, and t, respectively. Let the density and wave velocities be ρ_1 and c_1, respectively, for the

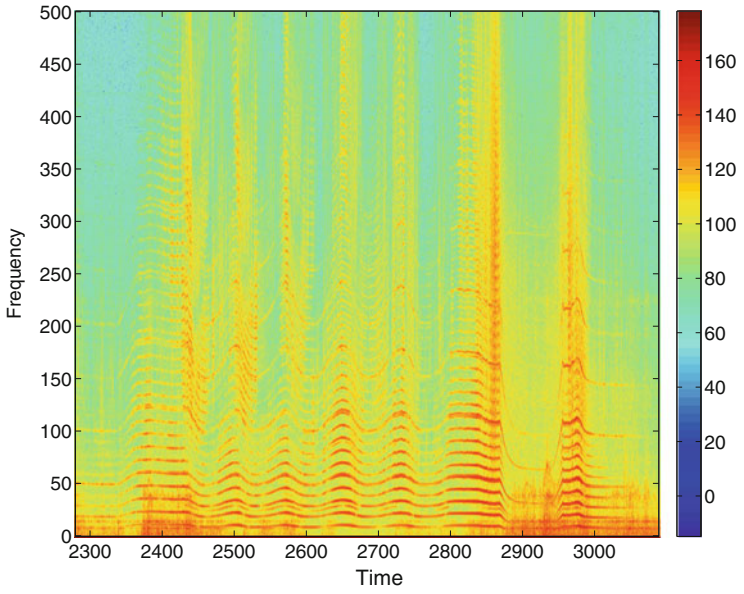


Fig. 5.4 Spectrogram of a helicopter

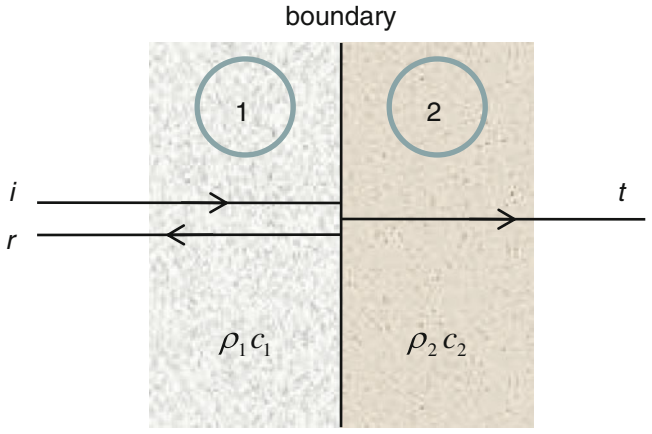


Fig. 5.5 Partial reflection of waves at boundary between two media

medium 1 and ρ_2 and c_2, for medium 2. The particle velocity and acoustic pressure must be continuous across the boundary [77]. This leads to

$$\nabla v_i + \nabla v_r = \nabla v_t \tag{5.6}$$

and

$$p_i + p_r = p_t \tag{5.7}$$

where ∇v_i is the displacement and v_i corresponds to the particle velocity of the incident wave. The direction of the particle velocity of the incident wave is assumed to be positive. The acoustic pressure is associated with the displacement by

$$p = \frac{\beta}{c}\,\nabla v = \rho c \nabla v, \tag{5.8}$$

where β is the bulk modulus. Substituting (5.8) in (5.6) and reversing the sign for the reflected wave as it is traveling in the negative direction, we get

$$\rho_1 c_1 \nabla v_i - \rho_1 c_1 \nabla v_r = \rho_2 c_2 \nabla v_t. \tag{5.9}$$

From (5.6) and (5.9) and eliminating ∇v_t, we get

$$\frac{\nabla v_r}{\nabla v_i} = \frac{\rho_1 c_1 - \rho_2 c_2}{\rho_1 c_1 + \rho_2 c_2}. \tag{5.10}$$

Similarly, by eliminating ∇v_r, we get

$$\frac{\nabla v_t}{\nabla v_i} = \frac{2\rho_1 c_1}{\rho_1 c_1 + \rho_2 c_2}. \tag{5.11}$$

Acoustic Impedance

The acoustic impedance, z_s, at a point in the field of sound is defined as

$$z_s = \frac{p}{v} \tag{5.12}$$

where p and v are the excess pressure and particle velocity, respectively.

We note that the ratio of particle displacements is identical to the ratio of the velocities. Hence, $\nabla v_t / \nabla v_i$ can be replaced by v_t / v_i. Now, let us look at the intensity of the wave, which is given by $(\rho c v_{\text{rms}}^2)$. By replacing the displacements with velocities, we can compute the relative intensities of the waves as

$$\frac{I_r}{I_i} = \frac{\rho_1 c_1 (v_r)_{\text{rms}}^2}{\rho_1 c_1 (v_i)_{\text{rms}}^2} = \left(\frac{\rho_1 c_1 - \rho_2 c_2}{\rho_1 c_1 + \rho_2 c_2}\right)^2 = \left(\frac{(z_s)_1 - (z_s)_2}{(z_s)_1 + (z_s)_2}\right)^2, \tag{5.13}$$

and

$$\frac{I_t}{I_i} = \frac{\rho_2 c_2 (v_t)^2_{\text{rms}}}{\rho_1 c_1 (v_i)^2_{\text{rms}}} = \frac{\rho_2 c_2}{\rho_1 c_1} \left(\frac{2\rho_1 c_1}{\rho_1 c_1 + \rho_2 c_2} \right)^2 = \frac{(z_s)_2}{(z_s)_1} \left(\frac{2(z_s)_1}{(z_s)_1 + (z_s)_2} \right)^2. \quad (5.14)$$

For a non-dissipative medium, $I_i = I_r + I_t$.

Relative phase of the acoustic pressures

By simple manipulation of (5.6) and (5.7) and making use of the relation (5.8), we get

$$\frac{p_r}{p_i} = \frac{\rho_2 c_2 - \rho_1 c_1}{\rho_1 c_1 + \rho_2 c_2} = \frac{(z_s)_2 - (z_s)_1}{(z_s)_1 + (z_s)_2}, \quad (5.15)$$

and

$$\frac{p_t}{p_i} = \frac{2\rho_2 c_2}{\rho_1 c_1 + \rho_2 c_2} = \frac{2(z_s)_2}{(z_s)_1 + (z_s)_2}. \quad (5.16)$$

From (5.15), we notice that if $\rho_1 c_1 > \rho_2 c_2$, then p_r is 180° out of phase; when $\rho_1 c_1 < \rho_2 c_2$, p_r is in phase with p_i and reinforcement occurs. If the incident wave does not strike the boundary at 90°, we only need to consider the normal component of the particle velocities in the incident wave, since the component parallel to the boundary remains unaffected.

Now consider the case depicted in Fig. 5.6 with a source and a receiver located at $(0, z_s)$ and (r, z), respectively. The received signals consists of two parts (a) waves traveling along the direct path and (b) waves reflected by the ground. Using (5.3), the received complex acoustic pressure can be written as

$$p_c = S \frac{\exp(ikR_1)}{R_1} + QS \frac{\exp(ikR_2)}{R_2} \quad (5.17)$$

where $R_1 = \sqrt{r^2 + (z - z_s)^2}$ and $R_2 = \sqrt{r^2 + (z + z_s)^2}$. The quantity Q is called the reflection coefficient and can be calculate using (5.15).

The reflection coefficient Q is a function of the following:

- wave number k (or frequency f),
- distance R_2,
- reflection angle θ,
- normalized ground impedance z.

There are several techniques [81] available to measure the normalized ground impedance.

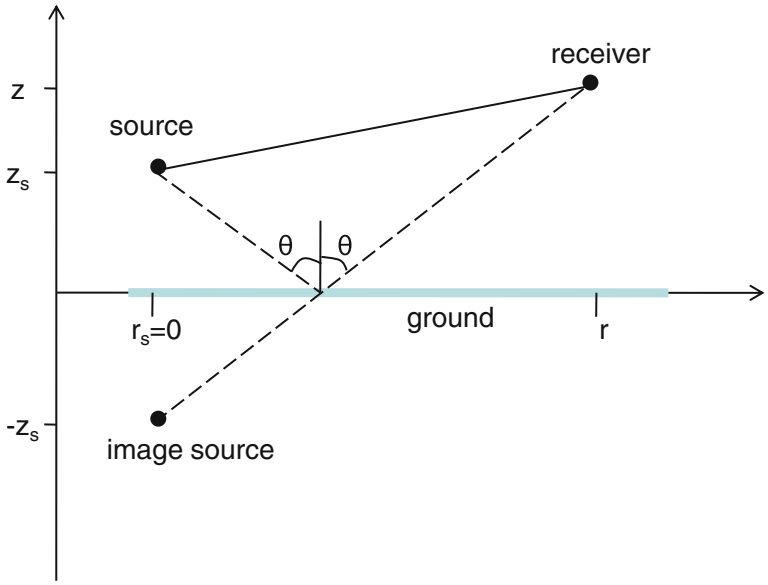


Fig. 5.6 Reflection due to the ground

5.2 Atmospheric Refraction

In the previous section, it was assumed that the sound propagated in a non-refracting atmosphere over a ground surface. However, this is only true if the propagation distance is less than ∼100 m. Atmospheric refraction has a large effect on acoustic pressure if the distances are larger than 100 m, in particular, if the source and receivers are close to the ground at heights of a few meters. Some of the effects on sound waves due to refraction are shown in Fig. 5.7. The refraction is caused due to the temperature change with height. If the receiver is in a shadow region (see Fig. 5.7), it will not receive the sound transmitted by the source. For the case where the speeds of the sounds are discontinuous, as shown in Fig. 5.8, the refraction of the sound waves obey Snell's law:

$$\frac{\cos\gamma_1}{c_1} = \frac{\cos\gamma_2}{c_2} \tag{5.18}$$

The above equation is for discrete speeds in the media. However, in the atmosphere, the temperature change is not discrete, rather it is continuous, hence, the speed of the sound is also continuous. For continuous speeds, Snell's law stipulates that the ratio

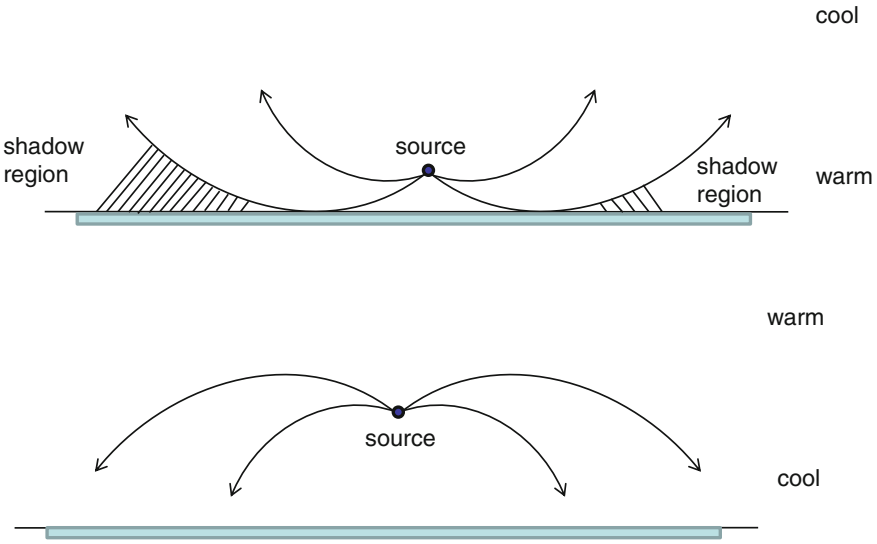


Fig. 5.7 Refraction due to temperature change **a** upward and **b** downward

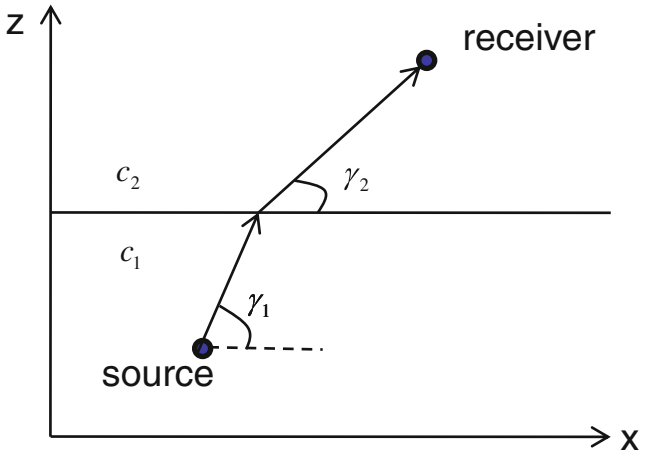


Fig. 5.8 Refraction at the boundary of two media

$$\frac{\cos\gamma(z)}{c(z)} = k, \tag{5.19}$$

where k is some constant, $\gamma(z)$ and $c(z)$ are functions of z. In Fig. 5.8, $\gamma_2 < \gamma_1$, which corresponds to the case $c_2 > c_1$, where the atmosphere is called a downward

refracting atmosphere. On the other hand, if $\gamma_2 > \gamma_1$ and $c_2 < c_1$, the atmosphere is considered an upward refracting atmosphere.

Effective Sound Speed

So far we have assumed a non-moving atmosphere; however, there may be a wind preset moving at a certain speed. If so then the effective sound speed can be approximated as

$$c_{\text{eff}}(z) = c(z) + u(z), \tag{5.20}$$

where $u(z)$ is the component of the wind velocity in the direction of the sound propagation.

There are several numerical methods available in the literature for sound propagation in a refracting atmosphere over a ground surface:

- the Fast Field Program (FFP),
- the Crank-Nicholson Parabolic equation method, and
- the Green's function Parabolic equation method.

The details of these methods can be found in [5, 8, 81]. These references also provide methods for dealing with rough terrain and turbulence in atmosphere. Their discourse is beyond the scope of this book.

5.3 Compensation for Propagation Delay

Unlike microwaves, sound waves travel at relatively slow speed, 340 m/s. If we are tracking vehicles that are far away, then the time the sound takes to reach the receiver is significant. For example, one can track a helicopter using several microphone sensor arrays at a distance greater than 3 km. So, if the helicopter is 3 km from a sensor array, then it takes roughly 9 s for the sound to reach the array. Figure 5.9 shows the direction of arrival (DOA) (azimuth) angle estimation of a helicopter and the ground truth (the actual angle). We notice that the measured azimuth angle is different from that of the ground truth data. Moreover, it lags behind the truth data sometimes and in at other times is leading it. The difference between the ground truth and measured data is due to the propagation delay, since the ground truth data corresponds to the target location, while the measurements are made at the sensor location. The measurements ϕ can be compensated using [22]

$$\hat{\phi} = \phi + \tau\, \delta\phi \tag{5.21}$$

where $\tau = d/c$ is the time delay, d—distance between the sensor and the target, and $\delta\phi$ is the change in the measured angles in 1 s. The compensated angles are also plotted and shown in Fig. 5.9. Although, they tend to be closer to the true angles,

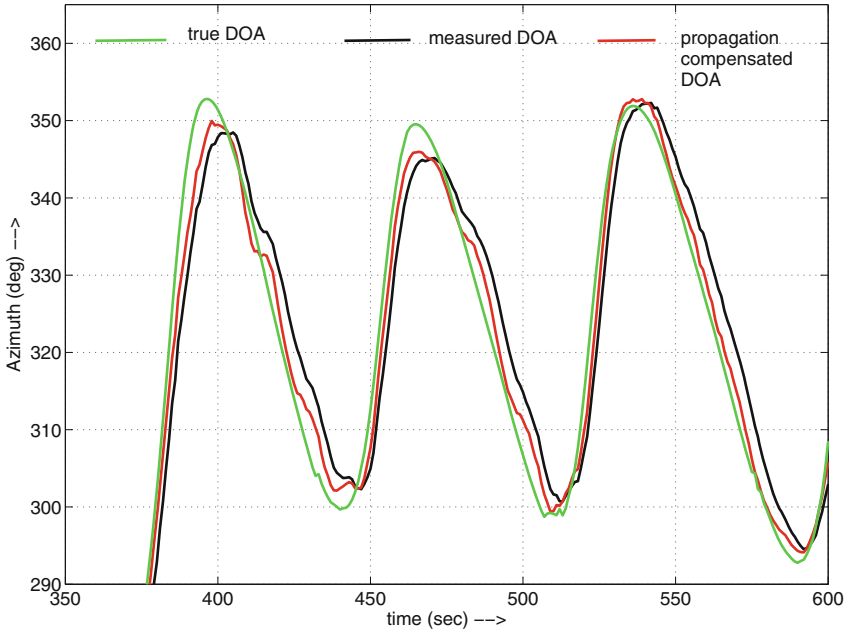


Fig. 5.9 Compensation for propagation delay

still there is some difference. This could be due to the fact that the estimation of distance d is not quite as accurate as it is made using noisy azimuth and elevation angles measured by multiple arrays.

Chapter 6
Acoustic Arrays

The majority of applications for acoustic sensors in modern warfare concern the following: (a) sniper localization, finding the location of the shooter; (b) mortar detection, launches and detonations; (c) rocket-propelled grenade (RPG) detection, launches and detonations; (d) vehicle detection, tracking and classification; and (e) personnel detection. Invariably, these applications use acoustic sensors in very different ways, either as distributed microphones or an array of microphones. Sensor arrays consisting of 3–16 microphones have been used for example, in sniper/mortar detection using a sensor array of 4 microphones with the microphones placed at the vertices of a tetrahedron, as shown in Fig. 6.1; and in tracking vehicles using an array of 7 microphones with six of them placed at the vertices of a hexagon and the 7th microphone placed at the center is used. Other applications feature other configurations, with each application using the array configuration that is optimal for its mission.

Here, we provide the theory behind the arrays and estimate the response of the array in resolving two closely located targets. This theory provides a guideline as to what configuration is optimal for a given application. For instance, a uniform linear array (ULA), one consisting of microphones deployed in straight line, has a 180° ambiguity, thus an ULA cannot resolve on which side of the array target is. In contrast, a circular array has a 360° coverage. Although the typical frequency range of interest for vehicle tracking is 20–500 Hz, the maximum frequency at which a given array can operate depends on the aperture of the array. Surpassing the maximum operating range of the array leads to spatial aliasing making the signals coming from different angles indistinguishable. For linear arrays, determining the maximum frequency of operation is straightforward and is given by $\frac{c}{2d}$, where c is the propagation velocity of sound and d denotes the spacing between two elements of the array. In the case of circular arrays, one of the criteria for selecting the radius of the array and the number of microphones in the array depends on the angular resolution needed to distinguish multiple targets that are closely spaced.

T. Damarla, *Battlefield Acoustics*, DOI 10.1007/978-3-319-16036-8_6

Fig. 6.1 Tetrahedral acoustic sensor array

In this chapter, we provide the theory pertinent to calculating the beam width of an array, which is needed to determine the resolution capability. The beam width depends on the frequency of operation, number of microphones and the spacing between them.

6.1 Uniform Linear Arrays

Figure 6.2 shows an uniform linear array with 4 microphones on which a plane wave is impinging at an angle ϕ. The signal wave is generated by an acoustic target, such as a vehicle, gunshot, etc. The natural question that comes to mind is from what direction is the target sound coming from? Humans and many animals are equipped with the ability to tell the direction of a sound. Our ears process the sound and more or less tell us the direction from where the sound is coming. In fact, one of the main goals of array processing is to determine the direction of arrival (DOA) of the signal. In order to develop the theory behind estimating the direction of arrival of the signal,

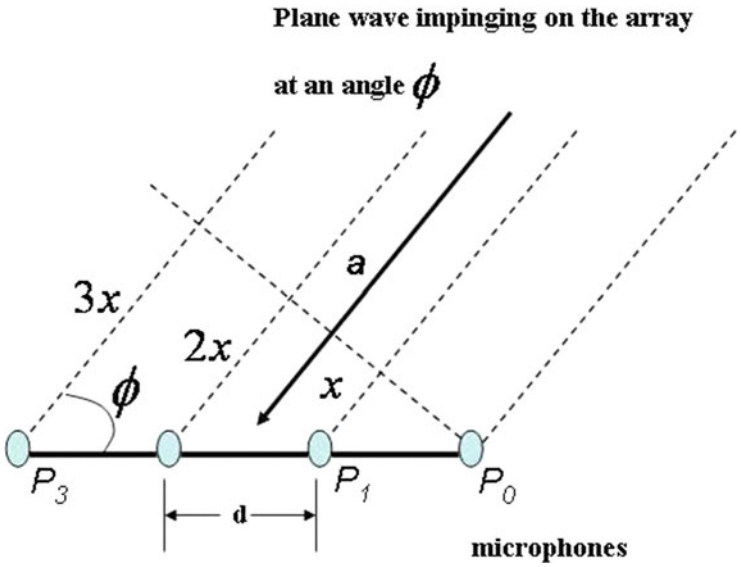


Fig. 6.2 Uniform linear array

let us assume that the target is emitting a signal $s(t)$ and the impulse response of the channel between the source and sensor is $h(t)$, then output of the ith sensor is given by

$$y_i(t - \tau_i) = h(t - \tau_i) * s(t)$$

where '*' denotes the convolution operation and τ_i is the time delay. Let the unit vector to the target be denoted by

$$\mathbf{a}(\phi, \theta) = [\cos\theta\cos\phi \quad \cos\theta\sin\phi \quad \sin\theta]^T$$

where ϕ is the azimuth angle, θ is the elevation angle as shown in Fig. 6.3 and T denotes the transpose. Note that we used bold face for the vector and regular font for the scalar. Let $\mathbf{p}_i = [x_i \; y_i \; z_i]^T$ denote the coordinates of the ith sensor, then the time delay τ_i is given by

$$\tau_i = \frac{\mathbf{a}^T \mathbf{p_i}}{c} \tag{6.1}$$

where 'c' is the propagation velocity of sound.

One of the simplest forms of signal processing is to add all the outputs from the sensors in an array to improve the signal quality, that is SNR. The rationale is that the noise is un-correlated and hence would cancel out to a great degree while the signals add. One would improve the signal level if they were added in phase. This implies that we have to delay and add the signals in phase as shown in Fig. 6.4.

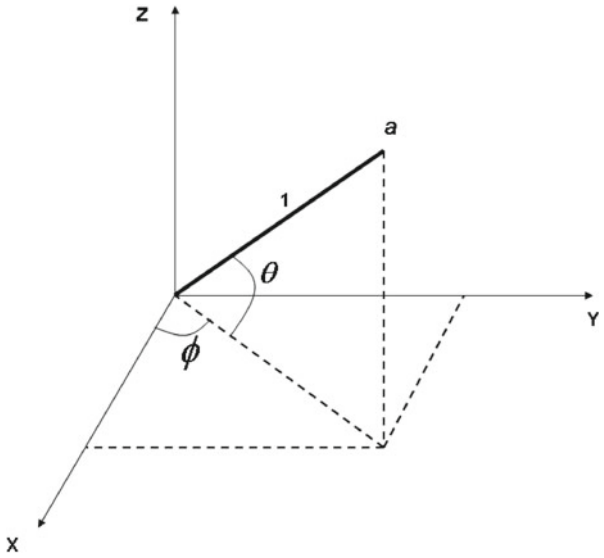


Fig. 6.3 Unit vector

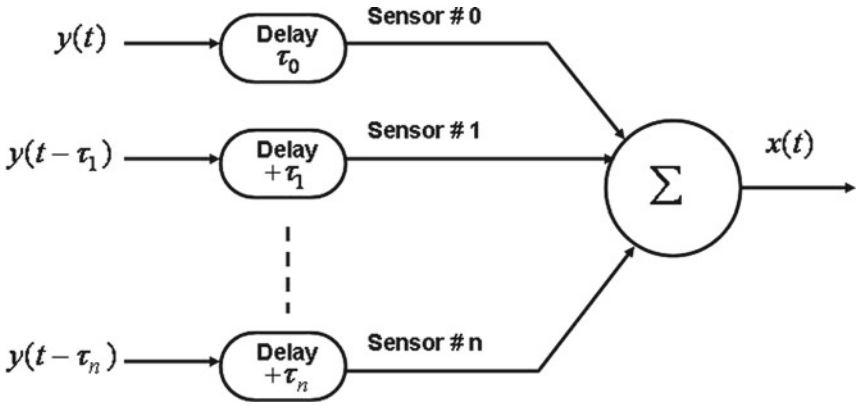


Fig. 6.4 Delay and sum

We represent the signal $y(t) = A\ e^{i2\pi f_c t}$ in complex notation, where $i = \sqrt{-1}$, and f_c is the carrier frequency of the signal. Then the output of the jth microphone is

$$y(t - \tau_j) = A\ e^{iw_c(t-\tau_j)} = A\ e^{iw_c t}\ e^{-iw_c\tau_j}$$

Substituting (6.1) in $e^{-iw_c\tau_j}$ and writing for all n in a vector form, we get

$$\mathbf{v} = \left[e^{-i\frac{2\pi}{\lambda}\mathbf{a}\bullet\mathbf{p_0}}\ \ e^{-i\frac{2\pi}{\lambda}\mathbf{a}\bullet\mathbf{p_1}} \cdots e^{-i\frac{2\pi}{\lambda}\mathbf{a}\bullet\mathbf{p_n}}\right]^T \tag{6.2}$$

where $\lambda = \frac{c}{f}$ is the wave length and $\mathbf{a} \bullet \mathbf{p_j}$ is the dot product between the vectors $\mathbf{a}$ and $\mathbf{p_j}$. The vector $\mathbf{v}$ is called the steering vector. Often, the quantity

$$\mathbf{k} = \frac{w}{c} = \frac{2\pi}{\lambda} \tag{6.3}$$

is called the *wavenumber*. The output vector from the sensor is

$$\mathbf{Y} = A\, e^{i w_c t}\, \mathbf{v} \tag{6.4}$$

Let us define a vector of delays given in Fig. 6.4 by

$$\mathbf{W} = [w_0\ w_1 \cdots w_n]^T = \left[e^{i w \tau_0}\ e^{i w \tau_1} \cdots e^{i w \tau_n}\right]^T \tag{6.5}$$

The output after summation in Fig. 6.4 is give by

$$x(t) = W^T\ \mathbf{Y} = n\ y(t) \tag{6.6}$$

hence there is a gain of n-fold in the signal by delaying the signals and adding the outputs of n microphones in phase. Clearly, the gain is obtained by electronically rotating the array perpendicular to the unit vector in the direction of target. This is also called the beam forming, that is, rotating the array by shifting the phase angles. We just showed that the gain is highest if we rotate the array perpendicular to the direction of the signal propagation. It is informative to understand how the gain decreases away from that direction. In order to derive the equation, let us set the weight vector W to

$$W = \left[\frac{1}{n}\ \frac{1}{n}\ \cdots \frac{1}{n}\right] \tag{6.7}$$

then from (6.4) and (6.6) we get $x(t)$ as

$$x(t) = W^T\ \mathbf{Y} = \frac{A}{n}\, e^{i w_c t} \sum_{j=1}^{n} e^{-i\ \mathbf{k} \bullet \mathbf{p}_j} \tag{6.8}$$

From Fig. 6.2, and from Eq. (6.3) we find that $-\mathbf{k} \bullet \mathbf{p}_0 = -\frac{2\pi}{\lambda}\mathbf{a} \bullet \mathbf{p}_0 = 0$, $-\mathbf{k} \bullet \mathbf{p}_1 = \frac{2\pi}{\lambda} d \cos\phi$, and $-\mathbf{k} \bullet \mathbf{p}_n = \frac{2\pi}{\lambda}(n-1)d \cos\phi$. Substituting these values in (6.8) we obtain

$$x(t) = \frac{A}{n}\, e^{i w_c t} \sum_{j=1}^{n} e^{i\ \frac{2\pi}{\lambda}\ (j-1) d \cos\phi} = \frac{A}{n}\, e^{i w_c t}\, \frac{e^{i\ n\psi} - 1}{e^{i\ \psi} - 1} \tag{6.9}$$

where $\psi = \frac{2\pi}{\lambda} d \cos\phi$. The above equation can be re-written to make its nature apparent.

$$x(t) = \frac{A}{n}\, e^{i w_c t}\, \frac{e^{i\, n\psi} - 1}{e^{i\, \psi} - 1} = \frac{A}{n}\, e^{i w_c t}\, \frac{e^{i\, \frac{n\psi}{2}}}{e^{i\, \frac{\psi}{2}}}\, \frac{e^{i\, \frac{n\psi}{2}} - e^{-i\, \frac{n\psi}{2}}}{e^{i\, \frac{\psi}{2}} - e^{-i\, \frac{\psi}{2}}}$$

or

$$x(t) = \frac{A}{n}\, e^{i w_c t}\, e^{i\, \frac{(n-1)\psi}{2}}\, \frac{\sin\left(\frac{n\psi}{2}\right)}{\sin\left(\frac{\psi}{2}\right)} \tag{6.10}$$

Hence, the output of the array is a sync function $\frac{\sin(n\psi)}{\sin(\psi)}$.

The microphone array is characterized by its ability to resolve two closely spaced targets. In order to understand the resolution capability of the linear array, the output of the array is plotted in Fig. 6.5, where each curve corresponds to the response of an array with different number 'n' of microphones in the array. As the number of microphones increase, the output response of the array is sharper, hence increasing the resolving capability of the array. The response of the array is identical for the segments $\phi = 0$–180 and $\phi = 180$–360. Hence, the ULA has 180° ambiguity, that is, it can not distinguish the objects from one side to the other side of the array. The peak response occurs when $\phi = 90$ (270) since that is the angle at which the signal arrives with same phase at all microphones.

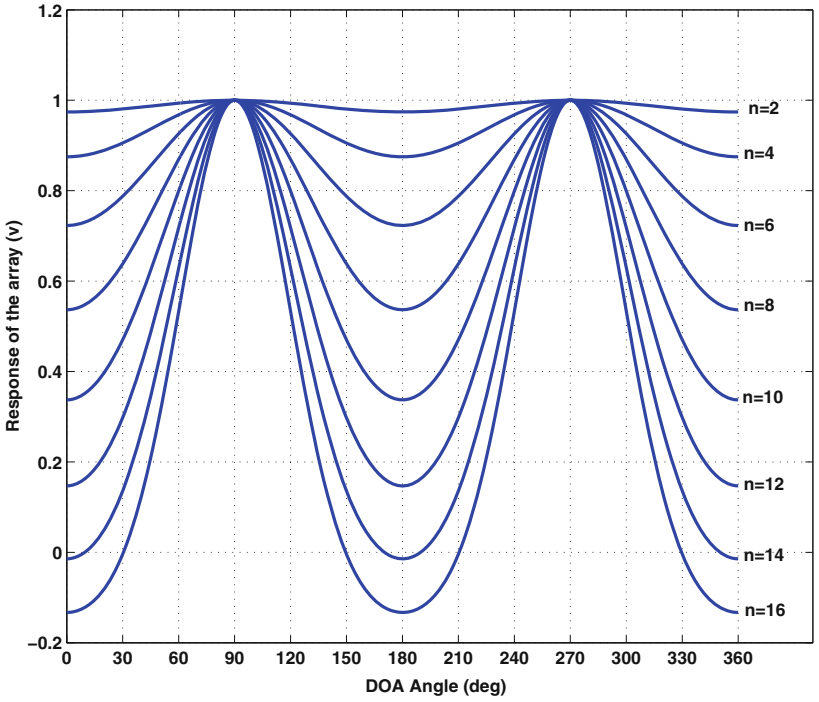


Fig. 6.5 Response of the ULA for different 'ϕ' with a different number of microphones 'n'

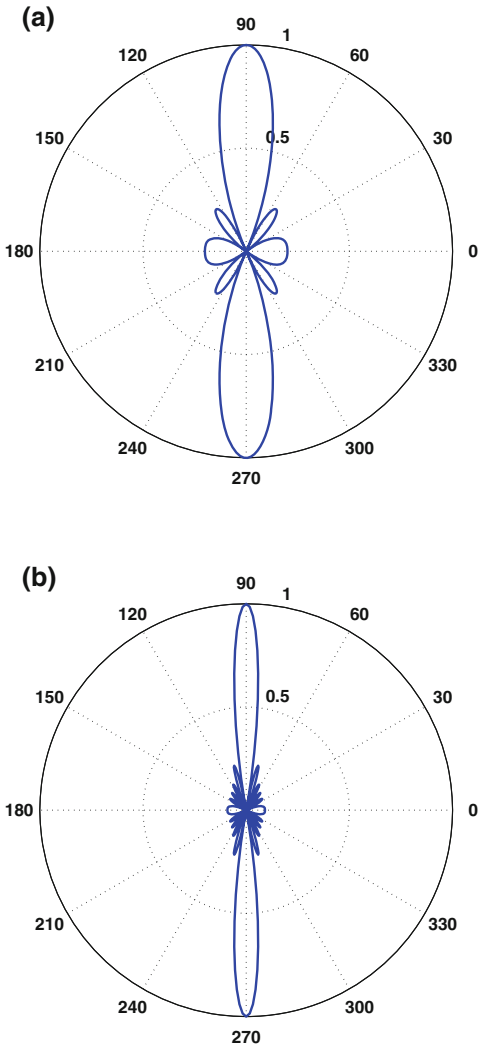


Fig. 6.6 Beam pattern $B_\theta(\theta)$ of ULAs with $d = 0.5\lambda$. **a** Beam Pattern for 5-element array. **b** Beam Pattern for a 11-element array

The beam pattern of the array is given by

$$B_\theta(\theta) = \frac{1}{n}\frac{\sin\left(\frac{n\psi}{2}\right)}{\sin\left(\frac{\psi}{2}\right)} = \frac{1}{n}\frac{\sin\left(\frac{n}{2}\cdot\frac{2\pi}{\lambda}\cos\theta\cdot d\right)}{\sin\left(\frac{1}{2}\cdot\frac{2\pi}{\lambda}\cos\theta\cdot d\right)}, \quad 0 \le \theta \le \pi. \tag{6.11}$$

and its plot is shown in Fig. 6.6.

Beam width is defined as the angle between the half power (–3 db) points of main lobe. The beam pattern in Fig. 6.6 also shows the 180° symmetry and it is clearly seen that the beam width is larger for 5-element ULA compared to an 11-element ULA.

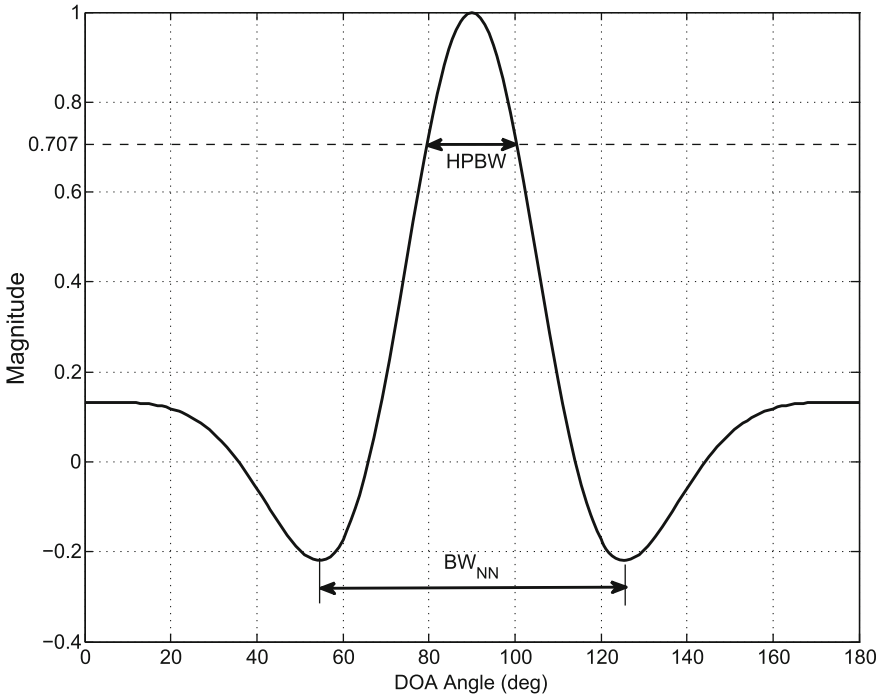


Fig. 6.7 Beam pattern $B_\theta(\theta)$ of ULAs with $d = 0.5\lambda$

The half power beam width is marked as HPBW in Fig. 6.7. The distance between two null points is denoted as the null to null beamwidth (BW_{NN}), also shown on Fig. 6.7. As mentioned earlier, the beam width is dependent on the aperture of the array. The aperture of the uniform linear array is given as the distance between two extreme elements (microphones) of the array and is given by

$$\mathcal{A} = (n-1)\, d. \tag{6.12}$$

Another characteristic of the array is the ability to resolve the targets without any ambiguity. As per the Rayleigh resolution criteria, two plane waves (presumably generated by two targets) are considered resolvable if the peak of the second beam pattern falls on or outside the null of the first pattern, that is, the angle separation has to be $\geq 0.5\ BW_{NN}$.

Grating Lobes: The response of the uniformly weighted linear array is plotted in Fig. 6.8 for different values of spacing between two microphones d as a function of DOA angle for an ULA with 9 microphones. The main lobe has the magnitude of '1' when the DOA angle is $\pm 90°$, that is, in the broad side for the cases when $d \leq 0.5\lambda$. However, when $d = \lambda$, there is another lobe at $0°$ and $\pm 180°$ with the same magnitude as the main lobe. These additional lobes with the same magnitude as the main lobe are called the grating lobes. These grating lobes occur when the

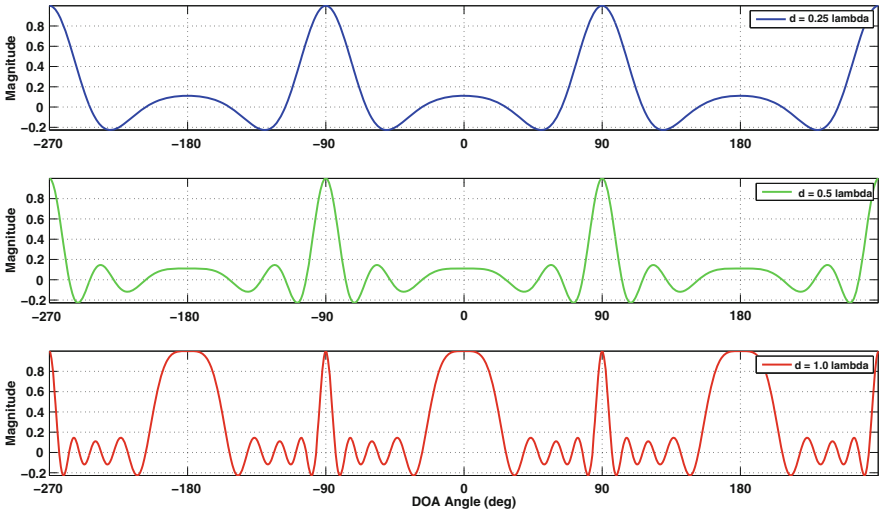


Fig. 6.8 Response of the ULA for different spacing d

numerator and denominator of Eq. (6.11) are equal to zero. The effect of the grating lobes is identical to that of aliasing in time series analysis when the time-domain waveform is undersampled.

6.1.1 Wavenumber Transforms

Wavenumber transforms are widely used in sonar and radar systems. Wavenumber $k = \frac{\omega}{c} = \frac{2\pi}{\lambda}$ is defined as the radian frequency 'ω' over the wave speed 'c' with dimensions of radians per meter. Wavenumbers can be thought of as a measure of wavelength relative to 2π. In the frequency domain, the phase response relates to the time delay, whereas in phase delay in the wavenumber domain relates to the spatial position. Consider the two-element array in Fig. 6.9, the plane wave is impinging on the array at an angle 'θ' and the additional distance the plane wave has to travel to the second element is '$R = d\cos\theta$'. The phase difference between the two array elements is simply kR, where k is the wavenumber. The bearing angle (direction of arrival) to the distant source can be determined by first measuring the phase difference $\Delta\phi_{21}$ between the two sensors p_1 and p_2

$$\Delta\phi_{21} = kR = kd\cos\theta$$

$$\theta = \cos^{-1}\left(\frac{\Delta\phi_{21}}{kd}\right). \tag{6.13}$$

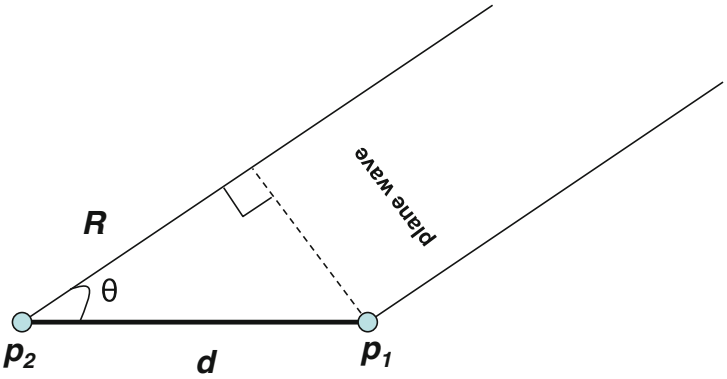


Fig. 6.9 Depending on the angle of incidence, the phase across the array changes

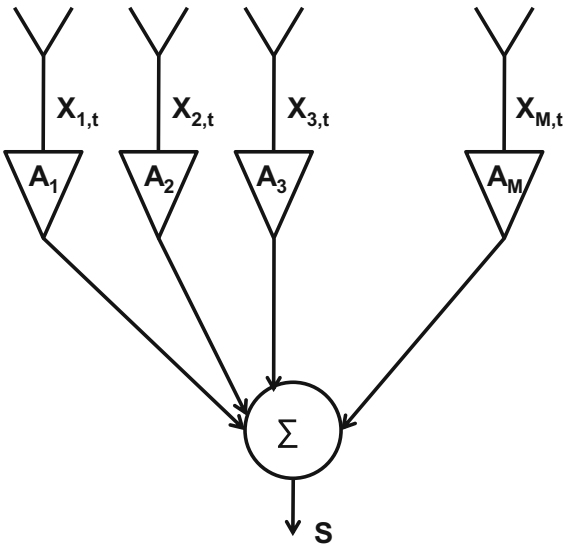


Fig. 6.10 Uniform linear array to determine the DOA of plane waves

Now consider a source at a distance R radiating the plane wave $P_0 e^{i\omega t}$ with a radian frequency ω. Since it is travelling a distance R, the phase at the first element of the array is simply kR and it is given by $P_0 e^{i(\omega t - kR)}$. If the bearing angle to the source is θ, then the delay sum output of the uniform linear array shown in Fig. 6.10 can be written as

$$S = P_0 A_1 e^{i(\omega t - kR)} + P_0 A_2 e^{i(\omega t - kR + kd\cos\theta)} + \cdots + P_0 A_M e^{i(\omega t - kR + (M-1)kd\cos\theta)},$$

or

$$S = P_0 e^{i(\omega t - kR)} \sum_{m=1}^{M} A_m e^{+ik(m-1)d\cos\theta} \tag{6.14}$$

The above equation has the same mathematical structure as the Fourier transform. The output S depends on the relative phases across the array. We have shown earlier it depends on the spacing of the array elements d with respect to λ. So $\frac{d}{\lambda}$ plays an important role in array processing.

6.1.2 Beam Steering

Beam steering is a technique used to search for the signals in a particular direction. Mechanical devices such as antennas, parabolic dishes, etc., are physically rotated to capture the signal in a particular direction. In the case of an array, it can be done electronically by subtracting the appropriate phase from each sensor. Hence, to determine if there is a target (radiating source) in the direction θ_d, simply multiply mth sensor output by

$$A_m = e^{-i2\pi \frac{d(m-1)}{\lambda} \cos\theta_d}. \tag{6.15}$$

One can look (steer the beam) simultaneously in several directions using parallel processing techniques. It can be accomplished by generating the steering vectors (Eq. 6.15) for different angles θ_d and multiplying the outputs of the sensors before summing using Eq. (6.14). The output of an 11-element uniform linear array with half lambda spacing that is steered 90° is shown in Fig. 6.11. The output magnitude in the

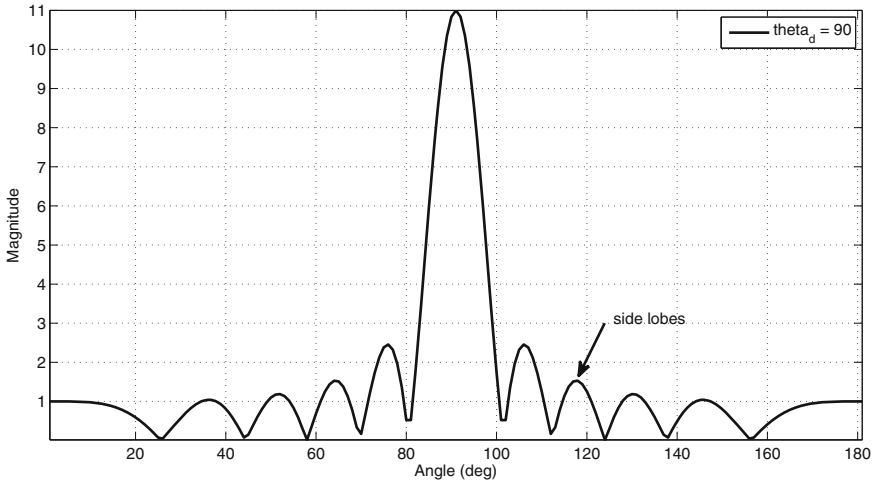


Fig. 6.11 Output of 11-element uniform linear array steered 90°

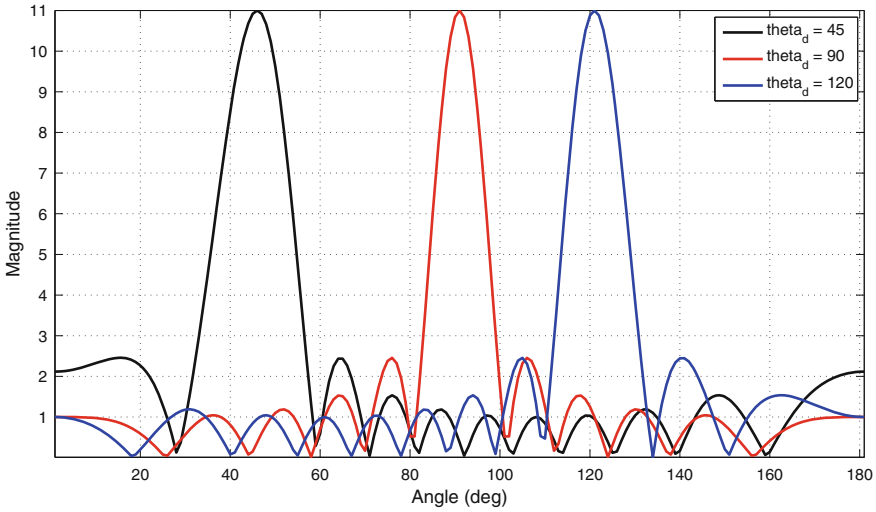


Fig. 6.12 Output of 11-element uniform linear array steered in different directions

direction of the steered angle is high compared to the output in the other angles. There are several sidelobes visible in Fig. 6.11. In general, the sidelobes are suppressed by using different windows, such as, hanning or rectangular windows. Figure 6.12 shows the output of the uniform linear array steered in different directions.

6.2 Circular Arrays

One of the most widely used arrays is the circular array. Unlike linear arrays, circular arrays do not have a 180° ambiguity. In fact, they cover all 360°. Clearly, the minimum number of microphones in a circular array is 3; however, having more microphones in the array results in a higher resolution. In other words, a circular array with more microphones will be able to distinguish targets that are closely spaced compared to an array with fewer microphones. We now present some of the properties of circular arrays.

Let us assume that there are 'n' number of microphones in a circular array, and the angle between any two microphones is $\gamma = \frac{2\pi}{n}$. It is a normal practice to place the first microphone on the y-axis, which is normally aligned with the true north and the rest of the microphones are arranged clockwise. For example, there are $n = 5$ microphones in Fig. 6.13b with the first microphone is on the axis corresponding to true north indicated by '**N**' and the angle $\gamma = 72°$. The rest of the microphones are arranged clockwise.

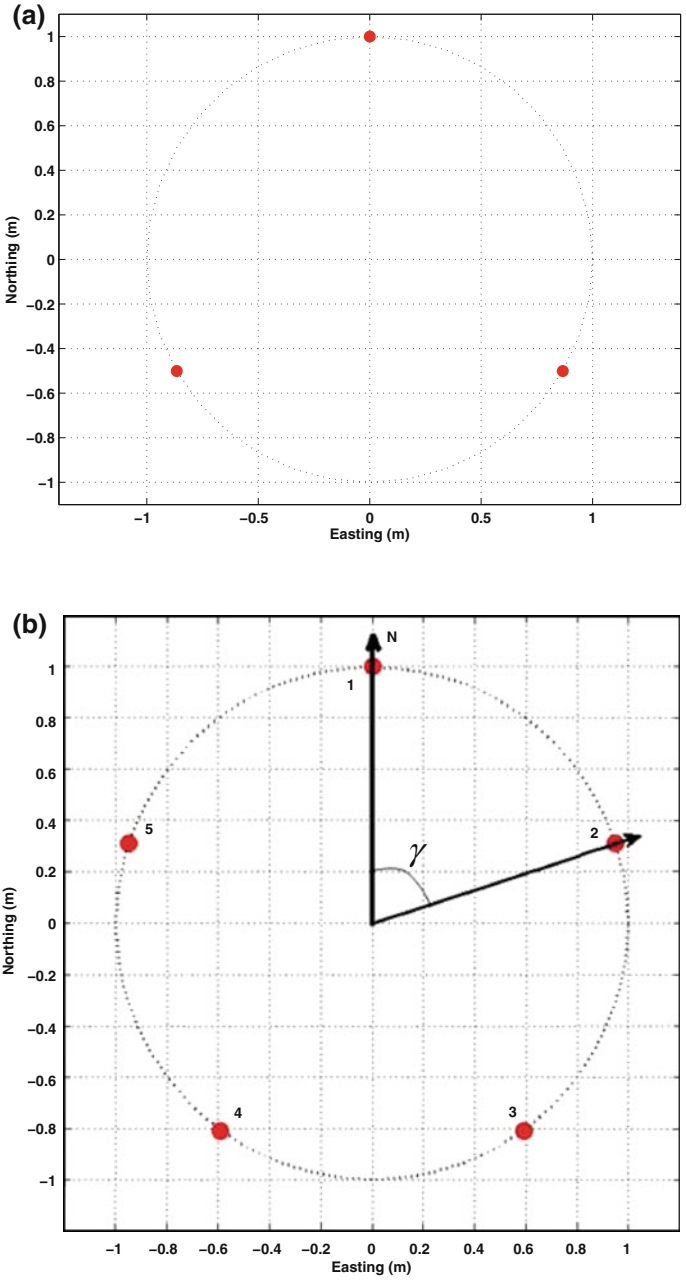


Fig. 6.13 Examples of Circular arrays. **a** 3 microphone circular array. **b** 5 microphone circular array

In order to understand the performance of the circular array, the outputs of the microphones will be delayed and added as shown in Fig. 6.4. Let the positions of microphones on the array represented by $\mathbf{P} = \{\mathbf{p_0}, \mathbf{p_1}, \ldots, \mathbf{p_{n-1}}\}$. Then, the delay $\tau_j = \frac{\mathbf{a}^T \mathbf{p_j}}{c}$ is given by (6.1), where $\mathbf{p_j} = R\left[\cos\gamma_j \;\; \sin\gamma_j \;\; 0\right]$, where R is the radius of the circular array, and $\gamma_j = \frac{2\pi j}{n}$ and the unit vector $\mathbf{a}(\phi, \theta) = [\cos\theta\cos\phi \;\; \cos\theta\sin\phi \;\; \sin\theta]^T$. Then the output of the jth microphone is

$$y(t - \tau_j) = A\, e^{\{iw_c(t-\tau_j)\}} = A\, e^{iw_c t}\, e^{-iw_c\tau_j}$$

Substituting τ_j in the above equation results in

$$y(t - \tau_j) = A\, e^{iw_c t}\, e^{-iw_c\tau_j} = A\, e^{iw_c t}\, e^{-i\frac{2\pi}{\lambda}R\,\cos\theta\,\cos(\phi-\gamma_j)}$$

Then the steering vector is given by

$$\mathbf{v} = \left[e^{-i\frac{2\pi}{\lambda}R\,\cos\theta\,\cos(\phi-\gamma_0)},\; e^{-i\frac{2\pi}{\lambda}R\,\cos\theta\,\cos(\phi-\gamma_1)}, \ldots, e^{-i\frac{2\pi}{\lambda}R\,\cos\theta\,\cos(\phi-\gamma_{n-1})}\right]^T \tag{6.16}$$

Now, to find the response of the uniform circular array, we perform the delay and sum operation as shown in Fig. 6.4 and the output is given by

$$x(t) = W^T Y = \frac{A}{n}\, e^{-iw_c t} \sum_{j=0}^{n-1} e^{i\,k\,\cos(\phi-\gamma_j)} \tag{6.17}$$

where $k = \frac{2\pi}{\lambda}\, R\, \cos\theta$. In order to get the closed form solution for (6.17), we invoke the Jacobi-Anger [17, 19] expansion given by

$$e^{i\,x\,\cos\alpha} = \sum_{m=-\infty}^{\infty} i^m\, J_m(x)\, e^{-i\,m\,\alpha} \tag{6.18}$$

where $J_m(x)$ is the Bessel function of the 1st kind of order m. Then the qth term in (6.17) becomes

$$e^{i\,\frac{2\pi}{\lambda}\,R\,\cos\theta\,\cos(\phi-\gamma_q)} = \sum_{m=-\infty}^{\infty} i^m\, J_m\left(\frac{2\pi}{\lambda}\, R\,\cos\theta\right) e^{-i\,m\,(\phi-\gamma_q)} \tag{6.19}$$

Now substituting (6.19) in (6.17) gives

$$\begin{aligned} x(t) &= \tfrac{A}{n}\, e^{-iw_c t} \textstyle\sum_{j=0}^{n-1} \sum_{m=-\infty}^{\infty} i^m\, J_m\left(\tfrac{2\pi}{\lambda}\, R\,\cos\theta\right) e^{im\gamma_j}\, e^{im\phi} \\ &= \tfrac{A}{n}\, e^{-iw_c t} \textstyle\sum_{m=-\infty}^{\infty} i^m\, J_m\left(\tfrac{2\pi}{\lambda}\, R\,\cos\theta\right) e^{im[\gamma_0+\gamma_1+\cdots+\gamma_{n-1}]}\, e^{im\phi} \end{aligned} \tag{6.20}$$

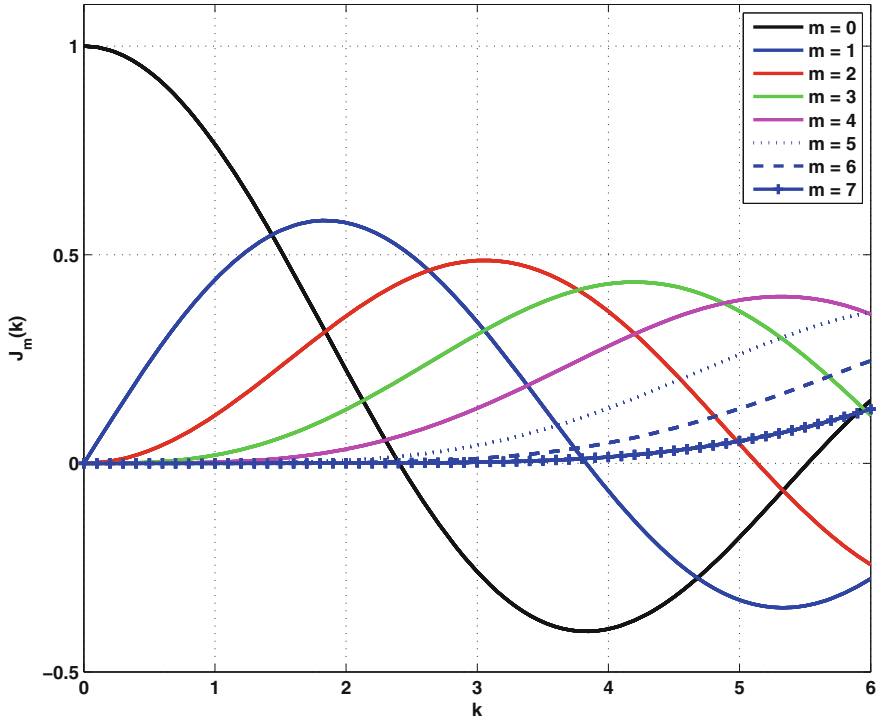


Fig. 6.14 Bessel functions of 1st kind

It is easy to see that $\left[\gamma_0 + \gamma_1 + \cdots + \gamma_{n-1}\right] = (n-1)\pi$, since $\gamma_j = j\,\frac{2\pi}{n}$, hence $e^{im[\gamma_0+\gamma_1+\cdots+\gamma_{n-1}]} = e^{i\,m\,(n-1)\,\pi} = 1$. Then (6.20) can be written as

$$x(t) = \frac{A}{n}\, e^{-i\,w_c t} \sum_{m=-\infty}^{\infty} i^m\, J_m\left(\frac{2\pi}{\lambda}\, R\, \cos\theta\right) e^{-im\phi} \tag{6.21}$$

Each term in (6.21) is called a phase mode excitation of the array [54, 92]. In order to determine how many modes can be excited for an array of radius R, we observe the behavior of Bessel functions in the visible region. Figure 6.14 shows the Bessel functions $J_m(k)$ of 1st kind versus $k = \frac{2\pi}{\lambda}\, R\, \cos\theta$ with $\frac{R}{\lambda} = 1$. The visible region is $0 < k < 2\pi$. Notice that in Fig. 6.14, J_7 has a slight positive value toward the end of the visible region, thus the number of modes can be approximated as

$$M \approx 2\pi R_\lambda \tag{6.22}$$

where $R_\lambda = \frac{R}{\lambda}$. Due to the symmetric nature of the Bessel functions, the number of modes that can be used are $2M + 1$. The sampling theorem that indicates the number of microphones needed to excite M modes is given by

$$n \geq 2\left(2\pi \frac{R}{\lambda}\right) + 1 \tag{6.23}$$

which implies that the spacing between two adjacent microphones in a uniform circular array should be

$$d_{cir} \leq \frac{\lambda}{2}. \tag{6.24}$$

Chapter 7
Bearing Estimation Using Acoustic Arrays

Whenever we hear sound, we instinctively turn our head towards the direction of the sound. Our brain is able to process the sound heard by both the ears and determine the direction with respect to the orientation of the head and turn towards the source producing the sound. In a battlefield, there may be explosions caused by the launch and detonation of artillery, rockets, guns, and grenades; or sounds due to ground-based vehicles such as tanks, humvee and civilian vehicles (cars, vans, trucks traveling on the road) or airborne vehicles such as helicopters, airplanes, and unattended aerial vehicles (UAVs) flying. The sound generated by these sources can be captured using the acoustic sensors/microphones. If these microphones are arranged in an array, their data can be processed to estimate the bearing angles of the targets.

This chapter presents some of the techniques used in determining the direction of arrival (DOA) angle or bearing angle of the sound waves impinging on the microphones in an array. Once we know the bearing angles from multiple arrays, it is easy to triangulate and determine the location of the source. Source localization is one of the fundamental aspects of battlefield acoustics. So, estimation of bearing angles is important. Bearing estimation has been a subject of research for several decades and it is still being studied by some. As a result of intensive study, several novel techniques have been developed using super resolution techniques, which we discuss in this chapter. We first present bearing estimation using the time of arrival of acoustic events.

7.1 Bearing Estimation Using Time of Arrival Information

Humans process sound based on the time of arrival of the sound at each ear. For example, if the sound source is located to the left side, the sound will be heard first by the left ear and then by the right ear. The delay appears due to the additional time (propagation time) required by the sound to travel the extra distance. This information is processed by the brain, which determines the direction of the sound source. We now analyze this phenomenon and derive the mathematical basis for estimating the

T. Damarla, *Battlefield Acoustics*, DOI 10.1007/978-3-319-16036-8_7

Fig. 7.1 Tetrahedral acoustic sensor array

bearing angles based on the time of arrival of sound at a pair of microphones. We assume, that we are able to estimate the time of arrival of the acoustic (sound) signal very accurately for the time being. In general, signals from acoustic sources can be classified as (a) transient and (b) continuous. Transient events are caused by explosions or events such as slamming of a door, etc., that cause a sudden burst of air movement. Continuous signals are due to ground or airborne vehicles. The time of arrival measurements to determine the bearing angles are particularly suited for transient events. For continuous signals, it is more appropriate to use delay sum or eigenvector projection based techniques to estimate the bearing angles; which are presented in the subsequent sections of this chapter.

Let us consider an array of microphones such as the one shown in Fig. 7.1 with its geometric representation shown in Fig. 7.2. Let the microphone coordinates be denoted by $p_i, i \in \{1, \ldots, n\}$, where n denotes the number of microphones in the array. The input to the array is a plane wave in the direction **a** where **a** can be expressed as (see Fig. 7.3)

$$\mathbf{a}(\theta,\ \phi) = [\cos\theta\ \cos\phi,\ \cos\theta\ \sin\phi,\ \sin\theta]^T\ , \tag{7.1}$$

where ϕ denotes the azimuth, θ denotes the elevation, and 'T' denotes the transpose. Let $\{\tau_1, \tau_2, \ldots, \tau_n\}$ denote the time of arrival of the acoustic signal at the microphones. Consider Fig. 7.4, which shows two microphones located at p_i and p_j and a plane wave impinging on them. Clearly, the plane wave reaches the microphone at

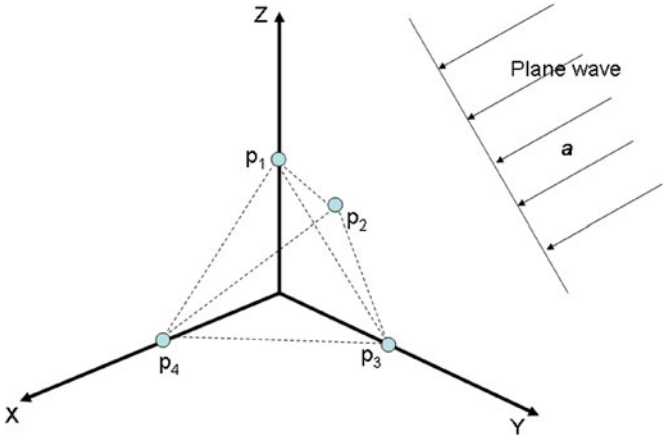


Fig. 7.2 Acoustic sensor array with plane wave input

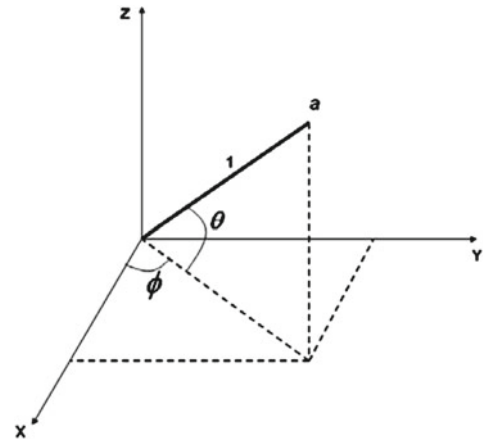


Fig. 7.3 Pointing Vector

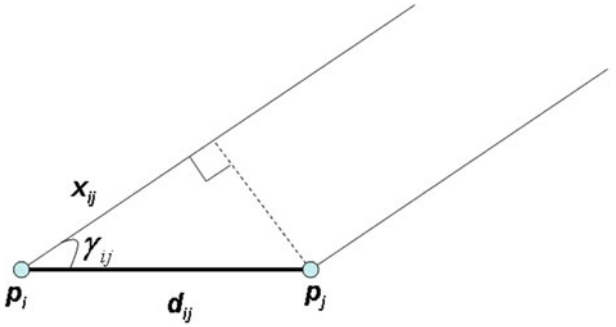


Fig. 7.4 Geometry of plane wave impinging on two microphones

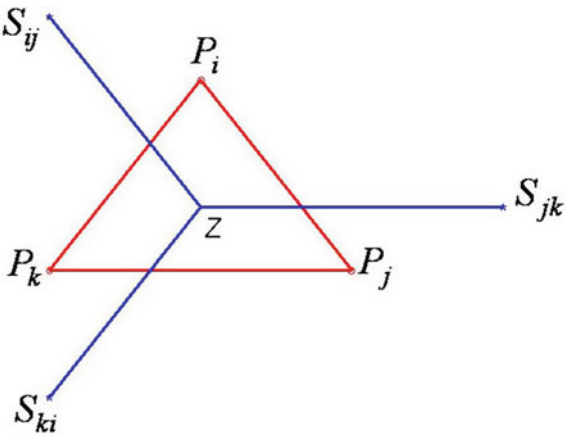


Fig. 7.5 Conversion of P_{ij} to S_{ij}

location p_j and then after traveling an additional distance x_{ij} it reaches the microphone at location p_i. The time it took the plane wave to travel the extra distance x_{ij} is $\tau_{ij} = \tau_i - \tau_j$ which is equal to x_{ij}/c, where c is the propagation velocity of sound. Then from Fig. 7.4, we find the angle γ_{ij} the pointing vector makes with the line joining the two microphones located at p_i and p_j

$$\cos\gamma_{ij} = \frac{x_{ij}}{d_{ij}} = \frac{\tau_{ij}}{t_{ij}} \tag{7.2}$$

where $t_{ij} = \frac{d_{ij}}{c}$ is the time taken by the sound to travel the distance d_{ij}. Clearly, the angle of arrival (AoA) γ_{ij} defines a cone around the axis joining the two microphones, as shown in Fig. 7.6a. The equation of the cone is given by

$$\frac{(p_i - p_j)^T \bullet U}{\|p_i - p_j\|} = \cos\gamma_{ij},$$

where $U = \mathbf{a}$ is the unit vector in the direction of the target given by (7.1). Now, each AoA for each pair of microphones generates a cone around their respective axes and they intersect. Figure 7.6b shows the intersection of three cones. These cones intersect along a line that points to the target, that is, the line of intersection gives the pointing vector U to the target.

Let $Z = [0\ 0\ 0]^T$ be the center of the array, and $S_{i,j}$ is a point in the space such that the vector $S_{i,j} - Z$ is parallel to the vector $p_i - p_j$ and the magnitude of the vector $\|S_{i,j} - Z\|$ equals to $\|p_i - p_j\|$. For a small array and large target distances, the AoA of the target with respect to the vector $S_{i,j} - Z$ would be same as the AoA with respect to the vector $p_i - p_j$. Figure 7.5 shows this transformation of coordinates. Then the equation of the cone with respect to the axis $(S_{i,j} - Z)$ is given by

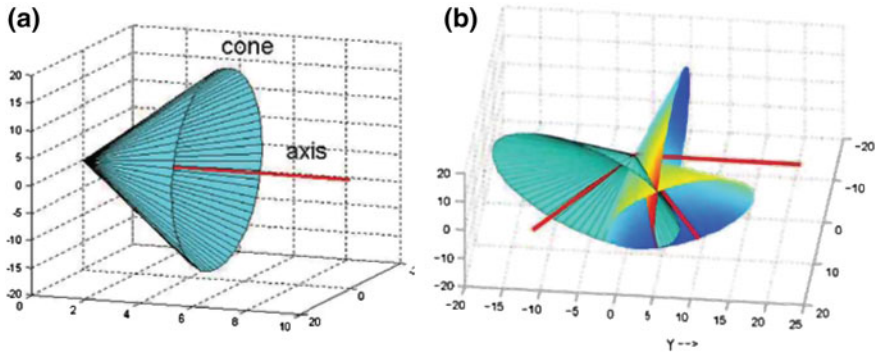


Fig. 7.6 **a** Single cone **b** intersection of cones

$$\frac{\left(S_{i,j} - Z\right)^T \bullet U}{\|S_{i,j} - Z\|} = \cos\gamma_{i,j} \ . \tag{7.3}$$

Then solving for the pointing vector U implies solving the linear equation:

$$\begin{bmatrix} \left(S_{1,2} - Z\right)^T \\ \left(S_{1,3} - Z\right)^T \\ \vdots \\ \left(S_{n-1,n} - Z\right)^T \end{bmatrix} U = \begin{bmatrix} \left\|S_{1,2} - Z\right\| \cos\gamma_{1,2} \\ \left\|S_{1,3} - Z\right\| \cos\gamma_{1,3} \\ \vdots \\ \left\|S_{n-1,n} - Z\right\| \cos\gamma_{n-1,n} \end{bmatrix} \tag{7.4}$$

or substituting (7.2) in (7.4)

$$\begin{bmatrix} \left(S_{1,2} - Z\right)^T \\ \left(S_{1,3} - Z\right)^T \\ \vdots \\ \left(S_{n-1,n} - Z\right)^T \end{bmatrix} U = \begin{bmatrix} \left\|S_{1,2} - Z\right\| \frac{\tau_{1,2}}{t_{1,2}} \\ \left\|S_{1,3} - Z\right\| \frac{\tau_{1,3}}{t_{1,3}} \\ \vdots \\ \left\|S_{n-1,n} - Z\right\| \frac{\tau_{n-1,n}}{t_{n-1,n}} \end{bmatrix} \tag{7.5}$$

From U, one can easily determine both the azimuth ϕ and elevation θ angles using (7.1). The algorithm developed here involves solving (7.4) or (7.5) for U using the ground truth of the sensor positions p_i, $\forall i$, and the estimation of AoA $\gamma_{i,j}$ or the time difference of arrival $\tau_{i,j}$, $\forall i \neq j$.

In this section, we presented the mathematical basis for estimating the bearing angles, both azimuth and elevation, using the time difference of arrival between pairs of microphones. In the next section, we present techniques on how to suppress the signals coming in from a particular direction and optimize the signals from a desired direction. These techniques are particularly useful in finding the AoA of targets distributed in multiple directions.

7.2 Adaptive Beam Forming

Adaptive beam forming is an approach where the SNR is optimized in the desired direction and steering nulls in the direction of the interfering sources. In general, the SNR depends on the size of the array, that is, the larger the array size the greater the SNR enhancement in the direction of look angle provided the space between the elements of the array is less than half-wavelength. When there are more elements in an array, it has the wider temporal frequency bandwidth where wavenumbers can be uniquely specified without spatial aliasing. However, on several occasions, multiple sources can emit the same frequency. For example, multiple echoes from nearby reflectors may impinge on the array.

Wavenumber is a spatial representation of a propagating sinusoidal wave. It is represented as

$$k = \frac{\omega}{c}$$

where $\omega = 2\pi f$, f is the frequency of the sinusoidal wave, ω is the radial frequency and c is the propagation velocity of the wave in the medium in m/sec. Since wavelength $\lambda = \frac{c}{f}$, the wavenumber can be written as $k = \frac{2\pi}{\lambda}$. Wavenumber sensor systems are typically are sensor arrays used to filter and detect waves from a particular direction (bearing). In this section, we present several techniques for estimating the DOA of the wave. A typical wavenumber sensor system everybody familiar is the ultrasound scanner used for medical diagnostic purposes.

The process used in estimating the DOA is adaptive beamforming to optimize the SNR in the look direction while carefully placing nulls in the direction of the interfering sources. The beamwidth of an array at any given frequency is controlled by the number of array elements. The higher the number of array elements, the smaller the beamwidth (see Fig. 6.7) and hence the resolution capability of the array in resolving closely spaced sources is better when the array elements are spaced less than half-wavelength (see Fig. 6.8).

Two different adaptive beamforming techniques are presented in the following sections. One is based on the block least squares filtering technique and another is based on the projection based least squared error filtering technique. In the block least squares filtering technique, a spatial whitening filter is created to predict the signal output of one of the sensors in the array using the weighted sum of the signals from the other array sensors. The wavenumber response of the filter for the array will have nulls in the directions of arrival of any spatially coherent sources. This technique is presented in Sect. 7.2.1.

7.2.1 DOA Estimation Using Array Null-Forming

Suppose there are multiple sources that are independently (that is, the noise of the individual source is independent) emitting signals, then it is possible to steer the

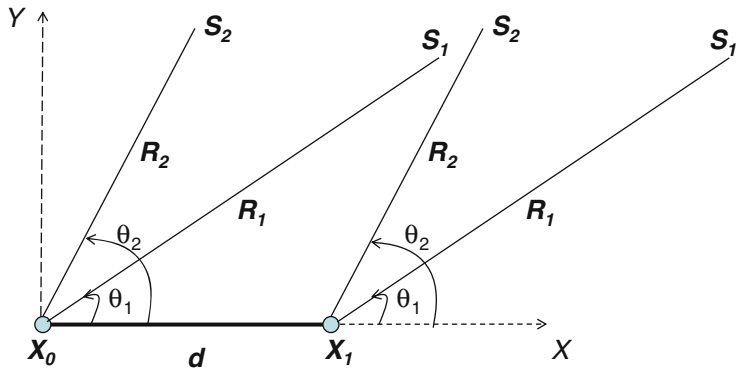


Fig. 7.7 Linear array of two elements receiving signals from S_1 and S_2

"null" output of the array, or the direction of the zero-output response, in the direction of one of the emitting sources. Consider two sources spatially distributed and located at angles θ_1 and θ_2 with respect to the x-axis measured counterclockwise, and emitting the same frequency, as shown in Fig. 7.7. Assuming that the signal amplitudes transmitted by the sources are A_1 and A_2 and the phases ϕ_1 and ϕ_2, then the output of the sensor X_0 located at the origin is given by

$$s_0 = A_1 e^{j(\omega t+\phi_1)} + A_2 e^{j(\omega t+\phi_2)} \tag{7.6}$$

Let the distance between the two sensors be d, then the output at the sensor X_1 is

$$s_1 = A_1 e^{j(\omega t+\phi_1+kd\cos\theta_1)} + A_2 e^{j(\omega t+\phi_2+kd\cos\theta_2)} \tag{7.7}$$

where k is the wavenumber. Now, we show that the cross correlation between two signals s_0 and s_1 has a Dirac delta function structure. The spatial cross correlation is defined as

$$\begin{aligned} R_{-d} &= E\left[s_0 s_1^*\right] \\ &= E\left[\left(A_1 e^{j(\omega t+\phi_1)} + A_2 e^{j(\omega t+\phi_2)}\right)\right. \\ &\quad \left.\left(A_1 e^{-j(\omega t+\phi_1+kd\cos\theta_1)} + A_2 e^{-j(\omega t+\phi_2+kd\cos\theta_2)}\right)\right] \end{aligned} \tag{7.8}$$

or

$$\begin{aligned} R_{-d} = E\left[\left(A_1^2 + A_2 A_1 e^{-j(\phi_1-\phi_2)}\right) e^{-jkd\cos\theta_1} + \right. \\ \left.\left(A_2^2 + A_1 A_2 e^{-j(\phi_1-\phi_2)}\right) e^{-jkd\cos\theta_2}\right] \end{aligned} \tag{7.9}$$

Notice that the cross correlation is dependent only on the magnitudes of the signals and the phases of the two sources. Hence, it cannot be estimated unless the amplitudes

and phases are known. However, since the sources are statistically independent, the phase difference between the two will be a random angle between $\pm\pi$. Then the cross correlation becomes

$$R_{-d} = A_1^2\, e^{-jkd\cos\theta_1} + A_2^2\, e^{-jkd\cos\theta_2} \tag{7.10}$$

since the expected value of $e^{-j(\phi_1-\phi_2)}$ is zero. The structure given by (7.10) is a Dirac delta function with a delta function at $d\cos\theta_1$ and $d\cos\theta_2$. From this, it is clear that the the sources can be resolved by either linear prediction or an eigen value approach so long as there are more sensors than sources.

In order to be able to achieve the resolution of multiple sources, it is important that the sources be independent. However, if only one snapshot of the signals is collected by the sensors, it is difficult to achieve source independence. Often one has to perform some manipulation in order to get the independence. If there are several sensors in an array, then the array data can be divided into sub-arrays and then one can calculate the ensemble average spatially of each sub-array's cross correlation to get an overall cross correlation with the dependencies suppressed. This technique is also called the spatial smoothing method for dealing with coherent signals.

Now the DOA estimation using null-forming is presented. Assume that there are 'M' number of sensor elements in a linear array as shown in Fig. 7.8. In this figure,

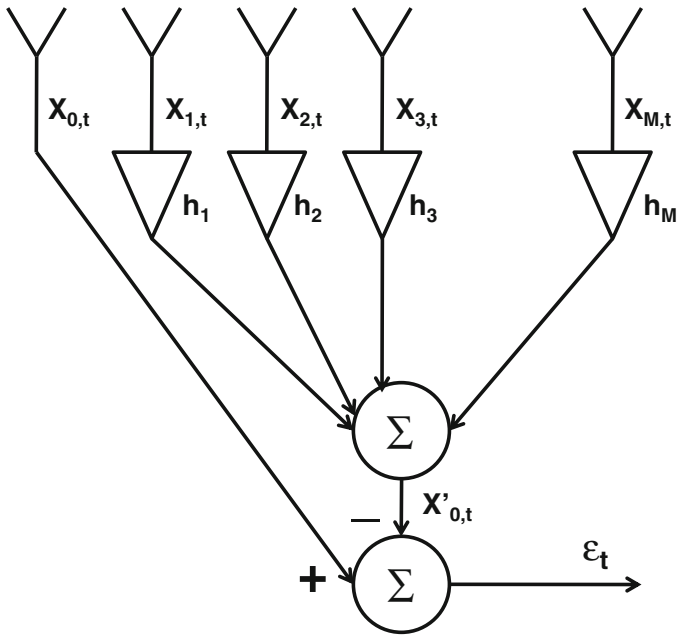


Fig. 7.8 Linear predictive filter for determining the wavenumbers corresponding to the arrival angles of spatially incoherent source signals

the response of the sensor X_0 is predicted as the weighted sum of the remaining sensor $X_1, \ldots, X_M$. The error, the difference between the actual output $X_{0,t}$ and the predicted output $X'_{0,t}$ at the time stamp t, is denoted by ε_t. Then the error over N time stamps is given as

$$\begin{bmatrix} \varepsilon_t \\ \varepsilon_{t-1} \\ \vdots \\ \varepsilon_{t-N+1} \end{bmatrix} = \begin{bmatrix} x_{0,t} \\ x_{0,t-1} \\ \vdots \\ x_{0,t-N+1} \end{bmatrix} - \begin{bmatrix} x_{1,t} & x_{2,t} & \cdots & x_{M,t} \\ x_{1,t-1} & x_{2,t-1} & \cdots & x_{M,t-1} \\ \vdots & \vdots & \cdots & \vdots \\ x_{1,t-N+1} & x_{2,t-N+1} & \cdots & x_{M,t-N+1} \end{bmatrix} \begin{bmatrix} h_1 \\ h_2 \\ \vdots \\ h_M \end{bmatrix} \tag{7.11}$$

or

$$\bar{\varepsilon} = \bar{x}_0 - \bar{X}\mathcal{H} \tag{7.12}$$

Our goal is to minimize the error ε by properly selecting the weights $\mathcal{H}$. This can be done by solving for $\mathcal{H}$. Now consider the sum of the squared error

$$\begin{aligned} \bar{\varepsilon}^H \bar{\varepsilon} &= \left(\bar{x} - \bar{X}\mathcal{H}\right)^H \left(\bar{x} - \bar{X}\mathcal{H}\right) \\ &= \bar{x}^H \bar{x} - \bar{x}^H \bar{X}\mathcal{H} - \mathcal{H}^H \bar{X}^H \bar{x} + \mathcal{H}^H \bar{X}^H \bar{X}\mathcal{H} \end{aligned} \tag{7.13}$$

where superscript H is the Hermitian transpose (transpose and complex conjugate). Hermitian transpose is applied if the elements of $\bar{x}$ and $\bar{X}$ are complex. The above equation is quadratic in nature with respect to the coefficients of $\mathcal{H}$, that is, the error surface is bowl-shaped and has only one extreme where the slope of the surface is zero. The error ε is minimum for the values of $\mathcal{H}$ where the slope of the error surface is zero provided that the second derivative with respect to $\mathcal{H}$ is positive definite, indicating that the error surface is concave up. The zero slope is found by computing the partial derivatives of (7.13) with respect to $\mathcal{H}$ while treating $\mathcal{H}^H$ as a constant and similarly finding the partial derivative with respect to $\mathcal{H}^H$ while treating $\mathcal{H}$ as a constant and adding both.

$$\begin{aligned} \frac{\partial \bar{\varepsilon}^H \bar{\varepsilon}}{\partial \mathcal{H}} + \left\{\frac{\partial \bar{\varepsilon}^H \bar{\varepsilon}}{\partial \mathcal{H}^H}\right\}^H &= -\bar{x}^H \bar{X} + \mathcal{H}^H \bar{X}^H \bar{X} + \left\{-\bar{X}^H \bar{x} + \bar{X}^H \bar{X}\mathcal{H}\right\}^H \\ &= -2\bar{x}^H \bar{X} + 2\mathcal{H}^H \bar{X}^H \bar{X} = 0 \end{aligned} \tag{7.14}$$

The solution for the value of $\mathcal{H}^H$, which gives the zero error surface slope in (7.14) is

$$\mathcal{H}^H = \bar{x}^H \bar{X} \left(\bar{X}^H \bar{X}\right)^{-1}. \tag{7.15}$$

The Hermitian transpose of (7.15) gives the value of $\mathcal{H}$ for the zero slope

$$\mathcal{H} = \left(\bar{X}^H \bar{X}\right)^{-1} \bar{X}^H \bar{x}. \tag{7.16}$$

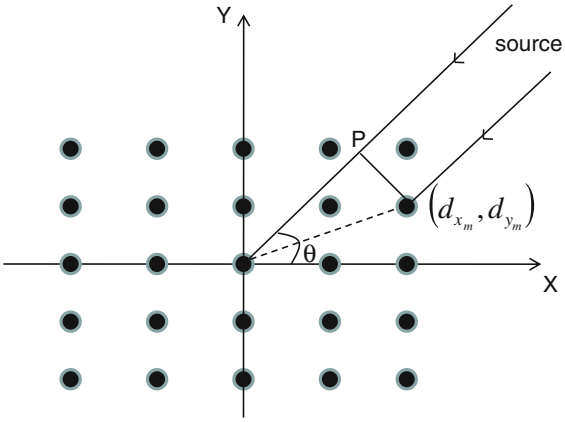


Fig. 7.9 2D-array of microphones

Now the solution for $\mathcal{H}$ is found, then ε_t in (7.11) can be written as

$$\varepsilon_t = x_{0,t} - \sum_{m=1}^{M} h_m x_{m,t} \tag{7.17}$$

or

$$\varepsilon_t = \sum_{m=0}^{M} A_m x_{m,t}, \quad A_0 = 1, \; A_m = -h_m. \tag{7.18}$$

Note that ε_t is the output of the whitening filter shown in Fig. 7.8. From the structure given by (7.18), it is possible to evaluate the whitening filter as a function of the complex variable $z = e^{-jkd_{l_m}\cos\theta}$, where d_{l_m} is the x-coordinate of the mth sensor in line array, k is the wavenumber and θ is the direction of interest. For linear array, the wavenumber scales with the cosine of the angle θ and hence (7.18) can be written as

$$D(\theta) = \sum_{m=0}^{M} A_m e^{-jkd_{l_m}\cos\theta} \tag{7.19}$$

Consider the two-dimensional array shown in Fig. 7.9 with several microphones. Let the angle at which the source be oriented with respect to the x-axis at the origin is θ. Then consider the mth microphone located at the coordinates (d_{x_m}, d_{y_m}), the acoustic wave first reaches the microphone at the location (d_{x_m}, d_{y_m}) and then the additional distance the acoustic wave needs to travel to the reference microphone at the origin is given by the projection of the vector $d_m = \begin{bmatrix} d_{x_m} & d_{y_m} \end{bmatrix}$ on to the wave traveling to the origin whose unit vector is given by $U = [\cos\theta \;\; \sin\theta]$. The projection is nothing but the dot product of the two vectors

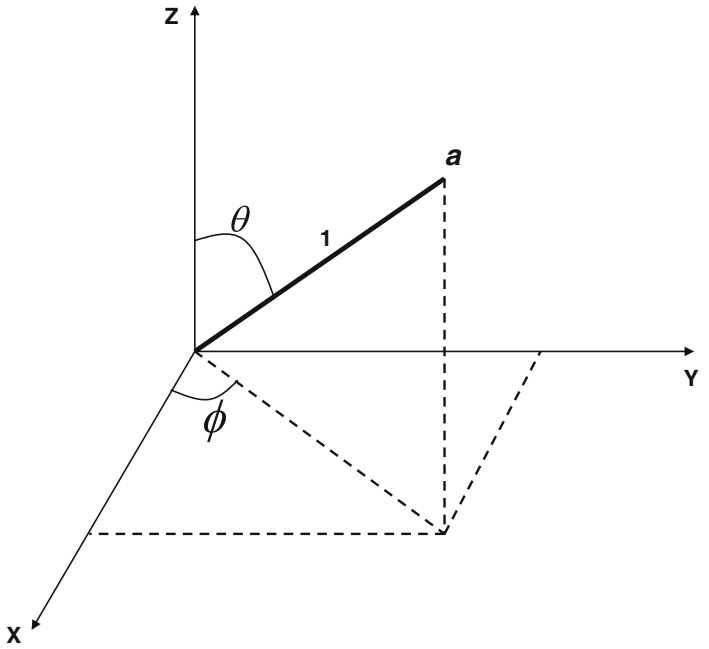


Fig. 7.10 3D unit vector

$$P = d_m \bullet U = d_{x_m} \cos\theta + d_{y_m} \sin\theta$$

Then the directional response of the spatial whitening filter is given by

$$D(\theta) = \sum_{m=0}^{M} A_m e^{-jkd_{x_m} \cos\theta} \; e^{-jkd_{y_m} \sin\theta} \tag{7.20}$$

For the case where the microphone array is three dimensional, then the unit vector corresponding to the wave impinging at the origin at an azimuth angle ϕ and elevation angle θ as shown in Fig. 7.10 is given by

$$U = [\sin\theta \cos\phi \quad \sin\theta \sin\phi \quad \cos\theta] \tag{7.21}$$

Then the directional response of the spatial whitening filter for a 3D array is given by

$$D(\phi, \theta) = \sum_{m=0}^{M} A_m e^{-jkd_{x_m} \sin\theta \cos\phi} \; e^{-jkd_{y_m} \sin\theta \sin\phi} \; e^{-jkd_{z_m} \cos\theta} \tag{7.22}$$

where the coordinates of the mth microphone are $\left(d_{x_m}, d_{y_m}, d_{z_m}\right)$.

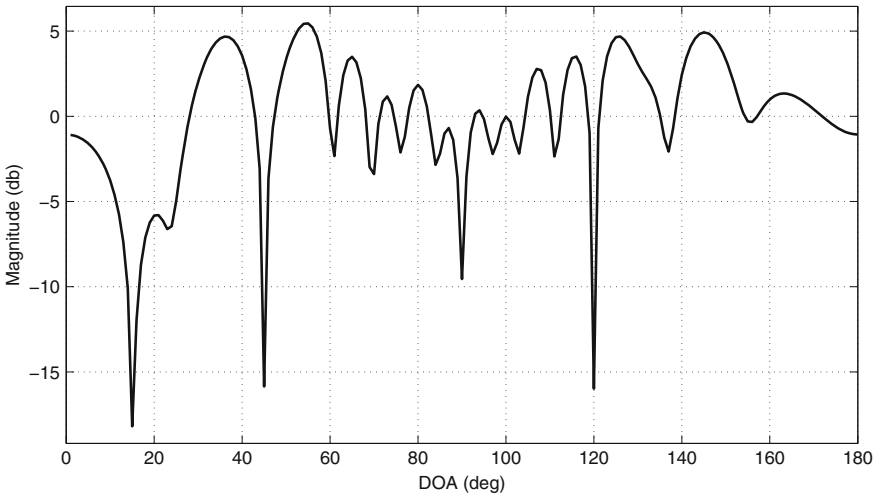


Fig. 7.11 Output of a whitening filter for three 100-Hz sources at bearing angles 15°, 45° and 120°

The DOA estimation using null forming is done by first generating the matrices in (7.11) from the given snapshots of the data impinging on the array and then solving for the weight matrix $\mathcal{H}$ using (7.16) and then evaluating $D(\theta)$ in (7.19) for angle $\theta \in \{0, 1, \ldots, 180\}$. Figure 7.11 shows the result for three sources at angle 15°, 45° and 120° and emitting 100 Hz using a linear array with 17 microphones. The number of snapshots used are 25.

A simulation example illustrating the workings of the DOA estimation using null-forming is presented below.

***Example*: *DOA estimation using null-forming*:**

In this example, a linear array of 11 microphones is considered. Three sources located far from the array at bearing angles of 15°, 45° and 120° are emitting 100-Hz plane waves. For designing the whitening filter, 11 snap shots are used. A MATLAB program to estimate the DOA angles using null-forming is presented at the end of the chapter.

In the next section, DOA estimation using an eigenvector projection approach is presented.

7.2.2 Eigenvector Projection Based DOA Estimation

The sound waves generated by the targets propagate through air and across the sensor array, as shown in Fig. 7.12. Each wave can be expressed as an eigenvector with a particular magnitude called the eigenvalue. There are other sounds such as noise caused by natural phenomena, winds blowing, turbulence, etc., also waft over the array. So the entire set of eigenvectors can be thought of the eigen space and it is partitioned into a "signal plus noise" and a "noise" space. Note that the first space

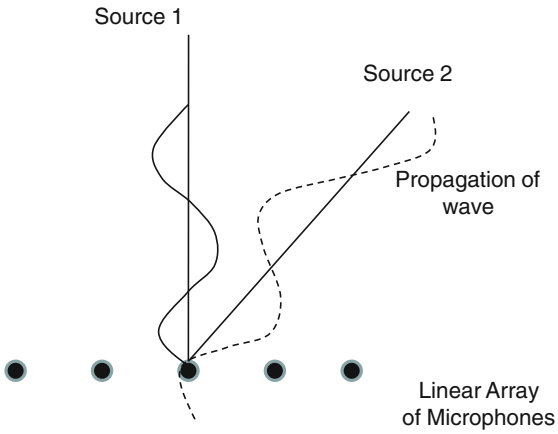


Fig. 7.12 Plane waves travelling over the array

is called signal plus noise space since most of the signals are corrupted by noise as they propagate through the medium.

One of the main advantages of the eigen approach is that the eigenvectors are orthonormal, that is, the dot product of two different eigenvectors is zero and the dot product of an eigenvector with itself is one. This implies that the spatial response of the eigen vector has a main lobe in the direction of its source and zeros in the direction of other sources. Due to the presence of noise, the eigenvector response may not be ideal in the direction of its source. However, the noise eigenvector will have its nulls in the direction of the sources with a random look direction. So if one sums the response of all the noise eigenvectors, the beam response will be random everywhere with null in the real source directions just as in the null forming spatial whitening filter. This technique is called Multiple Signal Classification (MUSIC).

In order to generate the eigenvectors, first the covariance matrix of the data is constructed. However, if there are multiple sources emitting the same frequency, to achieve independence of the sources one of two approaches, namely, (a) time averaging or (b) spatial averaging is done.

***Time Averaging*:**

In this approach, a number of time snapshots are collected just as in the case of a null forming spatial whitening filter. Each time snapshot is used to generate a covariance matrix. Then the average of all the covariance matrices is used to extract the eigenvectors and eigenvalues of the covariance matrix. Let $X = \left\{x_{1,t}, x_{2,t}, \ldots, x_{M,t}\right\}$, $t = \{1, 2, \ldots, N\}$ is the output of a linear array of M microphones. Then the covariance of the array signals is given by

$$R = \begin{bmatrix} R_{1,1} & R_{1,2} & R_{1,3} & \cdots & R_{1,M} \\ R_{2,1} & R_{2,2} & R_{2,3} & \cdots & R_{2,M} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ R_{M,1} & R_{M,2} & R_{M,3} & \cdots & R_{M,M} \end{bmatrix}; \quad R_{i,j} = E\left[(X_i - \mu_i)^* \left(X_j - \mu_j\right)\right] \tag{7.23}$$

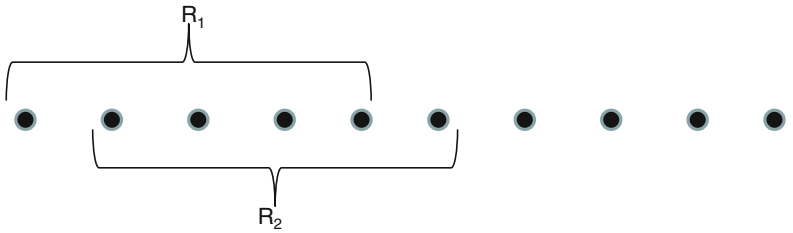


Fig. 7.13 Sub-arrays considered for spatial smoothing

where x^* is the conjugate of x, and μ_i is the mean of vector $X_i = \{x_{i,t}\}$, $\forall t \in \{1, \ldots, N\}$. Now, if there are L number of time snapshots, then the averaged covariance matrix is given by

$$R = \frac{1}{L} \sum_{i=1}^{L} R_i \tag{7.24}$$

***Spatial Averaging*:**

Spatial averaging can be used if multiple time snapshots are not available. If the number of sources are L, then use a sliding window of length $Q > L$ to construct the covariance matrix of Q sensors (microphones) at a time, as shown in Fig. 7.13, and then find their average [15, 46].

Now that the covariance matrix is constructed, it should be decomposed into eigenvectors and eigenvalues. The general eigenvalue problem is defined by the following equation:

$$R\nu_i = \nu_i \lambda_i, \quad i = 1, 2, \ldots, M \tag{7.25}$$

where ν_i and λ_i are the eigenvector and eigenvalues, respectively. The eigenvalues λ_i can be written as diagonal elements of a matrix Λ with zeros everywhere else as shown below:

$$\Lambda = \begin{bmatrix} \lambda_1 & 0 & 0 & \cdots & 0 \\ 0 & \lambda_2 & 0 & \cdots & 0 \\ 0 & 0 & \lambda_3 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & \lambda_M \end{bmatrix} \tag{7.26}$$

Then (7.25) can be written as

$$RV = V\Lambda \tag{7.27}$$

where

$$V = [\nu_1 \ \nu_2 \ \nu_3 \ \cdots \ \nu_M] \tag{7.28}$$

Then

$$V^H R V = \Lambda. \tag{7.29}$$

There are several well-known algorithms that generate the eigenvectors and eigenvalues, which can be found in a linear algebra book [3]. If the diagonal elements of Λ are not in descending order, they can be arranged. Assuming the diagonal elements are in descending order, the first L eigenvectors and their eigenvalues correspond to the "signal plus noise" space and the rest of the eigenvectors correspond to the "noise space," provided the number of independent sources are L. Typically, the eigenvalues corresponding to the signal have much larger values than those corresponding to the noise. If the number of sources is not known, one can estimate [96] them using several measures such as the Akaike Information Criterion (AIC) and Minimum Description Length (MDL).

7.2.2.1 MUSIC Algorithm for DOA Estimation Using Noise Eigenvector Projection

We now present the MUSIC algorithm for estimating the angle of arrival:
MUSIC algorithm

- Generate the covariance matrix $[R]$ using either time average or spatial averaging.
- Factor $[R]$ in to a product of eigenvectors and eigenvalues.
- Determine the eigenvectors that belong to the "signal plus noise" and "noise" space depending on the eigenvalues. The eigenvectors corresponding to the large eigenvalues belong to the signal plus noise, while the rest of the eigenvectors belong to the noise space. Let the noise eigenvectors be

$$\mathcal{V} = \left[\nu_{L+1} \ \nu_{L+2} \ \cdots \ \nu_M\right] \tag{7.30}$$

  where ν is a column vector, L is the number of radiating sources and M is the number of sensors in the linear array. Estimation of L is done by AIC or MDL.
- Generate the steering vector (also called the array manifestation)

$$\mathcal{S}(\theta) = \left[1 \ e^{-jkd\cos\theta} \ e^{-jk2d\cos\theta} \ \cdots \ e^{-jk(M-1)d\cos\theta}\right], \quad \forall\theta \in \{0, 1, \ldots, 180\} \tag{7.31}$$

  where k is the wavenumber and d is the spacing between the sensor elements.
- Now compute the response of the array

$$\mathcal{O}_{MUSIC}(\theta) = \mathcal{S}(\theta)\ \mathcal{V}\ \mathcal{V}^H\ \mathcal{S}^H(\theta), \ \forall\theta \in \{0, 1, \ldots, 180\}. \tag{7.32}$$

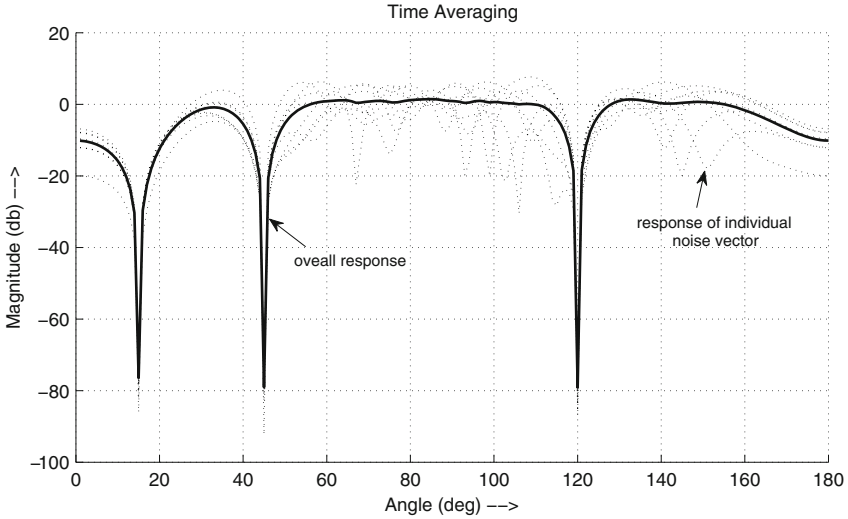


Fig. 7.14 Response of noise eigenvectors for a 11-element linear array; time average is done with 10 snapshots of the time-domain data

In order to illustrate the workings of the time and spatial averaging techniques, a linear array of $M = 11$ sensors (microphones) is considered. The number of independent sources are $L = 3$ located somewhere in the direction of $15°, 45°$ and $120°$ and radiating 100-Hz signals. Generation of a covariance matrix can be done using (a) time domain data or (b) frequency domain data if the signals are narrowband. Since the data collected by the sensors are in time domain, it can be converted in to frequency domain by taking a fast Fourier transform (FFT). In this example, both time and frequency domain data are used. For the time average case, ten snapshots are considered for generating ten covariance matrices and then averaging them to get the final covariance matrix. Figures 7.14 and 7.15 show the individual noise eigenvector response and the average response of all the noise eigenvectors of covariance matrices obtained using time averaging of the (a) time and (b) frequency domain data. Figures 7.16 and 7.17 show the responses for the spatial averaging case with 5 sensors considered at a time to generate the covariance matrices.

The Matlab code "doa_eigen_projection" used to simulate the signals and generate Figs. 7.16 and 7.17 is given at the end of the chapter.

7.2.2.2 Minimum Variance Distortionless Response Beamformer

Consider an M-element array of microphones as shown in Fig. 7.18. Its weighted output S is given by

$$\begin{aligned} S &= e^{-j\omega t}\left[w_1 + w_2 e^{-jkd\cos\theta} + \cdots + w_M e^{-jkd(M-1)\cos\theta}\right] \\ &= e^{j\omega t}\sum_{m=1}^{M} w_m e^{-jk(m-1)d\cos\theta}, \end{aligned} \tag{7.33}$$

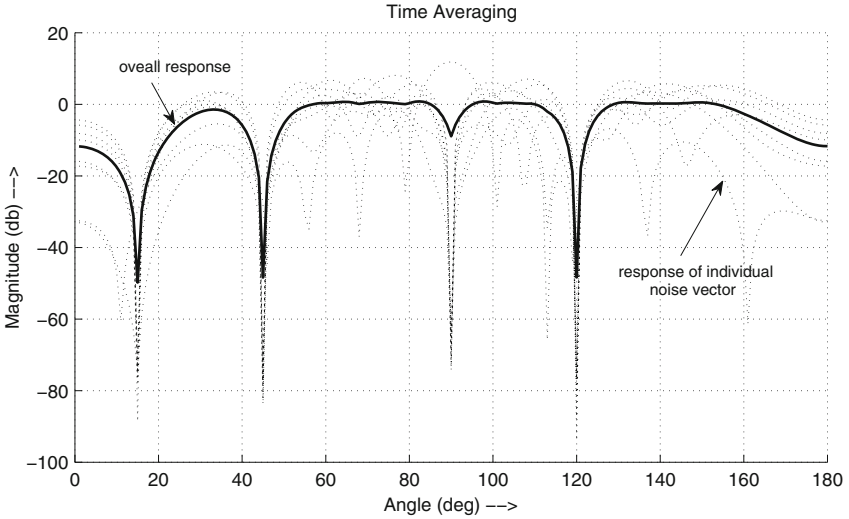


Fig. 7.15 Response of noise eigenvectors for a 11-element linear array; time average is done with 10 snapshots of the frequency-domain data

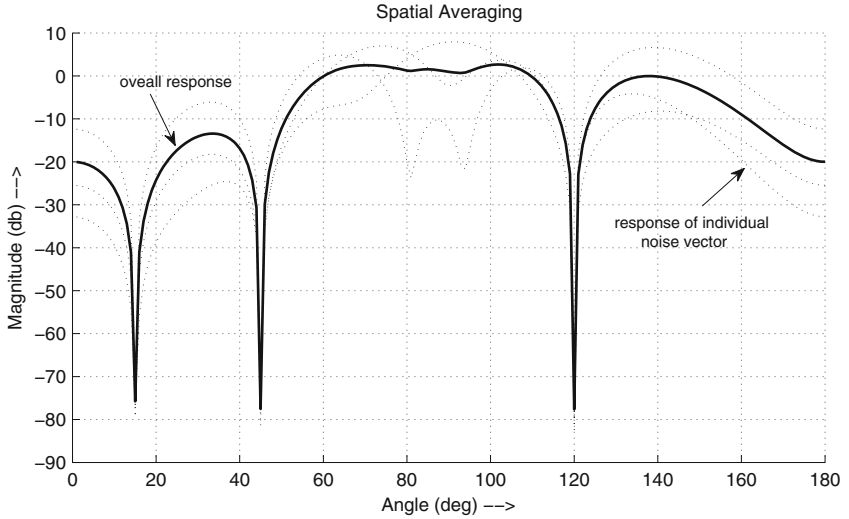


Fig. 7.16 Response of noise eigenvectors for a 11-element linear array; spatial averaging is done using 5 sensors and the covariance is computed using the time-domain data

where k is the wavenumber and d is the separation between the two array elements. In order to avoid the grating lobes the space between two adjacent array elements is $d \leq \frac{\lambda}{2}$. Equation (7.33) can be represented as a transversal filter, as shown in Fig. 7.19, where $u = e^{j\omega t}$ and $z = e^{jkd\cos\theta}$. In several filtering applications, it is

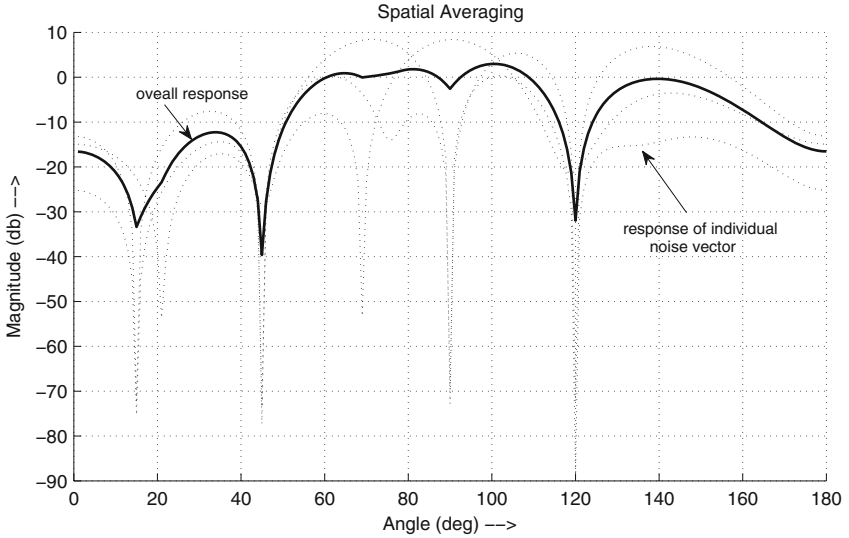


Fig. 7.17 Response of noise eigenvectors for a 11-element linear array; spatial averaging is done using 5 sensors and the covariance is computed using the frequency-domain data

necessary to design the filter to minimize the mean square value of the output subject to a specific constraint. One such constrained optimization problem is to find the optimum set of coefficients $w_1, w_2, \ldots, w_M$ such that

$$\sum_{m=1}^{M} w_m e^{-jk(m-1)d\cos\theta} = g \tag{7.34}$$

where is g is a complex valued gain. Denoting $\phi = kd\cos\theta$, the above constraint can be written as

$$\sum_{m=0}^{M-1} w_m \, e^{-jm\phi} = g. \tag{7.35}$$

In order to solve the constrained optimization problem, we use a method of Lagrange multipliers. Let us define a real-value cost function J that combines the two parts of the constrained optimization problem

$$J = \sum_{n=0}^{M-1} \sum_{m=0}^{M-1} w_n^{'} w_m \, e^{jn\phi} e^{-jm\phi} \; + \; Re\left[\lambda\left(w_m e^{-jm\phi} - g\right)\right] \tag{7.36}$$

where $w_n^{'}$ denotes the complex conjugate and λ is the Lagrange multiplier. The first part in (7.36) represents the output power and the second part represents the linear

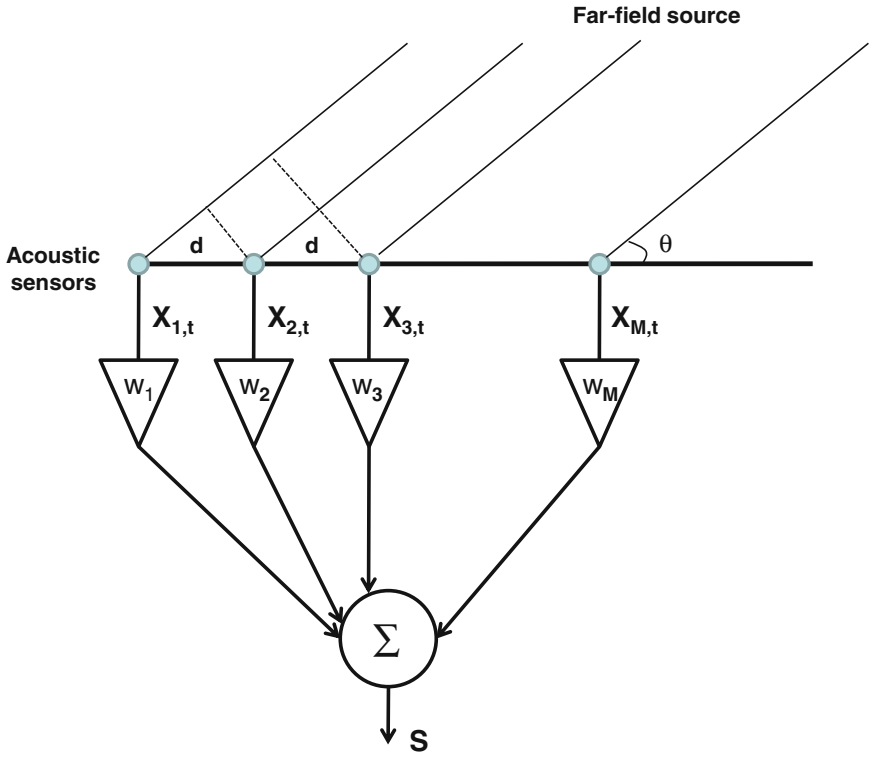


Fig. 7.18 Weighted sum of signals from an M-element array with a far-field source at angle θ

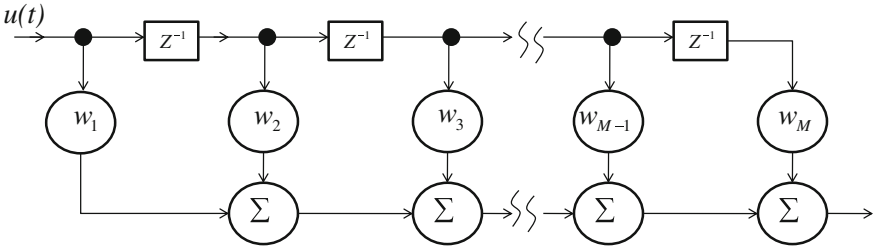


Fig. 7.19 Linear transversal filter

constraint. The optimal solution is obtained by taking the gradient of J and setting it equal to zero. The mth element of the gradient vector ∇J is

$$\nabla_m J = 2 \sum_{n=0}^{M-1} w_n \, e^{-j(n-m)\phi} + \lambda e^{-jm\phi} \tag{7.37}$$

Setting (7.37) to zero results in the optimal ith weight w_{oi}

$$\sum_{i=0}^{M-1} w_{oi}\ e^{-j(i-m)\phi} = -\frac{\lambda}{2}\ e^{-jm\phi} \tag{7.38}$$

Equation (7.38) represents M simultatneous equations and can be written in matrix notation as

$$RW_o = -\frac{\lambda}{2}\ S\left(\phi\right) \tag{7.39}$$

where R is the M-by-M correlation matrix and

$$S(\phi) = \left[1, e^{-j\phi}, \cdots, e^{-j(M-1)\phi}\right] \tag{7.40}$$

is the steering vector, where $\phi = kd\cos\theta$ and k is the wavenumber. Solving Eq. (7.39) for W_o results in

$$W_o = -\frac{\lambda}{2}\ R^{-1}S(\phi) \tag{7.41}$$

where R^{-1} is the inverse of the correlation matrix $R = E\left[XX^H\right]$ and X is the input to the sensor array elements. It is assumed that the inverse of R is nonsingular in (7.41). In order to avoid the singularity, the diagonal elements of R are added with small random numbers as shown in [61]. In order to estimate the Lagrange multiplier λ in (7.41), first the linear constraint (7.35) is rewritten as

$$W_o^H S(\phi) = g. \tag{7.42}$$

Taking the Hermitian transpose of both sides of (7.41) and multiplying by $S(\phi)$, we get

$$W_o^H S(\phi) = -\frac{\lambda}{2}\ S^H(\phi)R^{-1} \tag{7.43}$$

where we used the fact that $R^{-H} = R^{-1}$. Substituting (7.42) in (7.43), we get the Lagrange multiplier as

$$\lambda = -\frac{2g}{S^H(\phi)R^{-1}S(\phi)} \tag{7.44}$$

Substituting the value of λ in (7.41), we get the optimal weights as

$$W_o = \frac{gR^{-1}S(\phi)}{S^H(\phi)R^{-1}S(\phi)} \tag{7.45}$$

Selecting a particular value $\phi = \phi_0$ and minimizing the output power by finding the optimal weights with the linear constraint tends to attenuate the signals coming from different angles away from the angle corresponding to ϕ_0. The beamformer characterized by the weight W_o is referred to as a linearly constrained minimum variance (LCVM) beamformer. When $g = 1$, it is called the minimum variance distortionless response beamformer.

The output power given by (7.36) of the optimum beamformer is expressed as

$$J_{min} = W_o^H R W_o. \tag{7.46}$$

Substituting Eqs. (7.45) in (7.46) and simplifying gives the output power

$$J_{min} = \frac{1}{S^H(\phi_0) R^{-1} S(\phi_0)}. \tag{7.47}$$

The spectrum of the minimum variance distortionless beamformer as a function of ϕ is given by

$$S_{MVDR}(\phi) = \frac{1}{S^H(\phi) R^{-1} S(\phi)}. \tag{7.48}$$

As mentioned, the minimum variance beamformer attenuates the interference and noise not originated in the direction of the look angle ϕ.

An example of MVDR is presented with three sources located at bearing angles 15°, 45° and 120° radiating at 100-Hz is considered. An 11-element array is used for finding the bearing angles. The MVDR algorithm is developed using both time and frequency data. The MATLAB code is presented at the end of this chapter. The results of MVDR are shown in Figs. 7.20 and 7.21.

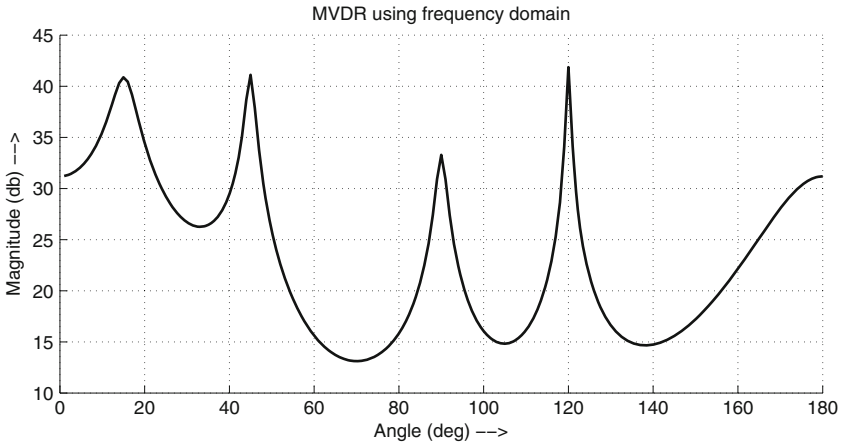


Fig. 7.20 MVDR output for three sources located at 15°, 45° and 120° using frequency-domain data

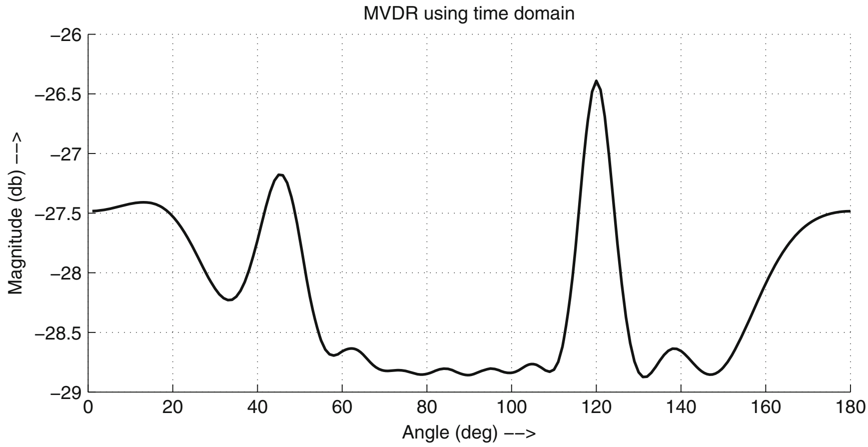


Fig. 7.21 MVDR output for three sources located at 15°, 45° and 120° using time-domain data

Here, we present the MATLAB code used for the examples presented above.

```
function null_forming3

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%   this function uses null forming techniques to estimate the bearing
%   angles
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

s_confg = [-5:1:5]'; % this is a linear array with spacing of 1 m
ns = numel(s_confg);
freq  = [100 100 100]; % This is the freq at which source is emitting the signal
spd = 343; % propagation velocity of sound
lambda = spd/freq(1);
half_lambda = lambda/2;
s_confg = s_confg.*half_lambda;
y = zeros(ns,1);
z = zeros(ns, 1);
s_confg = [s_confg, y, z];
fs = 8*1024; % sampling rate

% generate the signal
snap_shots = 11;
dat = zeros(snap_shots, fs, ns);
for i = 1:snap_shots
    S = gen_sig(s_confg, spd, freq, fs);
    S = S';
    S = flipud(S);
    dat(i, :,:) = S;
end
[n m] = size(S);

% we start estimating the output of the whitening filter output
wave_num = (2*pi)/lambda;
Y = zeros(180, 1);
D = zeros(1,180);

for k = 1:snap_shots,
    X = squeeze(dat(:,k,:));
    % now solve for the H coefficients
```

```
    H = solve_for_h(X);
    b = -1 * ones(1,ns);
    b(1) = 1;
    H = [1; H];
    b = b'.*H;
    % compute the output of the filter at each angle 1-180
    for j = 1: 180,
        theta = deg2rad(j);
        i = sqrt(-1);
        arr = exp(-i*wave_num*cos(theta)*s_confg(:,1));
        C = b.*arr;
        D(j) = D(j) + sum(C);
    end
end
x = 10*log10(abs(D));
plot(x), grid on
axis([0 180 min(x)-1 max(x)+1])

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% simulate the signals
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function S = gen_sig(s_confg, spd, freq, fs)

ang = [15 45 120];
n_ang = numel(ang);
omega1 = 2. * pi * freq;
n = 1:fs;
ts = 1/fs;
t1 = ts * n;
ns = size(s_confg, 1);
sigs = [];

for num = 1:n_ang,
    phi = ang(num)*(pi/180);
    x = cos(phi);
    y = sin(phi);
    z = 0;
    wave_vec = [x; y; z];
    td = s_confg*wave_vec;
    td = td./spd;
    switch num
        case 1
            omega = omega1(num);
        case 2
            omega = omega1(num);
        case 3
            omega = omega1(num);
    end
    k = rand(1,10);
    k = k(1)*pi;
    for n_mics = 1:ns,
        mic_delay = td(n_mics) + k;
        i = sqrt(-1);
        p1 = exp(i * omega * (t1 + mic_delay)); % signal generated
        x = rand(1,fs); % generated noise to add to the signal
        p1 = p1 + x.*0.24; % noise added
        sigs(num, n_mics,:) = p1;
    end
end
S = zeros(ns, fs);
for i = 1:n_ang,
    y = squeeze(sigs(i,:,:));
    S = S + y;
end
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function X = create_xmat

% this function creates the X matrix with several snap shots
[n, m] = size(s_confg);

blk_size = 15;
X = zeros(blk_size, n);
for i = 1:blk_size
    for j = 1:n,
        S = gen_sig(s_confg, spd, freq, fs);
        S = S’;
        X(i,:) = X(i,:) + S(i, :);
    end
end
X = X./n;
X = flipud(X);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function H = solve_for_h(Y)

X0 = Y(:,1);
X = Y(:,2:end);
H = (X’ * X)^(-1) * X’ * X0;

function doa_eigen_projection

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%   this function uses eigen projection techniques to estimate the bearing
%   angles
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

s_confg = [-5:1:5]’; % this is a linear array with spacing of 1 m
ns = numel(s_confg);
freq  = [100 100 100]; % This is the freq at which source is emitting the signal
spd = 343; % propagation velocity of sound
lambda = spd/freq(1);
wave_num = (2*pi)/lambda;
half_lambda = lambda/2;
s_confg = s_confg.*half_lambda;
y = zeros(ns,1);
z = zeros(ns, 1);
s_confg = [s_confg, y, z];
nsensors = size(s_confg, 1);
fs = 8*1024; % sampling rate

% Now we do the spatial averaging
S = gen_sig(s_confg, spd, freq, fs);
% now we will do the spatial averaging
[n m] = size(S);

ncol = 5;
sr = zeros(ncol+1, ncol+1);
sr2 = zeros(ncol+1, ncol+1);
SF = fft(S, fs, 2);
SF(:,4097:end) = [];
for i = 1:n-(ncol + 1)
    X = SF(i:i+ncol,:);
    R = cov(X’);
    sr = sr + R;
    X2 = S(i:i+ncol,:);
    R2 = cov(X2’);
    sr2 = sr2 + R2;
end
sr = sr./(n-(ncol + 1));
sr2 = sr2./(n-(ncol + 1));
```

```
[Q, R] = eig(sr);
[Q2, R2] = eig(sr2);
[QO] = order_eig_vec(Q, R);
Q = QO;
[QO] = order_eig_vec(Q2, R2);
Q2 = QO;
h1 = figure;
h2 = figure;
d = half_lambda;
gen_response(Q, Q2, wave_num, d, h1, h2);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% now we will do the time average using multiple snapshots
% Average of the covariance matrices for multiple snapshots -- time
% averaging
snap_shots = 10;
sr = zeros(ns, ns);
srf = zeros(ns, ns);
for i = 1:snap_shots
    S = gen_sig(s_confg, spd, freq, fs);
    XF = fft(S,fs,2);
    RF = cov(XF');
    R = cov(S');
    sr = sr + R;
    srf = srf + RF;
end
sr = sr./snap_shots;
srf = srf./snap_shots;
% now decompose the covariance matrix in~to eigenvectors and eigenvalues
[Q, L] = eig(sr); % Q contains the eigenvectors and L contains eigenvalues
                  % most significant eigenvectors are the last three
[QF, LF] = eig(srf);
[Q11] = order_eig_vec(Q, L);
[Q22] = order_eig_vec(QF, LF);
h1 = figure;
h2 = figure;
d = half_lambda;
gen_response(Q11, Q22, wave_num, d, h1, h2);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% simulate the signals
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function S = gen_sig(s_confg, spd, freq, fs)

ang = [15 45 120];
n_ang = numel(ang);
omega1 = 2. * pi * freq;
n = 1:fs;
ts = 1/fs;
t1 = ts * n;
ns = size(s_confg, 1);
sigs = [];

for num = 1:n_ang,
    phi = ang(num)*(pi/180);
    x = cos(phi);
    y = sin(phi);
    z = 0;
    wave_vec = [x; y; z];
    td = s_confg*wave_vec;
    td = td./spd;
    switch num
        case 1
            omega = omega1(num);
        case 2
            omega = omega1(num);
```

```
        case 3
            omega = omega1(num);
    end
    k = rand(1,10);
    k = k(1)*pi;
    for n_mics = 1:ns,
        mic_delay = td(n_mics) + k;
        i = sqrt(-1);
        p1 = cos(omega * (t1 + mic_delay)); % signal generated
        p2 = hilbert(p1);
        x = rand(1,fs); % generated noise to add to the signal
        p1 = p2 + x.*0.24; % noise added
        sigs(num, n_mics,:) = p1;
    end
end
S = zeros(ns, fs);
for i = 1:n_ang,
    y = squeeze(sigs(i,:,:));
    S = S + y;
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function [QO] = order_eig_vec(Q, R)

q = diag(R);
[a, b] = sort(q);
b = flipud(b);
A = Q(:,b);
B = R(:,b);
QO = A;
R = flipud(B);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function gen_response(Q, Q2, wave_num, d, h1, h2)
[nq, mq] = size(Q);
arr1 = zeros(1,180);
arr2 = zeros(1,180);
A = Q(:,4:nq);
B = Q2(:,4:nq);
[nq, mq] = size(A);

for p = 1:mq
    arr3 = zeros(1,180);
    arr4 = zeros(1,180);
    % generate the steering vec V
    for i = 1:180,
        theta = deg2rad(i);
        V = [];
        for k = 0:nq-1,
            q = -j * wave_num * k*d * cos(theta);
            V = [V, exp(q)];
        end
        % response for the time signal
        g = V * A * A' * V';
        arr1(i) = arr1(i) + abs(g); % for average
        g = V * A(:,p) *  A(:,p)' * V';
        arr3(i) =  abs(g);
        % response for the fft
        g2 = V * B * B' * V';
        arr2(i) = arr2(i) + abs(g2); % for average
        g2 = V * B(:,p) *  B(:,p)' * V';
        arr4(i) =  abs(g2);
    end
    figure(h1)
    hold on
    plot(10*log(arr3), 'k:'), grid on
    figure(h2)
```

```
    hold on
    plot(10*log(arr4), 'k:'), grid on
end
figure(h1)
hold on
% find the average response
arr1 = arr1./(nq-3);
plot(10*log(arr1), 'k', 'LineWidth', 2), grid on
ylabel 'Magnitude (db) -->'
xlabel 'Angle (deg) -->'

figure(h2)
hold on
% find the average response
arr2 = arr2./(nq-3);
plot(10*log(arr2), 'k', 'LineWidth', 2), grid on
ylabel 'Magnitude (db) -->'
xlabel 'Angle (deg) -->'

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%    END OF MUSIC FOR TIME ANDD SPACE AVERAGING
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%             MVDR ALGORITHM
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%


function mvdr

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%   this function uses MVDR techniques to estimate the bearing
%   angles
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
close all
s_confg = [0:1:10]'; % this is a linear array with spacing of 1 m
ns = numel(s_confg);
freq  = [100 100 100]; % This is the freq at which source is emitting the signal
spd = 343; % propagation velocity of sound
lambda = spd/freq(1);
wave_num = (2*pi)/lambda;
half_lambda = lambda/2;
s_confg = s_confg.*half_lambda;
y = zeros(ns,1);
z = zeros(ns, 1);
s_confg = [s_confg, y, z];
% nsensors = size(s_confg, 1);
fs = 8*1024; % sampling rate

% Now we do the spatial averaging
S = gen_sig(s_confg, spd, freq, fs);
% now we will do the spatial averaging
[n, m] = size(S);

ncol = 5;
sr = zeros(ncol + 1, ncol + 1);
sr2 = zeros(ns, ns);
SF = fft(S, fs, 2);
SF(:,4097:end) = [];
arr = [];
d = half_lambda;
h2 = figure;
for i = 1:n-(ncol + 1) % for freq. domain we are doing spatial averaging
    X = SF(i:i+ncol,:);
    R = cov(X');
```

```
    sr = sr + R;
    strt = (i-1)*250 + 1;
    stp = strt + 4096;
    X2 = S(:,strt:stp);
    R2 = cov(X2');
    sr2 = sr2 + R2;
end
sr = sr./(n-(ncol + 1));
sr2 = sr2./(n-(ncol + 1));


h1 = figure;
comp_mvdr_spec(sr, wave_num, d, h2);
% R = cov(S');
comp_mvdr_spec(sr2, wave_num, d, h1);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% simulate the signals
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function S = gen_sig(s_confg, spd, freq, fs)

ang = [15 45 120];
n_ang = numel(ang);
omega1 = 2. * pi * freq;
n = 1:fs;
ts = 1/fs;
t1 = ts * n;
ns = size(s_confg, 1);
sigs = [];

for num = 1:n_ang,
    phi = ang(num)*(pi/180);
    x = cos(phi);
    y = sin(phi);
    z = 0;
    wave_vec = [x; y; z];
    td = s_confg*wave_vec;
    td = td./spd;
    switch num
        case 1
            omega = omega1(num);
        case 2
            omega = omega1(num);
        case 3
            omega = omega1(num);
    end
    k = rand(1,10);
    k = k(1)*pi;
    for n_mics = 1:ns,
        mic_delay = td(n_mics) + k;
        p1 = cos(omega * (t1 + mic_delay)); % signal generated
        p2 = hilbert(p1);
        x = rand(1,fs); % generated noise to add to the signal
        p1 = p2 + x.*0.24; % noise added
        sigs(num, n_mics,:) = p1;
    end
end
S = zeros(ns, fs);
for i = 1:n_ang,
    y = squeeze(sigs(i,:,:));
    S = S + y;
end
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function arr1 = comp_mvdr_spec(R, wave_num, d, h1)

[nr, mr] = size(R);
arr1 = zeros(1,180);
RI = inv(R);
j = sqrt(-1);

for i = 1:180,
    theta = deg2rad(i);
    V = [];
    for k = 0:nr-1,
        q = -j * wave_num * k*d * cos(theta);
        V = [V, exp(q)];
    end
    g = real(V * RI * V');
    arr1(i) = arr1(i) + (1/g); % for average
end
figure(h1)
hold on

plot(10*log10(arr1), 'k', 'LineWidth', 2), grid on
ylabel 'Magnitude (db) -->'
xlabel 'Angle (deg) -->'


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%    END OF MVDR FOR TIME AND FREQ. SPACE AVERAGING
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

Chapter 8
Tracking

In a battlefield, situational awareness is of paramount importance. Situational awareness demands knowledge of surroundings and knowledge of all the vehicles coming into and going out of the field. In a battlefield, it is absolutely crucial to know where enemy vehicles are at all times. This means one must be able to track all the vehicles. To do so, one may deploy several types of sensors such as radar, distributed acoustic arrays, etc. In this chapter, we discuss the use of distributed acoustic arrays to track vehicles, as shown in Fig. 8.1. The arrays find the DOAs of the signals emitted by each vehicle. The DOAs are communicated to the nearby hub, which relays them to the central information processing (CIP) system. The CIP process the DOAs to estimate the coordinates of the vehicles by triangulation. Note that, in order to triangulate a vehicle's position, one has to use the DOA's from multiple arrays that correspond to the same vehicle. This process calls for data association. Data association can be done based on the frequencies (acoustic signals) emitted by the vehicles and the dynamics of the vehicles, that is, the position, velocity, etc. In this chapter, we study various aspects of tracking that are applicable for battlefield acoustics.

Tracking is a process used to follow a target over a period of time. When tracking is done electronically, that is, no person in the loop to guide the process, estimation of the expected location of a target is crucial. For example, radio detection and ranging (radar) is widely used to track airborne traffic. The radar sweeps the space 360° every few seconds. The targets appear on the screen as bright dots. When the radar sweeps second time, the targets have already moved and appear on the screen at a slightly different position from the first sweep. Thus the new set must be associated with the previous ones. If the targets are far apart, it is easy to associate them. However, if they are close, then they can only be associated based on the estimation of the location of the targets. The estimation of a target's location is done based on the velocity of the target, direction in which it is traveling, etc. In Fig. 8.2 there are two tracks corresponding to two vehicles. The solid lines correspond to the actual tracks, while the '*' and 'o' correspond to the measurements made by a radar or some other tracking device. Often, the measurements are corrupted by noise, which make the tracking and data association difficult. Measurement errors result due to several reasons. For example, in the case of acoustic sensors, the DOA angles are estimated

T. Damarla, *Battlefield Acoustics*, DOI 10.1007/978-3-319-16036-8_8

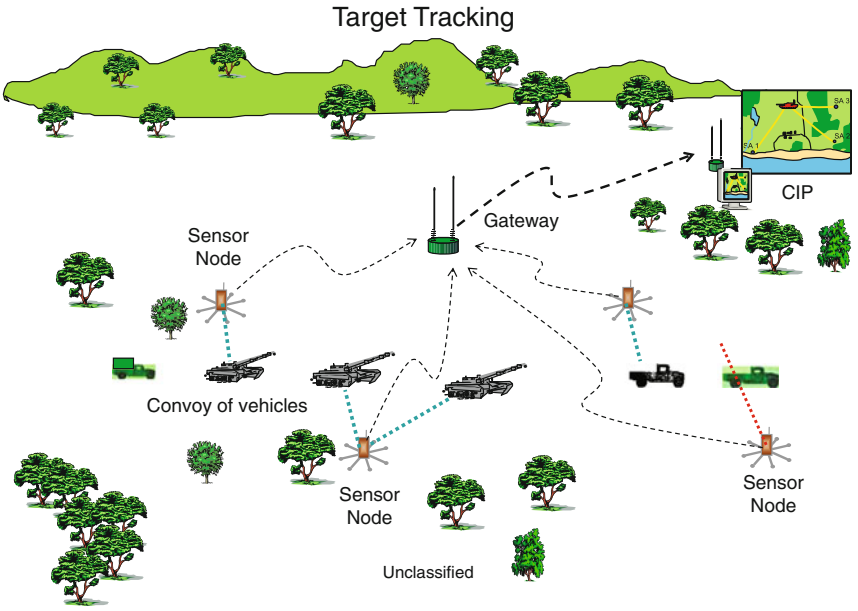


Fig. 8.1 Vehicles traveling

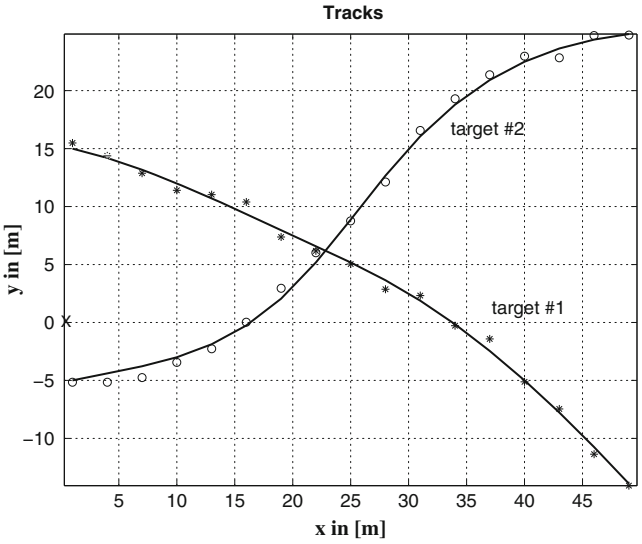


Fig. 8.2 Tracks of two vehicles

by two sensor arrays and then the location of the target is found by triangulation. As discussed in the Chap. 7, DOA estimates are influenced by wind noise, equipment resolution (i.e., the number of array elements), etc. Clearly, data association at the

intersection of the two tracks in Fig. 8.2 would be difficult if the measurement errors are large. However, basing such measurements on the previous history of the tracks of each target, their velocities, and their direction of travel should be sufficient to disambiguate the tracks.

Target motion is best represented using a state space model. Let

$$X(t) = \begin{bmatrix} x(t) \\ y(t) \\ v_x \\ v_y \end{bmatrix} \tag{8.1}$$

be the state vector giving the position of a vehicle at time 't', where $x(t)$ and $y(t)$ are the coordinates of the target at time t, and v_x and v_y correspond to the velocity in the x and y directions. Let F be a state transition matrix that takes a given state $X(t-1)$ to the next state $X(t)$, then

$$X(t) = FX(t-1) \tag{8.2}$$

and an example of the state transition matrix is

$$F = \begin{bmatrix} 1\,0\,1\,0 \\ 0\,1\,0\,1 \\ 0\,0\,1\,0 \\ 0\,0\,0\,1 \end{bmatrix}. \tag{8.3}$$

For a system with noisy measurements (8.2) is re-written as

$$X(t) = FX(t-1) + \begin{bmatrix} \sigma_x^2 & \sigma_y^2 & \sigma_{v_x}^2 & \sigma_{v_x}^2 \end{bmatrix}^T \tag{8.4}$$

where σ^2 is the noise variance and T is the transpose.

8.1 Localization of a Target

In order to track vehicles/targets using acoustic arrays, it is necessary to deploy several arrays in the area where the target tracking is done and the DOAs of the target at each array are measured. Once the DOAs of a target are available, they are used to locate the target position as shown in Fig. 8.3. Consider the target is at an angle θ_1 with respect to acoustic array 1 and θ_2 with respect to array 2 and the distance between the two arrays is d. Triangulation is just drawing a line (the dotted line on Fig. 8.3) at an angle θ_1 at array 1 with respect to the x-axis and another line at an angle θ_2 at array 2. These dotted lines intersect at a point on the XY-plane where the target is located.

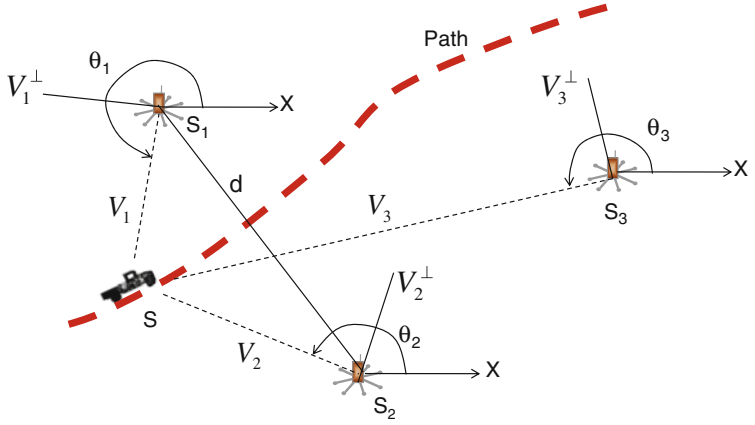


Fig. 8.3 Triangulation

Let us denote the array locations as $\{S_1, S_2, \ldots, S_n\}$, where $S_i = [x_i \; y_i]$, and the location of the target as $S = [x \; y]$. The vectors corresponding to the DOAs are denoted by $\{V_1, V_2, \ldots, V_n\}$, where $V_i = [\cos\theta_i \;\; \sin\theta_i]$. Let $V_i^\perp = [-\sin\theta_i \;\; \cos\theta_i]$ denote the vector that is orthogonal to V_i. Then the vector joining the two locations S and S_1 is orthogonal to $V_1^\perp$ (see Fig. 8.3). Then one can write

$$\begin{bmatrix} S - S_1 \\ S - S_2 \\ \vdots \\ S - S_n \end{bmatrix} \begin{bmatrix} V_1^\perp \\ V_2^\perp \\ \vdots \\ V_n^\perp \end{bmatrix}^T = 0 \tag{8.5}$$

or

$$S \begin{bmatrix} V_1^\perp \\ V_2^\perp \\ \vdots \\ V_n^\perp \end{bmatrix}^T = \begin{bmatrix} S_1 \\ S_2 \\ \vdots \\ S_n \end{bmatrix} \begin{bmatrix} V_1^\perp \\ V_2^\perp \\ \vdots \\ V_n^\perp \end{bmatrix}^T \tag{8.6}$$

Let us denote

$$V_P = \begin{bmatrix} V_1^\perp \\ V_2^\perp \\ \vdots \\ V_n^\perp \end{bmatrix} = \begin{bmatrix} -\sin\theta_1 & \cos\theta_1 \\ -\sin\theta_2 & \cos\theta_2 \\ \vdots & \vdots \\ -\sin\theta_n & \cos\theta_n \end{bmatrix}$$

and the sensor locations are denoted by

$$S_L = \begin{bmatrix} S_1 \\ S_2 \\ \vdots \\ S_n \end{bmatrix} = \begin{bmatrix} x_1 & y_1 \\ x_2 & y_2 \\ \vdots & \vdots \\ x_n & y_n \end{bmatrix}$$

Then

$$[S_L]\,[V_P]^T = \text{diag}\left[[S_L]\,[V_P]^T\right] = \begin{bmatrix} -x_1 \sin\theta_1 + y_1 \cos\theta_1 \\ -x_2 \sin\theta_2 + y_2 \cos\theta_2 \\ \vdots \\ -x_n \sin\theta_n + y_n \cos\theta_n \end{bmatrix}$$

where 'diag' implies diagonal elements of a matrix. Let us denote

$$D_{SV} = \text{diag}\left[[S_L]\,[V_P]^T\right]$$

in the above expression. Then from (8.6), the solution for the target location is given by

$$S = [D_{SV}]^T \left[[V_P]^{-1}\right]^T. \tag{8.7}$$

Equation (8.7) gives the coordinates of the target $S = [x\ y]$. So by estimating the DOA angles of a target, one can localize the position of the target. By finding the location of the target at regular intervals of time, one can track the target over a period of time. Often the estimated DOA angles are noisy, hence the target position estimates will be noisy too, as shown in Fig. 8.2. So, in order to track a target one has to use some kind of filter to provide a smooth estimate of the target position over the time intervals. The rest of the chapter provides several filters, Bayes filter, Kalman, extended Kalman, unscented Kalman and particle filters, which are widely used in practice.

The Bayes filter [43] provides the mathematical foundation for the sequential estimation.

8.2 Bayes Filter

Prior to developing filter theory, let us first understand the problem of sequential estimation. Sequential estimation is the process of understanding the data incrementally as it is gathered. We have seen the problem of sequential estimation of the mean of a random variable x (Chap. 4 in Eq. (4.2)). It is repeated below:

$$\bar{x} = \frac{k-1}{k}\,\bar{x}_{k-1} + \frac{1}{k}\,x_k \tag{8.8}$$

as the kth measurement available. Note that if we have the *pdf* $p_X(x)$ of x, we could directly compute the mean as

$$\bar{x} = \int x\; p_X(x)dx, \tag{8.9}$$

or if we don't have the *pdf* but have k number of samples, then

$$\bar{x} = \frac{1}{k}\sum_{i=1}^{k} x_i. \tag{8.10}$$

Problem Formulation for a Sequential Filter: Consider the system dynamic difference equation and the observation model

$$\mathbf{x}_k = f\left(\mathbf{x}_{k-1}\right) + ba_k + W_k, \tag{8.11}$$

$$\mathbf{y}_k = h\left(\mathbf{x}_k\right) + V_k, \tag{8.12}$$

where a_k is some driving force, W_k and V_k are the process and measurement noises, which are uncorrelated with covariance matrices Q_k and R_k, respectively. The first Eq. (8.11) dictates state transition, while the second Eq. (8.12) is the measurement model. The objective is to estimate recursively the signal $\mathbf{x}_k$, for all k, from the observations $\mathbf{y}_{1:k} = \left\{\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_k\right\}$. Consider the Bayes rule

$$p\left(\mathbf{x}_k|\mathbf{y}_k\right) = \frac{p\left(\mathbf{y}_k|\mathbf{x}_k\right)\, p\left(\mathbf{x}_k\right)}{p\left(\mathbf{y}_k\right)} \tag{8.13}$$

where $p\left(\mathbf{x}_k\right)$ is called the *prior*. If we denote the normalizing constant $p\left(\mathbf{y}_k\right)$ by $1/\alpha$, then

$$p\left(\mathbf{x}_k|\mathbf{y}_k\right) = \alpha\; p\left(\mathbf{y}_k|\mathbf{x}_k\right) \times prior. \tag{8.14}$$

The above equation states that the posterior estimate of the state vector $\mathbf{x}_k$ after the observation of $\mathbf{y}_k$ is equal to the likelihood that a particular state generated the observation times the prior. The problem of sequential estimation is starting from initially guessing $\mathbf{x}_0$ and estimating the subsequent values $\mathbf{x}_1, \mathbf{x}_2, \ldots$, that is, $\mathbf{x}_{k-1} \rightarrow \mathbf{x}_k$ after observing $\mathbf{y}_k$. (In the case of tracking, we measure the DOA angles at multiple arrays, hence, $\mathbf{y}_k$ in (8.12) represent the DOA angles and $\mathbf{x}_k$ represent the coordinates of the target). However, from Eq. (8.14), it is clear that we need the prior to estimate $\mathbf{x}_k$. Note, in the sequential estimation, whole data are not available to estimate the prior. Hence, the prior implies the best guess on the distribution of $\mathbf{x}_k$ before getting

$\mathbf{y}_k$. In sequential estimation, the prior uses all the observations and knowledge of the control actions. Thus, the prior distribution of $\mathbf{x}_k$ [43] is

$$\overline{bel}(\mathbf{x}_k) \equiv prior = p\left(\mathbf{x}_k|\mathbf{y}_{k-1}, \mathbf{y}_{k-2}, \ldots, \mathbf{y}_1, a_k, a_{k-1}, \ldots, a_0\right). \tag{8.15}$$

Then the posterior estimation of the state (see 8.14) is given by

$$bel(\mathbf{x}_k) = \alpha\ p\left(\mathbf{y}_k|\mathbf{x}_k\right)\ \overline{bel}(\mathbf{x}_k) \tag{8.16}$$

Hence, if the $\overline{bel}$ can be estimated, then we can estimate the posterior distribution. This equation is known as the *Measurement* update equation—and it shows how the measurement $\mathbf{y}_k$ changes the posterior into the prior.

In order to estimate $\overline{bel}(\mathbf{x}_k)$ as an iterative process, we like to express it in terms of $bel(\mathbf{x}_{k-1})$. This requires some manipulation—towards that goal we first write $\overline{bel}(\mathbf{x}_k)$ as a joint distribution of $\mathbf{x}_k$ and $\mathbf{x}_{k-1}$ conditioned on all previous observations and the controls. Note that the marginals can be obtained from the joint distributions by "integrating out" the unneeded variables. Therefore,

$$\begin{aligned}\overline{bel}(\mathbf{x}_k) &= p\left(\mathbf{x}_k|\mathbf{y}_{k-1}, \mathbf{y}_{k-2}, \ldots, \mathbf{y}_1, a_k, a_{k-1}, \ldots, a_0\right) \\ &= \int p\left(\mathbf{x}_k, \mathbf{x}_{k-1}|\mathbf{y}_{k-1}, \mathbf{y}_{k-2}, \ldots, \mathbf{y}_1, a_k, a_{k-1}, \ldots, a_0\right) d\mathbf{x}_{k-1}.\end{aligned} \tag{8.17}$$

We now use the fact that $p\left(A, B|C\right) = p(A|B, C)p(B|C)$ to obtain

$$\begin{aligned}\overline{bel}(\mathbf{x}_k) &= \int p\left(\mathbf{x}_k, \mathbf{x}_{k-1}|\mathbf{y}_{k-1}, \mathbf{y}_{k-2}, \ldots, \mathbf{y}_1, a_k, a_{k-1}, \ldots, a_0\right) d\mathbf{x}_{k-1} \\ &= \int p\left(\mathbf{x}_k|\mathbf{x}_{k-1}, \mathbf{y}_{k-1}, \mathbf{y}_{k-2}, \ldots, \mathbf{y}_1, a_k, a_{k-1}, \ldots, a_0\right) \\ &\qquad \times p\left(\mathbf{x}_{k-1}|\mathbf{y}_{k-1}, \mathbf{y}_{k-2}, \ldots, \mathbf{y}_1, a_k, a_{k-1}, \ldots, a_0\right) d\mathbf{x}_{k-1}\end{aligned} \tag{8.18}$$

The first term in (8.18) is the conditional distribution of the current state $\mathbf{x}_k$ given the previous state $\mathbf{x}_{k-1}$ and the previous observations and controls. From (8.11), we know that the current state only depends on the previous state and the current control signal and it does not depend on the previous observations or controls. Hence,

$$p\left(\mathbf{x}_k|\mathbf{x}_{k-1}, \mathbf{y}_{k-1}, \mathbf{y}_{k-2}, \ldots, \mathbf{y}_1, a_k, a_{k-1}, \ldots, a_0\right) = p\left(\mathbf{x}_k|\mathbf{x}_{k-1}, a_k\right). \tag{8.19}$$

Since the state $\mathbf{x}_{k-1}$ does not depend on the control a_k, then the second term in Eq. (8.18) becomes

$$\begin{aligned}p\left(\mathbf{x}_{k-1}|\mathbf{y}_{k-1}, \mathbf{y}_{k-2}, \ldots, \mathbf{y}_1, a_k, \ldots, a_0\right) = \\ p\left(\mathbf{x}_{k-1}|\mathbf{y}_{k-1}, \mathbf{y}_{k-2}, \ldots, \mathbf{y}_1, a_{k-1}, \ldots, a_0\right)\end{aligned} \tag{8.20}$$

All the filters presented in this chapter use this process. But the right-hand side of (8.20) is nothing but $bel(\mathbf{x}_{k-1})$. Thus, (8.18) becomes

$$\overline{bel}(\mathbf{x}_k) = \int p\,(\mathbf{x}_k|\mathbf{x}_{k-1})\;\; bel(\mathbf{x}_{k-1})\;\, d\mathbf{x}_{k-1}. \tag{8.21}$$

This equation is called the *Prediction* update; in other words, it is our understanding of what the current state should be given the previous state and the control.

The Bayes filter just combines the *Prediction* update with the *Measurement* update:

$$\overline{bel}(\mathbf{x}_k) = \int p\,(\mathbf{x}_k|\mathbf{x}_{k-1})\;\; bel(\mathbf{x}_{k-1})\;\, d\mathbf{x}_{k-1}. \tag{8.22}$$

$$bel(\mathbf{x}_k) = \alpha\;\, p\,(\mathbf{y}_k|\mathbf{x}_k)\;\; \overline{bel}(\mathbf{x}_k) \tag{8.23}$$

If the motion dynamics of the target are linear and the noise is Gaussian, it can be shown that the Kalman filter is optimal and should be used to track the targets. For nonlinear motions dynamics, there are several approximations that one can apply to the Kalman. These approximations are called extended Kalman, and unscented Kalman filters.

8.3 Kalman Filter

In 1960 Kalman published his seminal paper [57], "A new approach to linear filtering and prediction problems," which has been intensively studied for the past half a century by estimation theorists. The filtering approach proposed is elegant, simple, retractable and above all recursive. It provides a mathematical framework for estimating the state of a process that minimizes the mean squared error.

Suppose $\boldsymbol{X} \in \mathcal{R}^n$ is an n-dimensional vector that represents the state of a process that needs to be estimated at kth discrete time given its past state at $(k-1)$th time. Then the discrete process can be represented by the difference equation

$$\boldsymbol{X}_k = F\boldsymbol{X}_{k-1} + B\boldsymbol{u}_{k-1} + \boldsymbol{w}_{k-1}\ , \tag{8.24}$$

with the measurement $\boldsymbol{Y}_k \in \mathcal{R}^m$

$$\boldsymbol{Y}_k = H\boldsymbol{X}_k + \boldsymbol{v}_k\ , \tag{8.25}$$

where F is the state transition matrix that relates two states at steps $(k-1)$ and k, B is a $n \times q$ transformational matrix for the optional input $\boldsymbol{u} \in \mathcal{R}^q$, H is a $m \times n$ matrix relating state vector to the measurement vector and the random variables $\boldsymbol{w}_k$ and $\boldsymbol{v}_k$ represent the process and measurement noise, respectively. Both $\boldsymbol{w}_k$ and $\boldsymbol{v}_k$

are assumed to be independent and their distributions are a zero mean white noise process, which mean that the errors are not correlated backward or forward in time so that

$$E\left[\boldsymbol{w}_k \boldsymbol{v}_k^T\right] = 0, \quad \forall k \tag{8.26}$$

$$E\left[\boldsymbol{w}_k \boldsymbol{w}_j^T\right] = \begin{cases} 0, & \text{if } k \neq j \\ Q_k, & \text{if } k = j \end{cases} \tag{8.27}$$

$$E\left[\boldsymbol{v}_k \boldsymbol{v}_j^T\right] = \begin{cases} 0, & \text{if } k \neq j \\ R_k, & \text{if } k = j \ . \end{cases} \tag{8.28}$$

As mentioned earlier, our goal is to estimate the state $\boldsymbol{X}_k$ as accurately as possible based on the information we have about its state at step $(k-1)$ and the input $\boldsymbol{Y}_k$. So estimation of the state $\boldsymbol{X}_k$ is a two-step process, namely, (a) the a priori state $\widehat{\boldsymbol{X}}_k^-$ estimation at step k given the state at step $(k-1)$ and (b) the a posteriori state $\widehat{\boldsymbol{X}}_k$ estimation given the measurement $\boldsymbol{Y}_k$. This leads to the errors

$$\boldsymbol{e}_k^- \equiv \boldsymbol{X}_k - \widehat{\boldsymbol{X}}_k^- \ , \tag{8.29}$$

and

$$\boldsymbol{e}_k \equiv \boldsymbol{X}_k - \widehat{\boldsymbol{X}}_k \ , \tag{8.30}$$

and their respective covariances are

$$P_k^- = E\left[\boldsymbol{e}_k^- \left(\boldsymbol{e}_k^-\right)^T\right] \ , \tag{8.31}$$

$$P_k = E\left[\boldsymbol{e}_k \boldsymbol{e}_k^T\right] \ . \tag{8.32}$$

We compute the a priori state $\widehat{\boldsymbol{X}}_k^-$ estimation using (8.24) and the a posteriori state $\widehat{\boldsymbol{X}}_k$ as

$$\widehat{\boldsymbol{X}}_k = \widehat{\boldsymbol{X}}_k^- + K\left(\boldsymbol{Y}_k - H\widehat{\boldsymbol{X}}_k^-\right) \tag{8.33}$$

where K is called the Kalman gain and the difference between the measurement $\boldsymbol{Y}_k$ and the predicted value $H\widehat{\boldsymbol{X}}_k^-$; $\left(\boldsymbol{Y}_k - H\widehat{\boldsymbol{X}}_k^-\right)$ is called thee innovation or residual. Zero residual implies that the prediction and measurements are in agreement. Selection of the $n \times m$ matrix K is done so that the error covariance P_k is minimum.

We now estimate the error covariances. Re-writing the a priori and a posteriori states

$$\widehat{\boldsymbol{X}}_k^- = F\widehat{\boldsymbol{X}}_{k-1} + B\boldsymbol{u}_{k-1}, \tag{8.34}$$

$$\widehat{\boldsymbol{X}}_k = \widehat{\boldsymbol{X}}_k^- + K\left[\boldsymbol{Y}_k - H\widehat{\boldsymbol{X}}_k^-\right]. \tag{8.35}$$

Ignoring the deterministic input $\boldsymbol{u}_k$, the a priori error is

$$\boldsymbol{e}_k^- \equiv \boldsymbol{X}_k - \widehat{\boldsymbol{X}}_k^- = F\left(\boldsymbol{X}_{k-1} - \widehat{\boldsymbol{X}}_{k-1}^-\right) + \boldsymbol{w}_{k-1} = F\boldsymbol{e}_{k-1} + \boldsymbol{w}_{k-1} \tag{8.36}$$

Then from (8.31)

$$\begin{aligned} P_k^- &= E\left[\boldsymbol{e}_k^-\left(\boldsymbol{e}_k^-\right)^T\right] \\ &= E\left[F\boldsymbol{e}_{k-1}\boldsymbol{e}_{k-1}^T F^T\right] + E\left[F\boldsymbol{e}_{k-1}\boldsymbol{w}_{k-1}^T\right] + E\left[\boldsymbol{w}_{k-1}\boldsymbol{e}_{k-1}^T F^T\right] \\ &\quad + E\left[\boldsymbol{w}_{k-1}\boldsymbol{w}_{k-1}^T\right] \end{aligned} \tag{8.37}$$

From (8.24) we note that $\boldsymbol{e}_{k-1}$ and $\boldsymbol{w}_{k-1}$ correspond to two different time steps, hence they are uncorrelated, that is,

$$E\left[\boldsymbol{e}_{k-1}\boldsymbol{w}_{k-1}^T\right] = E\left[\boldsymbol{e}_{k-1}^T\boldsymbol{w}_{k-1}\right] = 0.$$

Equation (8.37) becomes

$$P_k^- = FP_{k-1}F^T + Q. \tag{8.38}$$

Now, we estimate the a posteriori error covariance

$$P_k = E\left[e_k e_k^T\right] = E\left[\left(\boldsymbol{X}_k - \widehat{\boldsymbol{X}}_k\right)\left(\boldsymbol{X}_k - \widehat{\boldsymbol{X}}_k\right)^T\right] \tag{8.39}$$

We substitute

$$\widehat{\boldsymbol{X}}_k = \widehat{\boldsymbol{X}}_k^- + K\left(\boldsymbol{Y}_k - H\widehat{\boldsymbol{X}}_k^-\right) \tag{8.40}$$

to compute

$$\begin{aligned} \boldsymbol{X}_k - \widehat{\boldsymbol{X}}_k = e_k &= \boldsymbol{X}_k - \left(\widehat{\boldsymbol{X}}_k^- + K\left(\boldsymbol{Y}_k - H\widehat{\boldsymbol{X}}_k^-\right)\right) \\ &= \left(\boldsymbol{X}_k - \widehat{\boldsymbol{X}}_k^-\right) - K\left(H\boldsymbol{X}_k + \boldsymbol{v}_k\right) + KH\widehat{\boldsymbol{X}}_k^- \\ &= \left(\boldsymbol{X}_k - \widehat{\boldsymbol{X}}_k^-\right) - KH\left(\boldsymbol{X}_k - \widehat{\boldsymbol{X}}_k^-\right) - K\boldsymbol{v}_k \\ &= (I - KH)\,\boldsymbol{e}_k^- - K\boldsymbol{v}_k, \end{aligned} \tag{8.41}$$

where the value of $\boldsymbol{Y}_k$ from (8.25) is used. Now, substituting e_k in (8.39)

$$\begin{aligned} P_k &= E\left[e_k e_k^T\right] \\ &= E\left[(I - KH)\,\boldsymbol{e}_k^-\left(\boldsymbol{e}_k^-\right)^T(I - KH)^T\right] - E\left[(I - KH)\,\boldsymbol{e}_k^-\boldsymbol{v}_k^T K^T\right] \\ &\quad - E\left[K\boldsymbol{v}_k\left(\boldsymbol{e}_k^-\right)^T(I - KH)^T\right] + E\left[K\boldsymbol{v}_k\boldsymbol{v}_k^T K^T\right] \end{aligned} \tag{8.42}$$

From (8.40), we note that $\boldsymbol{e}_k^-$ and $\boldsymbol{v}_k$ are uncorrelated since $\boldsymbol{e}_k$ (not $\boldsymbol{e}_k^-$) directly depends on $\boldsymbol{v}_k$. Therefore, $E\left[\boldsymbol{v}_k\left(\boldsymbol{e}_k^-\right)^T\right] = E\left[\boldsymbol{e}_k^-\boldsymbol{v}_k^T\right] = 0$. This leads to

$$P_k = (I - KH)\,P_k^-(I - KH)^T + KRK^T \tag{8.43}$$

The gain K is selected so as to minimize P_k, which is obtained by

$$\frac{\partial P_k}{\partial K} = -2(I - KH)\,P_k^- H^T + 2KR = 0 \tag{8.44}$$

which leads to

$$K = \frac{P_k^- H^T}{HP_k^- H^T + R} \tag{8.45}$$

Substituting (8.45) in (8.43) we get

$$\begin{aligned} P_k &= (I - KH)\,P_k^-(I - KH)^T + KRK^T \\ &= P_k^- - P_k^- H^T K^T - KHP_k^- + K\left(HP_k^- H^T + R\right)K^T \\ &= P_k^- - P_k^- H^T K^T - KHP_k^- + P_k^- H^T K^T \\ &= P_k^- - KHP_k^- = (I - KH)\,P_k^-. \end{aligned} \tag{8.46}$$

It is informative to analyze the gain equation (8.45); if the measurement noise covariance $R \to 0$, the gain K weights the residual heavily, that is,

$$\lim_{R\to 0} K = H^{-1}.$$

If the a priori error covariance $P_k^- \to 0$, the gain weights the residual less, that is

$$\lim_{P_k^-\to 0} K = 0.$$

Now, we give the Kalman filter algorithm.

Kalman filter algorithm:
Model:

$$\boldsymbol{X}_k = F\boldsymbol{X}_{k-1} + B\boldsymbol{u}_{k-1} + \boldsymbol{w}_{k-1}, \quad \boldsymbol{w}_{k-1} \sim \mathcal{N}(0, Q) \tag{8.47}$$

$$\boldsymbol{Y}_k = H\boldsymbol{X}_k + \boldsymbol{v}_k, \quad \boldsymbol{v}_k \sim \mathcal{N}(0, R) \tag{8.48}$$

Initialize:

$$\begin{aligned} \widehat{\boldsymbol{X}}(t_0) &= \widehat{\boldsymbol{X}}_0 \\ P_0 &= E\left[\left(\boldsymbol{X}_0 - \widehat{\boldsymbol{X}}_0\right)\left(\boldsymbol{X}_0 - \widehat{\boldsymbol{X}}_0\right)^T\right] \end{aligned} \tag{8.49}$$

Time Update:

- Estimate the a priori state

$$\widehat{\boldsymbol{X}}_k^- = F\widehat{\boldsymbol{X}}_{k-1} + B\boldsymbol{u}_{k-1} \tag{8.50}$$

$$P_k^- = FP_{k-1}F^T + Q \tag{8.51}$$

Measurement Update:

- Compute the gain K

$$K = P_k^- H^T \left(HP_k^- H^T + R\right)^{-1} \tag{8.52}$$

- Estimate the a posteriori state

$$\widehat{\boldsymbol{X}}_k = \widehat{\boldsymbol{X}}_k^- + K\left(\boldsymbol{Y}_k - H\widehat{\boldsymbol{X}}_k^-\right) \tag{8.53}$$

- Estimate the a posteriori covariance

$$P_k = (I - KH)\, P_k^- \tag{8.54}$$

Kalman Filter Example:

Here, an example of Kalman filter used for tracking is presented. The model is given by

$$\begin{aligned} X_k &= FX_{k-1} + \omega; & \omega &\sim \mathcal{N}(0, Q) \\ Y_k &= HX_k + v & v &\sim \mathcal{N}(0, R) \end{aligned}$$

where

$$X_k = [x_k \; y_k \; \dot{x}_k \; \dot{y}_k]^T$$

where $\dot{x}_k$ and $\dot{y}_k$ denotes the velocities in the x and y-directions and

$$F = \begin{bmatrix} 1 & 0 & 0.1 & 0 \\ 0 & 1 & 0 & 0.1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}; \quad H = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}$$

The process noise covariance Q

$$P = \begin{bmatrix} 0.1 & 0 & 0 & 0 \\ 0 & 0.1 & 0 & 0 \\ 0 & 0 & 5 & 0 \\ 0 & 0 & 0 & 10 \end{bmatrix}; \quad Q = 10^{-3} \begin{bmatrix} 0 & 0 & 0.0050 & 0 \\ 0 & 0 & 0 & 0.0050 \\ 0.0050 & 0 & 1.0000 & 0 \\ 0 & 0.0050 & 0 & 1.0000 \end{bmatrix}$$

is determined using information in [82, 45]. The entries in P corresponding to the velocities are high in order to reflect the most uncertainty in the velocities initially. The Matlab toolbox [45] on Kalman filters is an useful toolbox for those interested in implementing various filters. The actual track and measurements are plotted in Fig. 8.4 for the example used for the Kalman filter. The filtered output is plotted on the Fig. 8.5. Initial state of the target is set as $X_0 = [0 \;\; 0]^T$ and $R = [1 \; 0; \;\; 0 \; 1]^T$. From the Fig. 8.5, we notice that at the beginning of the track the filtered output and actual track differ quite a bit, but once the filter is stabilized, the Kalman filter is able to track the target closely to the actual trajectory. The Matlab code used for the Kalman filter is presented at the end of this chapter.

Kalman filter is optimal when the noise is Gaussian and only deals with the linear systems where the functions F and H are linear. In practice, very few systems

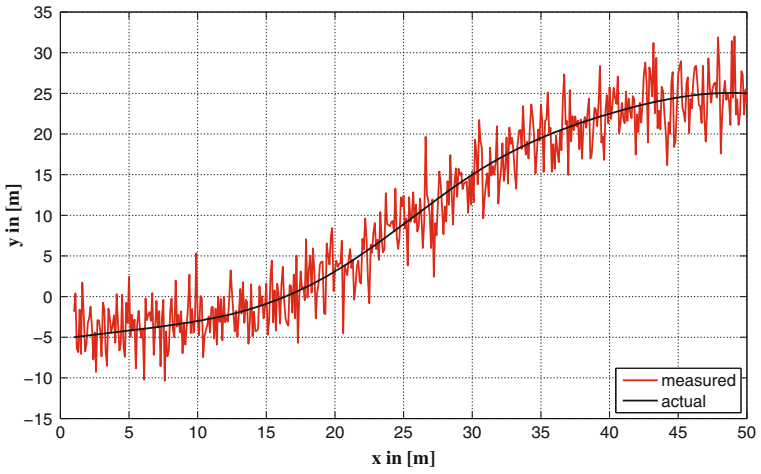


Fig. 8.4 Tracker example for Kalman filter—unfiltered

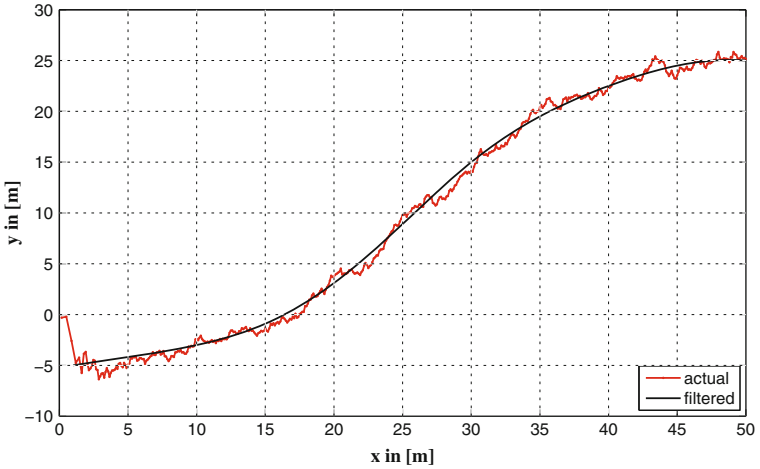


Fig. 8.5 Tracker example for Kalman filter—filtered

have linear behavior. As a result, the Kalman filter equations derived in this section cannot be used. To overcome this problem, several approximations are developed. The extended Kalman filter is one such an approximation and is presented in the following section.

8.4 Extended Kalman Filter

Quite often measurement and processing functions are not linear. For example, for tracking using acoustic sensors, one would first estimate the DOA angles of the target at several widely distributed sensor arrays. These DOAs are then used to triangulate the position of the target as shown in Fig. 8.3. Hence, the matrix H in (8.48) is not a linear function and in-fact it is a trigonometric function. Similarly, in complex systems, the matrix F in (8.47) could be a nonlinear function. The extended Kalman filter linearizes the non-linear functions in matrices F and H by using the first-order terms of Taylor's series approximations. Quite often, the first order terms of Taylor's expansion are sufficient. If better accuracy is needed, Taylor's expansion to the second-order term is used. The derivation of the extended Kalman filter equations is presented below.

Consider the system dynamic difference equation and the observation model

$$\mathbf{X}_k = f\left(\mathbf{X}_{k-1}\right) + W_{k-1}, \tag{8.55}$$

$$\mathbf{Y}_k = h\left(\mathbf{X}_k\right) + V_k, \tag{8.56}$$

where W_k and V_k are the process and measurement noises, which are uncorrelated with covariance matrices Q_k and R_k, respectively. The predicted value is

$$\widehat{\mathbf{X}}_k^- = f\left(\widehat{\mathbf{X}}_{k-1}\right) + W_{k-1} \tag{8.57}$$

The Taylor expansion of $f\left(\widehat{\mathbf{X}}_{k-1}\right)$ about an optimal value $\mathbf{X}_{k-1}^o$ of $\widehat{\mathbf{X}}_{k-1}$ is

$$f\left(\widehat{\mathbf{X}}_{k-1}\right)_{\mathbf{X}_{k-1}^o} \equiv f\left(\mathbf{X}_{k-1}^o\right) + J_f\left(\mathbf{X}_{k-1}^o\right)\left(\widehat{\mathbf{X}}_{k-1} - \mathbf{X}_{k-1}^o\right) + \text{higher order terms} \tag{8.58}$$

where J_f is the Jacobian of the nonlinear function f and is given by

$$J_f \equiv \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial f_n}{\partial x_1} & \frac{\partial f_n}{\partial x_2} & \cdots & \frac{\partial f_n}{\partial x_n} \end{bmatrix} \tag{8.59}$$

where $f(X) = [f_1(X), \ldots, f_n(X)]^T$ and $X = [x_1, \ldots, x_n]^T$. Equation (8.58) can be approximated as

$$\widehat{\mathbf{X}}_k^- \approx f\left(\mathbf{X}_{k-1}^o\right) + J_f\left(\mathbf{X}_{k-1}^o\right) \tag{8.60}$$

Then the error e_k^- is given by

$$\boldsymbol{e}_k^- \equiv \boldsymbol{X}_k - \widehat{\boldsymbol{X}}_k^- = f\left(\boldsymbol{X}_{k-1}\right) - f\left(\widehat{\boldsymbol{X}}_{k-1}^-\right) + \boldsymbol{w}_{k-1} = J_f\left(\mathbf{X}_{k-1}^o\right)\boldsymbol{e}_{k-1} + \boldsymbol{w}_{k-1} \tag{8.61}$$

Then the covariance $P_k^- = E\left[\boldsymbol{e}_k^-\left(\boldsymbol{e}_k^-\right)^T\right]$ is

$$P_k^- = J_f\left(\mathbf{X}_{k-1}^o\right) P_{k-1} J_f^T\left(\mathbf{X}_{k-1}^o\right) + Q_{k-1} \tag{8.62}$$

From (8.38) to (8.62), we notice that F is replaced by J_f for the extended Kalman filter. Now, consider (8.40) for the estimated value $\widehat{\mathbf{X}}_k$ as the sum of $\widehat{\mathbf{X}}_k^-$ plus the Kalman gain times the innovation.

$$\widehat{\boldsymbol{X}}_k = \widehat{\boldsymbol{X}}_k^- + K\left(\boldsymbol{Y}_k - h\left(\widehat{\boldsymbol{X}}_k^-\right)\right), \tag{8.63}$$

which requires the estimation of $h\left(\widehat{\boldsymbol{X}}_k^-\right)$. Once again using Taylor's series expansion of h, we can write

$$h\left(\widehat{\boldsymbol{X}}_k^-\right) \equiv h\left(\boldsymbol{X}_k^o\right) + J_h\left(\boldsymbol{X}_k^o\right)\left(\widehat{\boldsymbol{X}}_k^- - \boldsymbol{X}_k^o\right) + \text{higher order terms} \tag{8.64}$$

Then the posterior error

$$e_k = \boldsymbol{X}_k - \widehat{\boldsymbol{X}}_k = \boldsymbol{X}_k - \left(\widehat{\boldsymbol{X}}_k^- + K\left(\boldsymbol{Y}_k - h\left(\widehat{\boldsymbol{X}}_k^-\right)\right)\right). \tag{8.65}$$

or

$$e_k = \left(\boldsymbol{X}_k - \widehat{\boldsymbol{X}}_k^-\right) - K V_k - K\left(h\left(\boldsymbol{X}_k\right) - h\left(\widehat{\boldsymbol{X}}_k^-\right)\right) \tag{8.66}$$

Substituting (8.64) in (8.66)

$$\begin{aligned} e_k &\approx \left(\boldsymbol{X}_k - \widehat{\boldsymbol{X}}_k^-\right) - K V_k - K\left[h\left(\boldsymbol{X}_k^o\right) + J_h\left(\boldsymbol{X}_k^o\right)\left(\boldsymbol{X}_k - \boldsymbol{X}_k^o\right)\right] \\ &\quad -K\left[h\left(\boldsymbol{X}_k^o\right) + J_h\left(\boldsymbol{X}_k^o\right)\left(\widehat{\boldsymbol{X}}_k^- - \boldsymbol{X}_k^o\right)\right] \\ e_k &\approx \left(\boldsymbol{X}_k - \widehat{\boldsymbol{X}}_k^-\right)\left[1 - K J_h\left(\boldsymbol{X}_k^o\right)\right] - K V_k \\ e_k &\approx e_k^-\left[1 - K J_h\left(\boldsymbol{X}_k^o\right)\right] - K V_k \end{aligned} \tag{8.67}$$

Equation (8.67) has the same form as the e_k in (8.41), hence following the steps in (8.42–8.44) to estimate the gain K result in

$$K = \frac{P_k^- J_h\left(\boldsymbol{X}_k^o\right)}{J_h\left(\boldsymbol{X}_k^o\right) P_k^- J_h\left(\boldsymbol{X}_k^o\right) + R} \tag{8.68}$$

Similarly, the expression for P_k is

$$P_k = \left[1 - K J_h\left(\boldsymbol{X}_k^o\right)\right] P_k^-\left[1 - K J_h\left(\boldsymbol{X}_k^o\right)\right]^T + K R K^T \tag{8.69}$$

Substituting (8.68) in (8.69) and simplifying would result in

$$P_k = \left(I - K J_h\left(\boldsymbol{X}_k^o\right)\right) P_k^-. \tag{8.70}$$

Summary of Extended Kalman Filter

System Equations:

$$\begin{aligned} \boldsymbol{X}_k &= f\left(\boldsymbol{X}_{k-1}\right) + W_{k-1} \\ Y_k &= h\left(\boldsymbol{X}_k\right) + V_k \end{aligned}$$

Initialization:

$$\begin{aligned} \boldsymbol{X}_0 &= \boldsymbol{X}_0^o \quad \text{(mean value)} \\ P_0 &= E\left[\left(\boldsymbol{X}_0 - \boldsymbol{X}_0^o\right)\left(\boldsymbol{X}_0 - \boldsymbol{X}_0^o\right)^T\right] \end{aligned}$$

Model forecast/prediction:

$$\widehat{\boldsymbol{X}}_k^- = f\left(\boldsymbol{X}_{k-1}\right)$$

$$P_k^- = J_f\left(\boldsymbol{X}_{k-1}\right) P_{k-1} J_f\left(\boldsymbol{X}_{k-1}\right)^T + Q$$

Data assimilation/Update:

$$\begin{aligned} \boldsymbol{X}_k &= \widehat{\boldsymbol{X}}_k^- + K\left(Y_k - h\left(\widehat{\boldsymbol{X}}_k^-\right)\right) \\ K &= \frac{P_k^- J_h\left(\boldsymbol{X}_k^o\right)}{J_h\left(\boldsymbol{X}_k^o\right) P_k^- J_h\left(\boldsymbol{X}_k^o\right) + R} \\ P_k &= \left(I - K J_h\left(\boldsymbol{X}_k^o\right)\right) P_k^- \end{aligned}$$

We now present an example to illustrate the nuances of the extended Kalman filter and its implementation for real tracking problems.

Example: Bearing only tracker:

In this example, we try to track a vehicle that is traveling along a curved path using two acoustic arrays situated at different locations. The vehicle's engine makes sound, being a mechanical system, and it rotates at certain number of revolutions per minute (RPM). The sound propagates radially and is sensed by the two arrays. The acoustic arrays then estimate the DOA angles of the sound source (vehicle). Once the DOAs are available, the vehicle location can be triangulated. The measurements are in angles. We now formulate the problem:

State Vector

$$X_k = [x_k \; y_k \; \dot{x}_k \; \dot{y}_k]^T$$

and the state transition is governed by

$$X_k = F X_{k-1} + Q_{k-1}$$

where

$$F = \begin{bmatrix} 1 & 0 & dt & 0 \\ 0 & 1 & 0 & dt \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

where dt is the time interval between the measurements.

Measurement:

$$Y_k = \begin{bmatrix} \theta_1 \\ \theta_2 \\ \vdots \\ \theta_n \end{bmatrix} = H\left(X_k\right) = \begin{bmatrix} \arctan\left(\frac{y_k - S_y^1}{x_k - S_x^1}\right) + r_k^1 \\ \arctan\left(\frac{y_k - S_y^2}{x_k - S_x^2}\right) + r_k^2 \\ \vdots \\ \arctan\left(\frac{y_k - S_y^n}{x_k - S_x^n}\right) + r_k^n \end{bmatrix} \tag{8.71}$$

where (x_k, y_k) are the coordinates of the target at the kth time, the coordinates of the jth acoustic array are given by $S^j = \left(S_x^j, S_y^j\right)$ for all $j \in \{1, 2, \ldots, n\}$ and r_k^j is the measurement noise at the jth acoustic array. Clearly, H is nonlinear. In order to use the extended Kalman filter, one should compute the Jacobian of F and H. Since, F has no entries that depend on the target position, we need not compute the Jacobian for F. However, the measurement function H depends on the position of the target location. The following derivatives should be estimated:

$$\frac{\partial H^i}{\partial x_k} = \frac{\partial \theta_i}{\partial x_k}, \ \frac{\partial H^i}{\partial y_k} = \frac{\partial \theta_i}{\partial y_k}, \ \frac{\partial H^i}{\partial \dot{x}_k} = \frac{\partial \theta_i}{\partial \dot{x}_k}, \ \frac{\partial H^i}{\partial \dot{y}_k} = \frac{\partial \theta_i}{\partial \dot{y}_k}; \quad \forall i \in \{1, 2, \ldots n\}.$$

Using the expression for the derivative of $\frac{\arctan z}{dx} = \frac{1}{1+z^2}\frac{dz}{dx}$ and the chain rule for derivatives, we get

$$\begin{aligned}
\frac{\partial H^i}{\partial x_k} &= \frac{\partial\left(\arctan\left(\frac{y_k - S_y^i}{x_k - S_x^i}\right)\right)}{\partial x_k} = \frac{-\left(y_k - S_y^i\right)}{\left(x_k - S_x^i\right)^2 + \left(y_k - S_y^i\right)^2} \\
\frac{\partial H^i}{\partial y_k} &= \frac{\partial\left(\arctan\left(\frac{y_k - S_y^i}{x_k - S_x^i}\right)\right)}{\partial y_k} = \frac{\left(x_k - S_x^i\right)}{\left(x_k - S_x^i\right)^2 + \left(y_k - S_y^i\right)^2} \\
\frac{\partial H^i}{\partial \dot{x}_k} &= 0 \\
\frac{\partial H^i}{\partial \dot{y}_k} &= 0
\end{aligned}$$

and the Jacobian of function H can be written as

$$J_h = \begin{bmatrix} \frac{-\left(y_k - S_y^1\right)}{\left(x_k - S_x^1\right)^2 + \left(y_k - S_y^1\right)^2} & \frac{\left(x_k - S_x^1\right)}{\left(x_k - S_x^1\right)^2 + \left(y_k - S_y^1\right)^2} & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots \\ \frac{-\left(y_k - S_y^n\right)}{\left(x_k - S_x^1\right)^2 + \left(y_k - S_y^1\right)^2} & \frac{\left(x_k - S_x^n\right)}{\left(x_k - S_x^1\right)^2 + \left(y_k - S_y^1\right)^2} & 0 & 0 \end{bmatrix} \tag{8.72}$$

For this example, two sensor arrays, located at the coordinates (0, 0) and (15, 15), are used to estimate the bearings of the target every $dt = 0.1$ sec. The initial estimate of the target location is set at (0, 0) and the velocities are also set as $\dot{x} = 0$; $\dot{y} = 0$, hence

$$X_0 = [0\ 0\ 0\ 0]^T.$$

The state vector is given by

$$\widehat{X}_k^- = F\left(X_{k-1}\right) + Q$$

where

$$F = \begin{bmatrix} 1 & 0 & 0.1 & 0 \\ 0 & 1 & 0 & 0.1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix},$$

the process noise covariance matrix is set as

$$Q = 10^{-3} \begin{bmatrix} 0.0000 & 0 & 0.0050 & 0 \\ 0 & 0.0000 & 0 & 0.0050 \\ 0.0050 & 0 & 1.0000 & 0 \\ 0 & 0.0050 & 0 & 1.0000 \end{bmatrix},$$

and the process covariance matrix P_0 as

$$P_0 = \begin{bmatrix} 0.1 & 0 & 0 & 0 \\ 0 & 0.1 & 0 & 0 \\ 0 & 0 & 10 & 0 \\ 0 & 0 & 0 & 10 \end{bmatrix}$$

and the measurement equation is

$$Y_k = H\left(\widehat{X}_k^-\right) + R$$

where R is set at

$$R = \begin{bmatrix} 0.0025 & 0 \\ 0 & 0.0025 \end{bmatrix}$$

The algorithm for tracking the target using bearings only is given below:

Extended KF Algorithm:

1. Set $k = 0$, $X_0 = [0\ 0\ 0\ 0]^T$ and the $P_0 = \text{diag}\,[0.1\ 0.1\ 10\ 10]$.
2. Increment k by 1. Predict $\widehat{X}_k^-$ using

$$\widehat{X}_k^- = F * X_{k-1};$$

and update the covariance matrix P_k as

$$P_k^- = F * P_{k-1} * F' + Q.$$

3. Get the bearing measurements

$$Y_{\text{meas}} = \left[\theta_k^1 \ \ \theta_k^2\right]^T$$

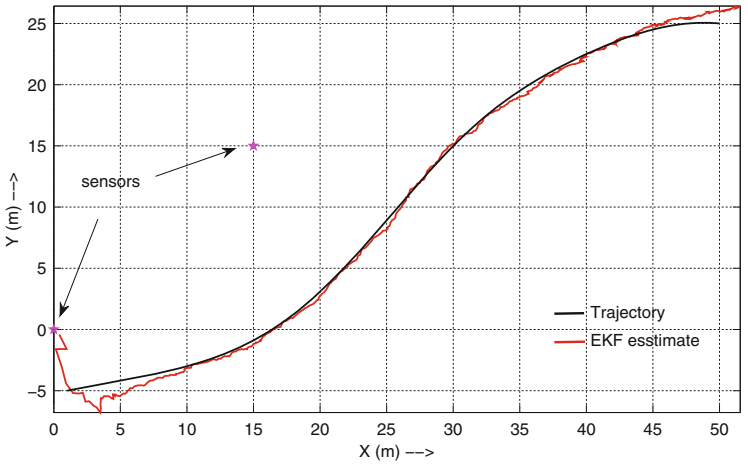


Fig. 8.6 Tracker example for extended Kalman filter using bearings only

4. Here, update X_k by first computing the Jacobian J_h of function H using (8.72) and computing the bearing angles Y of the target using Eq. (8.71) and the estimated coordinates $\widehat{X}_k^-$ with respect to each sensor location. Estimate the Kalman gain using (8.68).
   - Update

$$X_k = \widehat{X}_k^- + K\left(Y_{\text{meas}} - Y\right)$$

   - Update the covariance matrix P_k

$$P_k = P_k^- - K\left(J_h * P_k^- * J_h^T + R\right)K^T$$

5. Go to step 2.

Figure 8.6 shows the results of extended Kalman filter using bearings only. The MATLAB code used for extended Kalman filter for target tracking using bearings only is given at the end of the chapter.

The extended Kalman filter uses Jacobians. For a majority of functions in real-world applications, computation of Jacobians are, at best, difficult. Sometimes, Jacobians may not even exist. Instead, one can easily approximate the probability distributions. The unscented Kalman filter is based on the approximations of probability distributions and is presented in the next section.

8.5 Unscented Kalman Filter

The unscented Kalman filter is first introduced by Julier and Uhlmann [55] in their seminal paper titled "Unscented Filtering and Nonlinear Estimation." The Kalman filter uses the first two moments of the state (mean and the covariance) in its update rule. The mean and covariance are linearly transformable quantities. That is, if the error distribution has the mean $\bar{X}$ and covariance Σ_x, and if the distribution has undergone a linear transformation G, then the transformed mean and covariance are simply, $G\bar{X}$ and $G\Sigma_x G^T$, respectively. However, the majority of the distributions are nonlinear for real-world applications. The most widely used approach for nonlinear state estimation is the extended Kalman filter, as described in Sect. 8.4 where the nonlinear function is replaced by a linear approximation of the function. The extended Kalman filter simply linearizes all nonlinear transforms and substitutes Jacobian matrices for the linear transformations in the Kalman filter equations. However, the extended Kalman filter suffers from some limitations:

- Linearized transforms are useful if the error propagation is well approximated by the linear function.
- It works if Jacobians exist.
- Calculating Jacobians are cumbersome.

Unscented transformation is based on the principle that it is easier to approximate the probability distribution than it is to approximate an arbitrary nonlinear function. The approach consists of selecting a set of points (called sigma points) whose mean and covariance are $\bar{X}$ and Σ_x. The nonlinear function is applied to these points. The statistics (mean and covariance) of the transformed points result in the nonlinearly transformed mean and covariance of the nonlinear function as if we had worked with nonlinear function. The concept of the unscented transformation is shown in Fig. 8.7. The extended Kalman filter uses the first- or second-order Taylor series approximation of the nonlinear functions. The advantage of unscented transformation is that it captures the higher-order moments caused by the nonlinear transform.

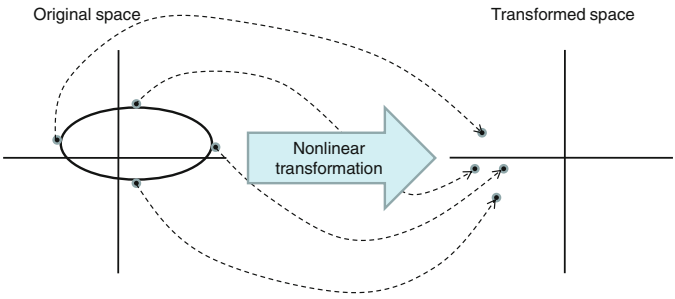


Fig. 8.7 Principle of unscented transformation

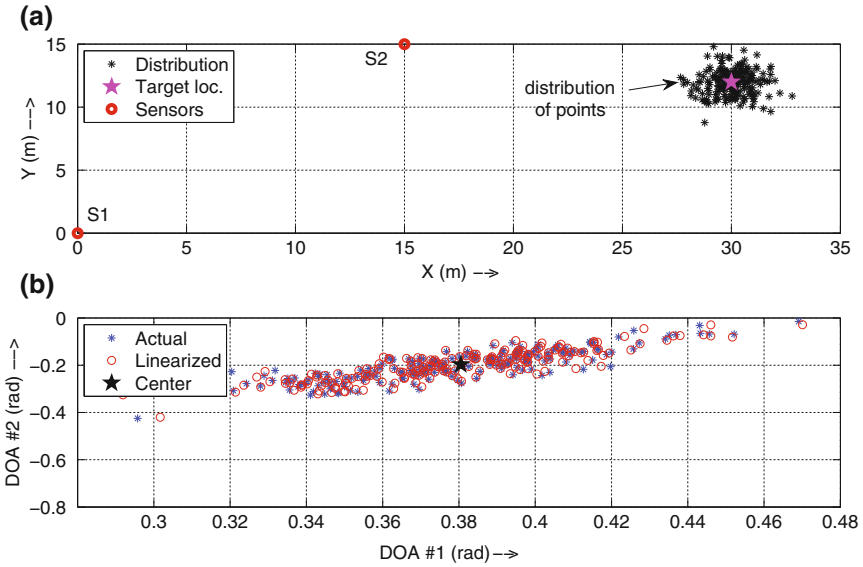


Fig. 8.8 **a** Distribution of points around a target location and, **b** DOA angles with and without linearization

As an example consider the tracking problem used for extended Kalman filter shown in Fig. 8.6 with two sensor arrays located at (0, 0) and (15, 15) making the DOA measurements. Let the target be located at (30, 12) and the two DOAs be $\theta_1 = 0.3805$ and $\theta_2 = -0.1974$ in radians. Now consider a set of points 'X' normally distributed around the target location, as shown in Fig. 8.8. The linearized DOA angles for the points, which are farther from the actual target location, deviated more from the actual DOA angles. This would cause an error in the estimation of target location (the mean) and the covariance; moreover, the error would build up as the tracking progresses.

The sigma points are carefully selected to have a given mean and covariance. Let the set of sigma points be denoted by

$$S = \{x_i\}, \; i \in \{1, 2, \ldots, q\}$$

and a set of weights $W_i, \; i = \{1, 2, \ldots, q\}$ used to provide unbiased estimate such that

$$\sum_{i=1}^{q} W_i = 1. \tag{8.73}$$

Then the mean and the covariance of the transformed sigma points are calculated as

$$\bar{Y} = \sum_{i=1}^{q} W_i H(x_i), \tag{8.74}$$

$$\Sigma_Y = \sum_{i=1}^{q} W_i \left(H(x_i) - \bar{Y}\right)\left(H(x_i) - \bar{Y}\right)^T. \tag{8.75}$$

Let Σ_x be the covariance matrix of X, then a set of sigma points that have the given mean and covariance can be generated as follows:

$$\begin{aligned} x_i &= \bar{x} + \left(\sqrt{N_x \Sigma_x}\right)_i; \quad W_i = \tfrac{1}{2N_x} \\ x_{i+N_x} &= \bar{x} - \left(\sqrt{N_x \Sigma_x}\right)_i; \quad W_{i+N_x} = \tfrac{1}{2N_x} \end{aligned} \tag{8.76}$$

where $\left(\sqrt{N_x \Sigma_x}\right)_i$ is the ith row or column of the matrix square root of $N_x \Sigma_x$, and W_i is the weight. There are other techniques [82, 45] one can employ in generation of sigma points. However, all these techniques involve finding the square root of covariance matrix Σ_x. As an example, the tracking problem used for extended Kalman filter is again used here for demonstration.

The unscented Kalman filter algorithm:

1. Set $k = 0$, $X_0 = [0\ 0\ 0\ 0]^T$, and the $P_0 = E\left[\left(X_0 - \bar{X}\right)\left(X_0 - \bar{X}\right)^T\right]$
2. Increment k by 1.
3. Prediction
   - Predict $\widehat{X}_k^-$ using

$$\widehat{X}_k^- = F * X_{k-1}.$$

   - Set $P_k^- = P_{k-1}$ and compute weights W_i, $\forall i \in \{1, 2, \ldots, N_x\}$ and generate sigma points x_i^p using (8.76), where $\bar{x} = \widehat{X}_k^-$ and $\Sigma_x = P_k^-\left(P_k^-\right)^T$.
   - Estimate the mean and covariance as

$$\bar{\widehat{X}}_k^- = \sum_{i=1}^{2N_x} W_i x_i^p \tag{8.77}$$

     and the covariance matrix $\widehat{P}_k^-$ as

$$\widehat{P}_k^- = \sum_{i=1}^{2N_x} W_i \left(x_i^p - \bar{\widehat{X}}_k^-\right)\left(x_i^p - \bar{\widehat{X}}_k^-\right)^T + Q \tag{8.78}$$

4. Get the bearing measurements

$$Y_{\text{meas}} = \left[\theta_k^1 \;\; \theta_k^2\right]^T$$

5. Update:

- Generate the weights and sigma points x_i^u using $\bar{x} = \bar{\widehat{X}_k^-}$ and $\Sigma_x = \widehat{P}_k^-$.
- Transform each sigma point x_i^u into the DOA angles by performing $Y_i = H x_i^u$.
- Estimate the mean of the transformed sigma points

$$\bar{Y} = \sum_{i=1}^{2N_x} W_i Y_i \tag{8.79}$$

- Compute covariance of Σ_Y

$$\Sigma_Y = \sum_{i=1}^{2N_x} W_i \left(Y_i - \bar{Y}\right)\left(Y_i - \bar{Y}\right)^T + R \tag{8.80}$$

- Compute cross covariance Σ_{XY}

$$\Sigma_{XY} = \sum_{i=1}^{2N_x} W_i \left(x_i^u - \bar{x}\right)\left(Y_i - \bar{Y}\right)^T \tag{8.81}$$

- Compute the Kalman gain

$$K = \frac{\Sigma_{XY}}{\Sigma_Y} \tag{8.82}$$

- Update

$$X_k = \bar{x} + K * \left(Y_{\text{meas}} - \bar{Y}\right) \tag{8.83}$$

$$P_k = \widehat{P}_k^- - K \Sigma_Y k^T. \tag{8.84}$$

6. Go to step 2.

The above algorithm is used to track the target using the DOA estimations measured at two sensors. The results from the unscented Kalman filter are shown in Fig. 8.9.

In the next section, we present particle filters, which are well suited for modeling the probability densities and became popular due to their robustness.

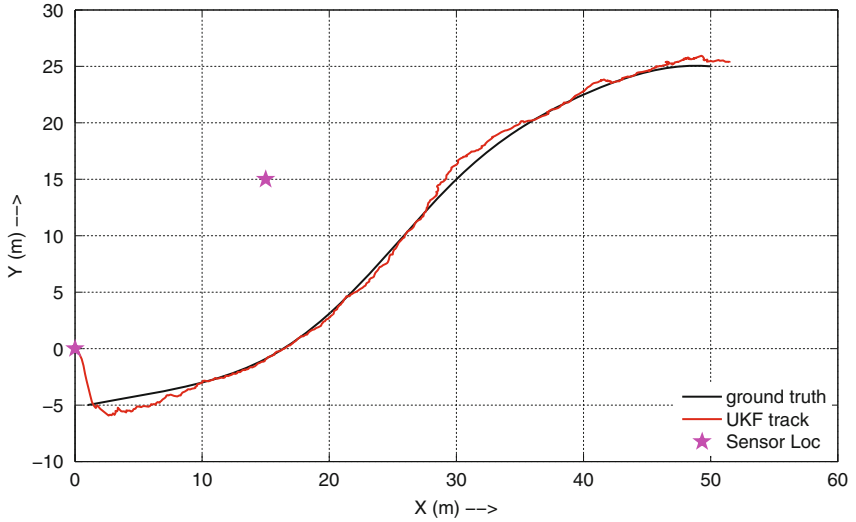


Fig. 8.9 Tracking using unscented Kalman filter

8.6 Particle Filter

We have seen in the previous sections various techniques used for tracking. When the noise is Gaussian and the unknowns are functions of linear variables, Kalman filters are shown to be optimal. For nonlinear signal models and non-Gaussian noise, several approximate solutions, namely, extended Kalman and unscented Kalman filters, are used. Yet there are other powerful filters, namely, sequential importance sampling (SIS), also known as particle filters, that can be used for non-Gaussian noise and nonlinear signal problems. They fall under a broader class of Markov chain Monte Carlo (MCMC) sampling techniques [4, 34].

Consider the system dynamic difference equation and the observation model

$$\mathbf{X}_k = f\left(\mathbf{X}_{k-1}\right) + W_k, \tag{8.85}$$

$$\mathbf{Y}_k = h\left(\mathbf{X}_k\right) + V_k, \tag{8.86}$$

where W_k and V_k are the process and measurement noises, which are uncorrelated with covariance matrices Q_k and R_k, respectively. The objective is to estimate recursively the signal $\mathbf{X}_k$, for all k, from the observations $\mathbf{Y}_{1:k} = \{\mathbf{Y}_1, \mathbf{Y}_2, \ldots, \mathbf{Y}_k\}$. The sequential signal estimation involves the filtering density $p\left(\mathbf{X}_k|\mathbf{Y}_k\right)$ and the predictive density $p\left(\mathbf{X}_{k+l}|\mathbf{Y}_k\right)$, $l \geq 1$.

From the Bayes filter theory presented in Sect. 8.2 using (8.13), (8.22) and (8.23) and with the understanding of the *Prediction* step, we can write

$$p\left(\mathbf{X}_k|\mathbf{Y}_{1:k-1}\right) = \int p\left(\mathbf{X}_k|\mathbf{X}_{k-1}\right)\ p\left(\mathbf{X}_{k-1}|\mathbf{Y}_{1:k-1}\right) d\mathbf{X}_{k-1}. \tag{8.87}$$

The above equation is a direct consequence of Chapman-Kolmogorov identity and Markov property. The *Update* step is

$$p\left(\mathbf{X}_k|\mathbf{Y}_{1:k}\right) = p\left(\mathbf{X}_k|\mathbf{Y}_k, \mathbf{Y}_{1:k-1}\right). \tag{8.88}$$

Using the probability identity

$$p\left(A|B, C\right) = \frac{p\left(B|A, C\right)\ p\left(A|C\right)}{p\left(B|C\right)} \tag{8.89}$$

Equation (8.88) becomes

$$\begin{aligned} p\left(\mathbf{X}_k|\mathbf{Y}_{1:k}\right) &= \frac{p(\mathbf{Y}_k|\mathbf{X}_k, \mathbf{Y}_{1:k-1})\ p(\mathbf{X}_k|\mathbf{Y}_{1:k-1})}{p(\mathbf{Y}_k|\mathbf{Y}_{1:k-1})} \\ &= \frac{p(\mathbf{Y}_k|\mathbf{X}_k)\ p(\mathbf{X}_k|\mathbf{Y}_{1:k-1})}{p(\mathbf{Y}_k|\mathbf{Y}_{1:k-1})} \end{aligned} \tag{8.90}$$

The term $p\left(\mathbf{Y}_k|\mathbf{X}_k\right)$ is the measurement model, $p\left(\mathbf{X}_k|\mathbf{Y}_{1:k-1}\right)$ is the current prior and $p\left(\mathbf{Y}_k|\mathbf{Y}_{1:k-1}\right)$ is the normalization constant. The above equation can be re-written as

$$p\left(\mathbf{X}_{0:k}|\mathbf{Y}_{1:k}\right) = \frac{p\left(\mathbf{Y}_k|\mathbf{X}_k\right) p\left(\mathbf{X}_k|\mathbf{X}_{k-1}\right)}{p\left(\mathbf{Y}_k|\mathbf{Y}_{1:k-1}\right)}\ p\left(\mathbf{X}_{0:k-1}|\mathbf{Y}_{1:k-1}\right). \tag{8.91}$$

Since the transition from $p\left(\mathbf{X}_{0:k-1}|\mathbf{Y}_{1:k-1}\right)$ to $p\left(\mathbf{X}_{0:k}|\mathbf{Y}_{1:k}\right)$ is analytically intractable we use approximation techniques. Particle filters are one such technique. The key idea in particle filters is to represent the posterior density $p\left(\mathbf{X}_k|\mathbf{Y}_k\right)$ function as sum of weighted random samples [4]. If the number of samples is large, then the SIS filter approaches the optimal Bayesian estimate.

Let $\left\{\mathbf{X}^i_{0:k}, w^i_k\right\}_{i=1}^{N_s}$ denote a random measure that characterizes the posterior *pdf* with $\left\{\mathbf{X}^i_{0:k}, i = 0, 1, \ldots, N_s\right\}$ representing a set of support points and their weights $\left\{w^i_k, i = 1, \ldots, N_s\right\}$ and $\mathbf{X}_{0:k} = \left\{\mathbf{X}_j, j = 0, \ldots, k\right\}$ is the set of all states upto kth time step. The weights are normalized such that $\sum_i w^i_k = 1$. Then the posterior density is

$$p\left(\mathbf{X}_{0:k}|\mathbf{Y}_{1:k}\right) \approx \sum_{i=1}^{N_s} w^i_k\ \delta\left(\mathbf{X}_{0:k} - \mathbf{X}^i_{0:k}\right). \tag{8.92}$$

where δ is the Dirac delta function. Such an approximation allows us to use summation instead of complicated integration. For example,

$$E\left(g(x)\right) = \int g(x)\ p(x)dx \tag{8.93}$$

is approximated by

$$E\left(g(x)\right) \approx \sum_{i=1}^{N_s} w^i g\left(x^i\right). \tag{8.94}$$

The weights are selected using the principle of *importance sampling* [7, 31, 33]. Suppose we want to approximate a *pdf* $p(x)$ with a discrete random measure akin to (8.94). One can generate particle x^i and use $w^i = \frac{1}{N_s}$ to approximate $p(x)$. However, if $p(x)$ is intractable, one can use a distribution $\pi(x)$, known as the *importance function* to generate the particles and assign the weights

$$w^{*i} = \frac{p(x)}{\pi(x)} \tag{8.95}$$

which after normalization becomes

$$w^i = \frac{w^{*i}}{\sum_{j=1}^{N_s} w^{*j}}. \tag{8.96}$$

Therefore, if the samples $\mathbf{X_{0:k}^i}$ were drawn from an importance density $\pi\left(\mathbf{X}_{0:k}|\mathbf{Y}_{1:k}\right)$, then the weights in (8.92) are defined by

$$w_k^i \propto \frac{p\left(\mathbf{X}_{0:k}^i|\mathbf{Y}_{1:k}\right)}{\pi\left(\mathbf{X}_{0:k}^i|\mathbf{Y}_{1:k}\right)} \tag{8.97}$$

In the case of sequential sampling, if we have samples constituting an approximation to $p\left(\mathbf{X}_{0:k-1}|\mathbf{Y}_{1:k-1}\right)$ and want to approximate $p\left(\mathbf{X}_{0:k}|\mathbf{Y}_{1:k}\right)$ with a new set of samples, one can select the importance density such that it factorizes as

$$\pi\left(\mathbf{X}_{0:k}|\mathbf{Y}_{1:k}\right) = \pi\left(\mathbf{X}_k|\mathbf{X}_{0:k-1}, \mathbf{Y}_{1:k}\right)\ \pi\left(\mathbf{X}_{0:k-1}|\mathbf{Y}_{1:k-1}\right) \tag{8.98}$$

Then the samples $\mathbf{X}_{0:k}^i \sim \pi\left(\mathbf{X}_{0:k}|\mathbf{Y}_{1:k}\right)$ can be obtained by augmenting the existing samples $\mathbf{X}_{0:k-1}^i \sim \pi\left(\mathbf{X}_{0:k-1}|\mathbf{Y}_{1:k-1}\right)$ with the new state $\mathbf{X}_k^i \sim \pi\left(\mathbf{X}_k|\mathbf{X}_{0:k-1}, \mathbf{Y}_{1:k}\right)$. This process is called the sequential importance sampling.

Now, we derive the weight update equation. Consider the (8.91); it can be written as

$$p\left(\mathbf{X}_{0:k}|\mathbf{Y}_{1:k}\right) \propto p\left(\mathbf{Y}_k|\mathbf{X}_k\right)\ p\left(\mathbf{X}_k|\mathbf{X}_{k-1}\right)\ p\left(\mathbf{X}_{0:k-1}|\mathbf{Y}_{1:k-1}\right) \tag{8.99}$$

Substituting (8.98) and (8.99) in (8.97) would yield

$$\begin{aligned} w_k^i &\propto \frac{p\left(\mathbf{Y}_k|\mathbf{X}_k^i\right)\, p\left(\mathbf{X}_k^i|\mathbf{X}_{k-1}^i\right)\, p\left(\mathbf{X}_{0:k-1}^i|\mathbf{Y}_{1:k-1}\right)}{\pi\left(\mathbf{X}_k^i|\mathbf{X}_{0:k-1}^i,\mathbf{Y}_{1:k}\right)\, \pi\left(\mathbf{X}_{0:k-1}^i|\mathbf{Y}_{1:k-1}\right)} \\ &= w_{k-1}^i\, \frac{p\left(\mathbf{Y}_k|\mathbf{X}_k^i\right)\, p\left(\mathbf{X}_k^i|\mathbf{X}_{k-1}^i\right)}{\pi\left(\mathbf{X}_k^i|\mathbf{X}_{0:k-1}^i,\mathbf{Y}_{1:k}\right)}. \end{aligned} \tag{8.100}$$

For the majority of cases, $\pi\left(\mathbf{X}_k|\mathbf{X}_{0:k-1},\mathbf{Y}_{1:k}\right) = \pi\left(\mathbf{X}_k|\mathbf{X}_{k-1},\mathbf{Y}_k\right)$, that is, the importance density is dependent only on $\mathbf{X}_{k-1}$ and $\mathbf{Y}_k$. This implies only the filtered estimate of $p\left(\mathbf{X}_k|\mathbf{Y}_{1:k}\right)$ is required at each time step. For this case, only $\mathbf{X}_k^i$ needs to be stored; therefore, one can discard the path $\mathbf{X}_{0:k-1}^i$ and the history of observations $\mathbf{Y}_{1:k-1}$. The modified weight is then

$$w_k^i \propto w_{k-1}^i\, \frac{p\left(\mathbf{Y}_k|\mathbf{X}_k^i\right)\, p\left(\mathbf{X}_k^i|\mathbf{X}_{k-1}^i\right)}{\pi\left(\mathbf{X}_k^i|\mathbf{X}_{k-1}^i,\mathbf{Y}_k\right)} \tag{8.101}$$

and the posterior filtered density $p\left(\mathbf{X}_k|\mathbf{Y}_{1:k}\right)$ can be approximated as

$$p\left(\mathbf{X}_k|\mathbf{Y}_{1:k}\right) \approx \sum_{i=1}^{N_s} w_k^i\, \delta\left(\mathbf{X}_k - \mathbf{X}_k^i\right) \tag{8.102}$$

where the weights are given by (8.101).

The Sequential Importance Sampling algorithm:

SIS Particle Filter Algorithm

$$\left[\left\{\mathbf{X}_k^i, w_k^i\right\}_{i=1}^{N_s}\right] = \text{SIS}\left[\left\{\mathbf{X}_{k-1}^i, w_{k-1}^i\right\}_{i=1}^{N_s}, \mathbf{Y}_k\right]$$

- For $i = 1 : N_s$,
  - Draw $\mathbf{X}_k^i \sim \pi\left(\mathbf{X}_k\, \mathbf{X}_{k-1}^i, \mathbf{Y}_k\right)$
  - Assign weights w_k^i for particles according to (8.101)
- End For

It is important to select the proper importance function. If the importance function has the same support as the probability distribution that is being approximated, then the approximation would be better. In practice, the two most widely used importance functions are the prior and optimal importance functions. The prior importance function is given by $p\left(\mathbf{X}_k|\mathbf{X}_{k-1}^i\right)$ and implies particle weight updates by

$$w_k^i \propto w_{k-1}^i\, p\left(\mathbf{Y}_k|\mathbf{X}_k^i\right). \tag{8.103}$$

The optimal importance function has better properties in terms of minimizing the variance [35] of the weights conditioned on the trajectory $\mathbf{X}_{0:k}^i$ and the observations $\mathbf{Y}_{1:k}$ and is given by $p\left(\mathbf{X}_k|\mathbf{X}_{0:k-1}^i, \mathbf{Y}_{1:k}\right)$. When the optimal importance function is used, the weights are changed according to

$$w_k^i \propto w_{k-1}^i \; p\left(\mathbf{Y}_k|\mathbf{X}_{k-1}^i\right). \tag{8.104}$$

It is easier to use the prior importance function rather than the optimal importance function as the latter $p\left(\mathbf{Y}_k|\mathbf{X}_{k-1}^i\right)$ requires integration.

Degeneracy Problem: One of the problems with the SIS particle filter is the degeneracy of the weights. After a few iterations, the majority of the particle weights tend to approach a value of zero, thereby contributing nothing to the filtering process and increasing the computational burden. In order to overcome this, resampling is done. The measure of degeneracy is the effective sample size N_{eff} suggested in [7] and defined as

$$N_{eff} = \frac{N_s}{1 + \text{Var}\left(w_k^{*i}\right)} \tag{8.105}$$

where $w_k^{*i} = p\left(\mathbf{X}_k^i|\mathbf{Y}_{1:k}\right)/\pi\left(\mathbf{X}_k^i|\mathbf{X}_{k-1}^i\mathbf{Y}_k\right)$ is referred to as the "true weight." It is difficult to estimate exactly, but an estimate $\widehat{N_{eff}}$ of N_{eff} can be obtained by

$$\widehat{N_{eff}} = \frac{1}{\sum_{i=1}^{N_s}\left(w_k^i\right)^2} \tag{8.106}$$

where w_k^i is the normalized weight obtained using (8.100).

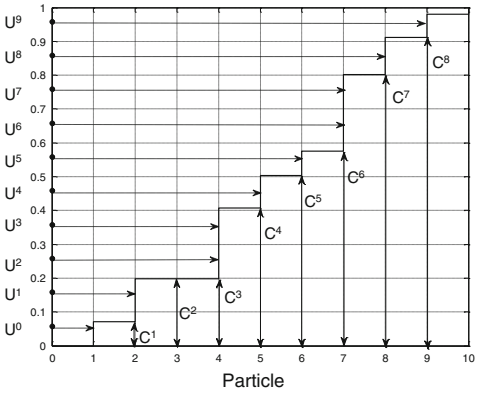
Resampling is a scheme to eliminate the particles with small weights and replicate the particles with large weights. There are several resampling schemes, namely, (a) systematic resampling and (b) residual resampling, etc. We will present the systematic resampling.

Systematic resampling: Let $w_k^i,\ i = \{1, 2, \ldots, N_s = 10\}$ be the particle weights.

i	1	2	3	4	5	6	7	8	9	10
w_k^i	0.0712	0.1975	0.1978	0.4080	0.5042	0.5758	0.8019	0.9115	0.9799	1.0

First we generate a uniform random number $U^0 \sim \mathcal{U}\left[0, \frac{1}{N_s}\right]$, and then the rest of the numbers are generated by adding $\frac{1}{N_s}$ to U^0, that is, $U^q = U^0 + (q-1)/N_s$. For example, if $N_s = 10$ and $U^0 = 0.0158$ is the random number drawn from uniform distribution $\mathcal{U}\left[0, \frac{1}{N_s}\right]$, then

U^0	U^1	U^2	U^3	U^4	U^5	U^6	U^7	U^8	U^9	U^{10}
0.0158	0.1158	0.2158	0.3158	0.4158	0.5158	0.6158	0.7158	0.8158	0.9158	1.0



i	w^i	r^i
1	0.0712	1
2	0.1263	2
3	0.0003	4
4	0.2102	4
5	0.0962	5
6	0.0716	6
7	0.2261	7
8	0.1096	7
9	0.0684	8
10	0.0201	9

Fig. 8.10 Systematic resampling for an example with $N_s = 10$

Systematic sampling first computes the cumulative sum of the weights $C^m = \sum_{i=1}^{m} w_k^i$, hence

C^0	C^1	C^2	C^3	C^4	C^5	C^6	C^7	C^8	C^9	C^{10}
0	0.0712	0.1975	0.1978	0.4080	0.5042	0.5758	0.8019	0.9115	0.9799	1.0000

and compares them with $U^m,\ m = 1, 2, \ldots, N_s$. The number of replications for particle m is determined as the number of uniform numbers in the range $\left[C^{m-1}, C^m\right)$, as shown in Fig. 8.10. For particle 4, U^2 and U^3 belong to range $\left[C^3, C^4\right)$ and hence it is replicated twice; for particle 3, it is discarded because no $U^m,\ m = 1, 2, \ldots, N_s$ appears in the range $\left[C^2, C^3\right)$.

Figures 8.11 and 8.12 show the filtered data for the bearings-only tracking problem considered earlier for the extended and un-scented Kalman filters.

```
function test_kf
%
% this function creates data for tracking and then uses the Kalman filter
% for tracking
%
% To run this program just type   ---     test_kf    in the command window
% of MATLAB
%
%
% written by Thyagaraju Damarla
%

close all
clear all
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%     Create the data for tracking

pop_cords = [1 -5; 5 -4.185; 10 -3; 15 -0.9; 20 3.1; 25 8.9; 30 15; 35 19.5; ...
    40 22.5; 45 24.5; 50 25];
x = pop_cords(1,1):0.1:pop_cords(end,1);
y = interp1(pop_cords(:,1), pop_cords(:,2), x, 'spline');
pop_cords = [x', y'];
```

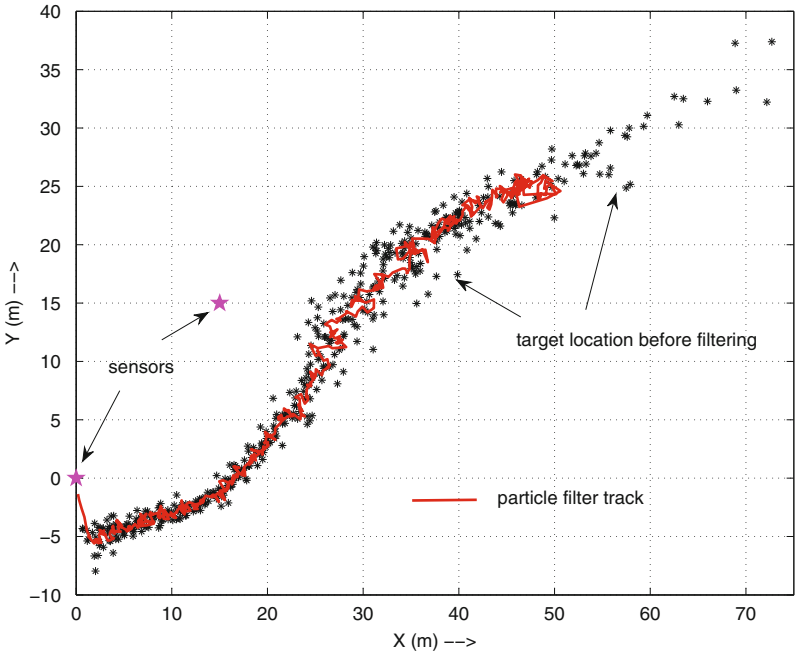


Fig. 8.11 Particle filter output and the unfiltered locations of the target

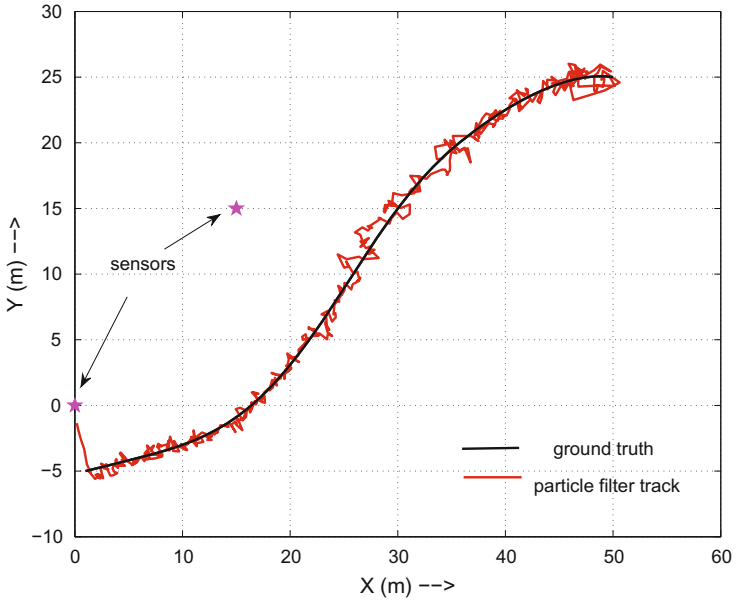


Fig. 8.12 Particle filter output and the ground truth

```
b = randn(size(pop_cords)).*3; % generate noise
tar_loc = pop_cords;
tar_loc(:,2) = tar_loc(:,2) + b(:,2); % add noise to obtain measurement data
figure
hold on
plot(tar_loc(:,1),tar_loc(:,2),'ro'), grid on
plot(pop_cords(:,1),pop_cords(:,2),'b', 'LineWidth', 1.5), grid on

% tar_loc -- represents the measured data
% pop_cords -- represents actual trajectory of the target (ground truth)

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%       Now develop the KF algorithm for tracking
X_k_minus_1 = [0;0;0;0]; % inital state of the state vector
P = diag([0.1 0.1 5 10]); % covariance of process noise
sd = 1.; % standard deviation of the measurement noise
R = sd^2*eye(2); % covariance of the measurement noise
H = [1 0 0 0;
    0 1 0 0]; % measurement transformation matrix
F = [1 0 0.1 0;  % Xk = F*Xkm1 + Q
    0 1 0 0.1;
    0 0 1 0;
    0 0 0 1];
Q =  1.0e-03 * [    0.0000         0              0.0050         0;
                    0              0.0000         0              0.0050;
                    0.0050         0              1.0000         0;
                    0              0.0050         0              1.0000];
[nx, mx] = size(F);
ny = size(pop_cords,1); % number of measurements
MM = zeros(nx, ny);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% perform KF algorithm
for tk = 1:ny,
    [X_hat_minus, CovX] = predict_X(X_k_minus_1, P, F, Q);
    y_measurement = tar_loc(tk, :)';
    [X_hat, P] = update_X(X_hat_minus, CovX, H, y_measurement, R);
    MM(:,tk) = X_hat;
    X_k_minus_1 = X_hat; % for the next cycle
end
close all
figure(1)
h = plot(tar_loc(:,1),tar_loc(:,2),'r',pop_cords(:,1),pop_cords(:,2), ...
     'k', 'LineWidth', 1.5), grid on
legend(h, 'measured', 'actual', 'Location', 'SouthEast');
xlabel('\bfx in [m]','FontSize',16,'FontName','Times');
ylabel('\bfy in [m]','FontSize',16,'FontName','Times');
figure(2)
h2 = plot(MM(1,:), MM(2,:), 'r.-', pop_cords(:,1),pop_cords(:,2),'k', ...
     'LineWidth', 1.5), grid on
legend(h2, 'actual', 'filtered', 'Location', 'SouthEast');
xlabel('\bfx in [m]','FontSize',16,'FontName','Times');
ylabel('\bfy in [m]','FontSize',16,'FontName','Times');

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function [X_hat_minus, CovX] = predict_X(X_k_minus_1, P, F, Q)
% in this function we predict the values of X(k) given X(k-1)
%
% X_k_minus_1 is the state vector at time k-1, P is the covariance matrix of F,
% F -- is the state transition matrix and Q is the process cov. matrix
% transform the data
X_hat_minus = F * X_k_minus_1;  % F is the state transition matrix X(k) = F*X(k-1);
CovX = F * P * F' + Q;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function [X_hat, P] = update_X(X_hat_minus, P, H, y_m, R)
%
% this function uses the measurement data and predicted value of
```

```
% measurement and updates the X_hat_minus and P with Kalman gain
%
Y = H * X_hat_minus;
HP = H * P * H' + R;
K = P * H' / HP;
X_hat = X_hat_minus + K * (y_m - Y);
P = P - K*HP*K';    % update cov

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function test_ekf
%
% this function creates data for tracking and then uses the Extended Kalman
% filter for tracking
%
% To run this program just type   ---     test_ekf    in the command window
% of MATLAB
%
%
% written by Thyagaraju Damarla
%

close all
clear all
deg2rad = pi/180;
 sensor_loc =[ 0 15;
               0 15]; % thesse are the coords of the sensors S1 = [0 0], S2 = [15 15]

pop_cords = [1 -5; 5 -4.185; 10 -3; 15 -0.9; 20 3.1; 25 8.9; 30 15; 35 19.5; ...
    40 22.5; 45 24.5; 50 25];

    x = pop_cords(1,1):0.1:pop_cords(end,1);
    y = interp1(pop_cords(:,1), pop_cords(:,2), x, 'spline');
    pop_cords = [x', y'];
    pop_angles = zeros(size(pop_cords,1), 4);
    for j=1:size(pop_cords,1)
        X = pop_cords(j,:)';
        y = get_doas(X,sensor_loc);
        pop_angles(j,1:2) = y';
    end
    b = randn(size(pop_cords)).*2;
    b = b.*deg2rad;
    pop_angles(:,3:4) = pop_angles(:,1:2) + b;
    tar_loc = zeros(size(pop_cords));
    for j=1:size(pop_angles,1)
        doas = pop_angles(j, 3:4);
        S = get_target_loc(doas, sensor_loc');
        tar_loc(j,:) = S;
    end
    figure
    hold on
    plot(tar_loc(:,1),tar_loc(:,2),'r'), grid on
    plot(pop_cords(:,1),pop_cords(:,2),'b', 'LineWidth', 1.5), grid on
    figure
    arr = 1:size(pop_angles, 1);
    plot(arr, pop_angles(:,1), 'b', arr, pop_angles(:,3), 'b.-', ...
        arr, pop_angles(:,2), 'r', arr, pop_angles(:,4), 'r.-'), grid on

sd = 0.05;
P = diag([0.1 0.1 10 10]); % covariance of F
R = sd^2*eye(2);
F = [1 0 0.1 0;  % Xk = F*Xkm1 + q
    0 1 0 0.1;
    0 0 1 0;
    0 0 0 1];
```

```
Q =  1.0e-03 * [    0.0000         0    0.0050         0;
    0    0.0000         0    0.0050;
    0.0050         0    1.0000         0;
    0    0.0050         0    1.0000];
[nx, mx] = size(F);
ny = size(pop_angles,1); % number of measurements
MM = zeros(nx, ny);
PP = zeros(nx, mx, ny);
X_k_minus_1 = [0;0;0;0];

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% perform UKF algorithm
for tk = 1:ny,
    [X_hat_minus, CovX] = predict_X(X_k_minus_1, P, F, Q);
    y_measurement = pop_angles(tk, 3:4)';
    [X_hat, P] = update_X(X_hat_minus, CovX, y_measurement, R, sensor_loc);
    MM(:,tk) = X_hat;
    X_k_minus_1 = X_hat; % for the next cycle
    PP(:,:,tk) = P;
end
figure
h2 = plot(tar_loc(:,1),tar_loc(:,2),'r', pop_cords(:,1),pop_cords(:,2),'g', ...
MM(1,:), MM(2,:), 'k.-', 'LineWidth', 1.5), grid on;
legend(h2, 'measured', 'ground truth', 'filtered', 'Location', 'SouthEast');
xlabel('\bfx in [m]','FontSize',16,'FontName','Times');
ylabel('\bfy in [m]','FontSize',16,'FontName','Times');

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function [X_hat_minus, CovX] = predict_X(X_k_minus_1, P, F, Q)
% in this function we predict the values of X(k) given X(k-1)
%
% X_k_minus_1 is the state vector at time k-1, P is the covariance matrix of F,
% F -- is the state transition matrix and Q is the process noise cov. matrix
% transform the data
X_hat_minus = F * X_k_minus_1;  % F is the state transition matrix X(k) = F*X(k-1);
CovX = F * P * F' + Q;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function [X_hat, P] = update_X(X_hat_minus, P, y_m, R, sensor_loc)
%
% this function uses the measurement data and predicted value of
% measurement and updates the X_hat_minus and P with Kalman gain
%
% Y = H * X_hat_minus;
H = get_jacobian(X_hat_minus(1:2), sensor_loc);
Y = get_doas(X_hat_minus(1:2), sensor_loc);

HP = H * P * H' + R;
K = P * H' / HP;
X_hat = X_hat_minus + K * (y_m - Y);
P = P - K*HP*K';    % update cov

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function H = get_jacobian(X_hat_minus, sensor_loc)
% this function transforms the X_k to Y using the first order approximation
%
% H = [ (x - x1)/norm(X, S1)^2  - (y-y1)/norm(X, S1)^2 0 0;
%       (x - x2)/norm(X, S2)^2  - (y-y1)/norm(X, S2)^2 0 0];
%
S1 = sensor_loc(:,1);
S2 = sensor_loc(:,2);
X = X_hat_minus(1:2);
d1 = vdist(X', S1');
if d1 == 0
    d1 = 0.1;
end
d2 = vdist(X', S2');
```

```
if d2 == 0
    d2 = 0.1;
end
H = [-(X(2) - S1(2))/d1^2  (X(1) - S1(1))/d1^2    0  0;
     -(X(2) - S2(2))/d2^2  (X(1) - S2(1))/d2^2    0  0];

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function Y = get_doas(x,s)
Y = zeros(size(s,2),size(x,2));

for i=1:size(s,2)
    h = atan2(x(2,:)-s(2,i),x(1,:)-s(1,i));
    np = find(h>0.5*pi);
    nn = find(h<-0.5*pi);
    if length(nn)>length(np)
        h(np) = h(np)-2*pi;
    else
        h(nn) = h(nn)+2*pi;
    end
    Y(i,:) = h;
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function S = get_target_loc(doas, sensor_loc)
doas = doas(:);
V = [cos(doas) sin(doas)];
VP = [-sin(doas) cos(doas)];
VP_I = pinv(VP);
SVP = diag(sensor_loc * VP');
S = SVP' * VP_I';   %this is the target location

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function r = vdist(v1, v2)
%
% r = vdist(v1, v2)
% make sure v1 is a matrix of [n, m] and v2 is a matrix of [1, m]

[n, m] = size(v1);

v2 = v2(:);
y = v2';
for i=1:n,
    x = v1(i,:);
    r(i) = norm(x-y);
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%   END OF EKF
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function test_ukf

%
% this function creates data for tracking and then uses the Kalman filter
% for tracking
%
% To run this program just type   ---     test_ukf    in the command window
% of MATLAB
%
%
% written by Thyagaraju Damarla
```

```
%

close all
clear all
deg2rad = pi/180;
sensor_loc =[ 0 15;
              0 15];

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%     Create the data for tracking

pop_cords = [1 -5; 5 -4.185; 10 -3; 15 -0.9; 20 3.1; 25 8.9; 30 15; 35 19.5; ...
    40 22.5; 45 24.5; 50 25];
dt = 0.1;

x = pop_cords(1,1):dt:pop_cords(end,1);
y = interp1(pop_cords(:,1), pop_cords(:,2), x, 'spline');
pop_cords = [x', y'];
pop_angles = zeros(size(pop_cords,1), 4);
for j=1:size(pop_cords,1)
    X = pop_cords(j,:)';
    y = get_doas(X,sensor_loc);
    %         y = comp_pop_angles(X', sensor_loc');
    %         y = y.*(pi/180);
    pop_angles(j,1:2) = y';
end
b = randn(size(pop_cords)).*2;
b = b.*deg2rad;
pop_angles(:,3:4) = pop_angles(:,1:2) + b;
tar_loc = zeros(size(pop_cords));
for j=1:size(pop_angles,1)
    doas = pop_angles(j, 3:4);
    S = get_target_loc(doas, sensor_loc');
    tar_loc(j,:) = S;
end
figure
hold on
plot(tar_loc(:,1),tar_loc(:,2),'r'), grid on
plot(pop_cords(:,1),pop_cords(:,2),'b', 'LineWidth', 1.5), grid on

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% now we design the UKF filter for tracking using bearing estimations only

sd = 0.05;
M = [0;0;0;0];
P = diag([0.1 0.1 10 10]); % covariance of F
R = sd^2*eye(2); % measurement noise
F = [1 0 0.1 0;  % Xk = F*Xkm1 + q
    0 1 0 0.1;
    0 0 1 0;
    0 0 0 1];
Q =  1.0e-03 * [    0.0000         0    0.0050         0;
    0    0.0000         0    0.0050;
    0.0050         0    1.0000         0;
    0    0.0050         0    1.0000];
[nx, mx] = size(F);
ny = size(tar_loc,1); % number of measurements
MM = zeros(nx, ny);
PP = zeros(nx, mx, ny);
ME = zeros(nx, 1);
X_k_minus_1 = M;

% sigma point parameters
kappa = 3 - nx;
alpha = 1;
beta = 0;
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%  compute the weights for sigma points
[Wm, Wc, C] = gen_weights(nx, alpha, beta, kappa);
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% perform UKF algorithm
for tk = 1:ny,
    [X_hat_minus, CovX] = predict_X(X_k_minus_1, P, F, Q, nx, Wm, Wc, C);
    y_measurement = pop_angles(tk,3:4)';
    [X_hat, P] = update_X(X_hat_minus, CovX, y_measurement, R, ...
             sensor_loc, Wm, Wc, C, nx);
    MM(:,tk) = X_hat;
    X_k_minus_1 = X_hat; % for the next cycle
    PP(:,:,tk) = P;
end
figure(2)
plot(pop_cords(:,1),pop_cords(:,2),'k', 'LineWidth', 1.5), grid on
hold on
plot(MM(1,:), MM(2,:), 'r', 'LineWidth', 1.5), grid on
plot(sensor_loc(1,:), sensor_loc(2,:), 'mp', 'LineWidth', 3), grid on
legend('ground truth', 'UKF track', 'Sensor Loc', 'Location', 'SouthEast');

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function [mu, CovX] = predict_X(X_k_minus_1, P, F, Q, nx, Wm, Wc, C)
% in this function we predict the values of X(k) given X(k-1)
%
% X_k_minus_1 is the state vector at time k-1, P is the covariance matrix of F,
% F -- is the state transition matrix and Q is the process cov. matrix
% first compute the sigma points using lemma 4\,in the Sarkka paper
% X = [m   ...  m] + sqrt(C)[0 sqrt(P) -sqrt(P)]
A = chol(P)';
B = [zeros(nx, 1) A -A];
X = sqrt(C)*B + repmat(X_k_minus_1, 1, 2*nx+1);
% transform the data
X_hat_minus = F * X;  % F is the state transition matrix X(k) = F*X(k-1);
mu = zeros(nx, 1);
CovX = zeros(nx, nx);
% ccompute mean of sigma points
num = 2*nx+1;
for j = 1:num,
    mu = mu + Wm(j) * X_hat_minus(:,j);
end
% compute the covariance
for j = 1:num,
    CovX = CovX + Wc(j) * (X_hat_minus(:,j) - mu) * (X_hat_minus(:,j) - mu)';
end
CovX = CovX +  Q; % this is the estimate of cov. of Y

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function [X_hat, P] = update_X(X_hat_minus, P, y_m, R, sensor_loc, Wm, Wc, C, nx)
%
% this function uses the measurement data and predicted value of
% measurement and updates the X_hat_minus and P with Kalman gain
%
deg2rad = pi/180;
num = 2*nx+1;
A = chol(P)';
B = [zeros(nx, 1) A -A];
X = sqrt(C)*B + repmat(X_hat_minus, 1, num);
% now transform X in~to Y
Y = get_doas(X,sensor_loc);
ny = size(Y,1);
mu = zeros(ny, 1);
CovY = zeros(ny, ny);
CovXY = zeros(nx, ny);
for j = 1:num,
    mu = mu + Wm(j) * Y(:,j);
```

```
end
for j = 1:num,
    CovY = CovY + Wc(j) * (Y(:,j) - mu) * (Y(:,j) - mu)';
    CovXY = CovXY + Wc(j) * (X(:,j) - X_hat_minus) * (Y(:,j) - mu)';
end
CovY = CovY + R;
% compute the Kalman gain
K = CovXY / CovY;
% update the
X_hat = X_hat_minus + K * (y_m - mu);
P = P - K * CovY * K';

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function [Wm, Wc, C] = gen_weights(nx, alpha, beta, kappa)
% here we compute the weights for sigma points

% nx -- num. elements in state vector X
% alpha, beta & kappa some parameters to control the position of sigma
% points
% lambda = alpha^2 * (nx + kappa) - nx
% Wm(0) = lambda/(nx + lambda)
% Wc(0) = lambda/((nx + lambda) + (1 - alpha^2 + beta))
% Wm(i) = 1/((2*(nx+lambda)), i = 1,...,2*nx
% Wc(i) = 1/((2*(nx+lambda)), i = 1,...,2*nx

Wm = zeros(2*nx+1,1);
Wc = Wm;
lambda = alpha^2 * (nx + kappa) - nx;
Wm(1) = lambda/(nx + lambda);
Wc(1) = lambda/((nx + lambda) + (1 - alpha^2 + beta));
Wm(2:end) = 1/(2*(nx+lambda));
Wc(2:end) = 1/(2*(nx+lambda));
C = nx + lambda;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function Y = bot_h(x,s)
Y = zeros(size(s,2),size(x,2));

for i=1:size(s,2)
    h = atan2(x(2,:)-s(2,i),x(1,:)-s(1,i));
    np = find(h>0.5*pi);
    nn = find(h<-0.5*pi);
    if length(nn)>length(np)
        h(np) = h(np)-2*pi;
    else
        h(nn) = h(nn)+2*pi;
    end
    Y(i,:) = h;
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function Y = get_doas(x,s)
Y = zeros(size(s,2),size(x,2));

for i=1:size(s,2)
    h = atan2(x(2,:)-s(2,i),x(1,:)-s(1,i));
    np = find(h>0.5*pi);
    nn = find(h<-0.5*pi);
    if length(nn)>length(np)
        h(np) = h(np)-2*pi;
    else
        h(nn) = h(nn)+2*pi;
    end
    Y(i,:) = h;
end
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function S = get_target_loc(doas, sensor_loc)
doas = doas(:);
V = [cos(doas) sin(doas)];
VP = [-sin(doas) cos(doas)];
VP_I = pinv(VP);
SVP = diag(sensor_loc * VP');
S = SVP' * VP_I';   %this is the target location

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%      END OF UKF
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function test_particle_filter

%
% this function creates data for tracking and then uses the Kalman filter
% for tracking
%
% To run this program just type   ---     test_particle_filter    in the command window
% of MATLAB
%
%
% written by Thyagaraju Damarla
%

close all
clear all

% now generate the tracking example
 sensor_loc =[ 0 15;
               0 15];
% Create a bit curved trajectory and angle
% measurements from two sensors

[pop_angles, pop_cords, tar_loc] = gen_tracking_ex(sensor_loc);

% now start designing the particle filter for tracking
sd = 0.002;
P = diag([0.5 0.5 2 2]); % covariance of F
R = sd^2*eye(2);
R = [0.001 0.00; 0.00 0.001];
F = [1  0  0.1 0;  % Xk = F*Xkm1 + q
     0  1  0   0.1;
     0  0  1   0;
     0  0  0   1];
Q =  1.0e-03 * [    0.0000         0              0.0050         0;
                    0              0.0000         0              0.0050;
                    0.0050         0              1.0000         0;
                    0              0.0050         0              1.0000];
[nx, mx] = size(F);
ny = size(tar_loc,1); % number of measurements
X_k_minus_1 = [0;0;0;0];
np = 200; % num. of particles
Nt = np/2;
wt = (1/np) * ones(1, np); % these are the initial weights
filt_dat = [];
for tk = 1:ny,
    if tk == 1
        [X] = generate_particles(X_k_minus_1, P, F, np);
    else
```

```
        [X] = update_particles(X(1:2,:), P);
    end
    Y = get_doas(X(1:2,:),sensor_loc);
    y_meas = pop_angles(tk, 3:4)';
    % now modify the weights based on the p_yk_given_xk
    wk = modify_weights(Y, y_meas, wt, R);
    wk = wk./sum(wk);
    Neff = 1/sum(wk.^2);
    Xm = X;
    if Neff < Nt
        disp('resample');
        wkm = wk;
        [X, wk] = resample(Xm, wkm);
    end
    wt = wk;
    x1 = sum(X(1,:).*wk);
    y1 = sum(X(2,:).*wk);
    dx = x1 - X_k_minus_1(1);
    dy = y1 - X_k_minus_1(2);
    X_k_minus_1 = [x1 y1 dx dy]';
    %     figure(1)
    %     hold on
    %     plot(x1, y1, 'go'), grid on
    filt_dat = [filt_dat; [x1 y1]];
end
figure
h2 = plot(pop_cords(:, 1), pop_cords(:,2), 'k', ...
    filt_dat(:, 1), filt_dat(:,2), 'r', 'LineWidth', 2);
legend(h2, 'ground truth', 'filtered', 'Location', 'SouthEast');
xlabel('\bfx in [m]','FontSize',16,'FontName','Times');
ylabel('\bfy in [m]','FontSize',16,'FontName','Times');

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function [X, wkm] = resample(Xm, wk)

[n, m] = size(Xm);
edges = min([0 cumsum(wk)],1);
edges(end) = 1;
u1 = rand/m;
% this returns the interval where the sample is to be found
[~, idx] = histc(u1:1/m:1, edges);
X = Xm(:,idx);
wkm = repmat(1/m, 1, m);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function [X] = update_particles(X_tm1, P)
[n, m] = size(X_tm1);
X = zeros(n, m);
for i = 1:m
    for j = 1:n,
        X(j, i) = X_tm1(j,i) + normrnd(0, P(j,j), 1);
    end
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function wk = modify_weights(Y, y_meas, wt, R)
% here we determine the weights based on the measurements
[n, m] = size(Y);
Y1 = Y - repmat(y_meas, [1, 200]);
M = zeros(1, n);
Z = mvnpdf(Y1', M, R);
wk = wt.*Z';

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function [X] = generate_particles(Xin, P, F, np)
% Xin = [x y vx vy]^T is the state vector
nx = numel(Xin);
```

```
X = zeros(nx, np);
Y = zeros(nx, np);
% next we generate random distributions around the individual components
% for i = 1:nx,
%     mu = Xin(i);
%     sigma = P(i,i);
%     Y(i,:) = normrnd(mu, sigma, 1, np);
% end
Y = mvnrnd(Xin, P, np);
% use the transition matrix to generate the possible condidates (particles
% for time k+1;
% for i = 1:np,
%     X(:,i) = F * Y(:,i);
% end
X = F*Y';
figure(2)
hold off
plot(X(1,:), X(2,:), 'k*'), grid on
hold on
plot(Xin(1), Xin(2), 'rp', 'LineWidth', 1.5);
hold off

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function Y = get_doas(x,s)
Y = zeros(size(s,2),size(x,2));

for i=1:size(s,2)
    h = atan2(x(2,:)-s(2,i),x(1,:)-s(1,i));
    np = find(h>0.5*pi);
    nn = find(h<-0.5*pi);
    if length(nn)>length(np)
        h(np) = h(np)-2*pi;
    else
        h(nn) = h(nn)+2*pi;
    end
    Y(i,:) = h;
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function S = get_target_loc(doas, sensor_loc)
doas = doas(:);
V = [cos(doas) sin(doas)];
VP = [-sin(doas) cos(doas)];
VP_I = pinv(VP);
SVP = diag(sensor_loc * VP');
S = SVP' * VP_I';   %this is the target location

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function [pop_angles, pop_cords, tar_loc] = gen_tracking_ex(sensor_loc)
deg2rad = pi/180;
pop_cords = [1 -5; 5 -4.185; 10 -3; 15 -0.9; 20 3.1; 25 8.9; 30 15; 35 19.5; ...
    40 22.5; 45 24.5; 50 25];

    x = pop_cords(1,1):0.1:pop_cords(end,1);
    y = interp1(pop_cords(:,1), pop_cords(:,2), x, 'spline');
    pop_cords = [x', y'];
    pop_angles = zeros(size(pop_cords,1), 4);
    for j=1:size(pop_cords,1)
        X = pop_cords(j,:)';
        y = get_doas(X,sensor_loc);
        pop_angles(j,1:2) = y';
    end
    b = randn(size(pop_cords)).*2;
    b = b.*deg2rad;
    pop_angles(:,3:4) = pop_angles(:,1:2) + b;
    tar_loc = zeros(size(pop_cords));
```

```
for j=1:size(pop_angles,1)
    doas = pop_angles(j, 3:4);
    S = get_target_loc(doas, sensor_loc');
    tar_loc(j,:) = S;
end
figure
hold on
plot(tar_loc(:,1),tar_loc(:,2),'r'), grid on
plot(pop_cords(:,1),pop_cords(:,2),'b', 'LineWidth', 1.5), grid on
```

Chapter 9
Localization of Transient Events

In battlefields, each side is often engaging in shelling targets with a wide variety of weapons: cannons, rockets, rocket-propelled grenades, tanks, etc., as shown in Fig. 9.1. These weapons are fired, for most part, from far distance (distances greater than 2–3 km). Firing a weapon involves an explosion of gunpowder for most conventional systems. Once the weapon is fired, the shell travels through the air and detonates on or just prior to contact with the ground or the target. Both the firing and detonation of the shells result in explosions that produce short-lived (transient) sound bursts. Similarly, when a gun is fired, a loud sound is emitted due to muzzle blast. This chapter presents the methods and techniques used to detect such transient events and localize them, that is, find the location of the explosions. Different types of weapon's fire require different levels and types of detection and localization. For instance, in the case of mortars, one must detect both the launch and detonation. Detecting and localizing the launch determines where the fire is coming from and detecting and locating the detonation helps in assessing the damage done and mobilizing the rescue personnel to the site. In the case of rifle fire, which produces two transient signals (shockwave and muzzle-blast), it is important to know the direction and location of the source of gunfire, called sniper localization, so one can either avoid or neutralize the shooter.

9.1 Detection of Transient Events

A typical supersonic rifle signature is shown in Fig. 9.2. The figure shows the three transient signals: (a) the N-wave due to the supersonic crack (shockwave), (b) a ground reflected supersonic crack, and (c) the signal due to muzzle blast. In the case of rifle fire, both the shockwave and muzzle blast signals can be detected and their time of arrivals are estimated to localize the gunfire. Detection of the N-wave due to the shockwave is relatively easy as it is quite large and the SNR is not a problem. However, the muzzle blast signal is quite often small, as its amplitude decreases with

T. Damarla, *Battlefield Acoustics*, DOI 10.1007/978-3-319-16036-8_9

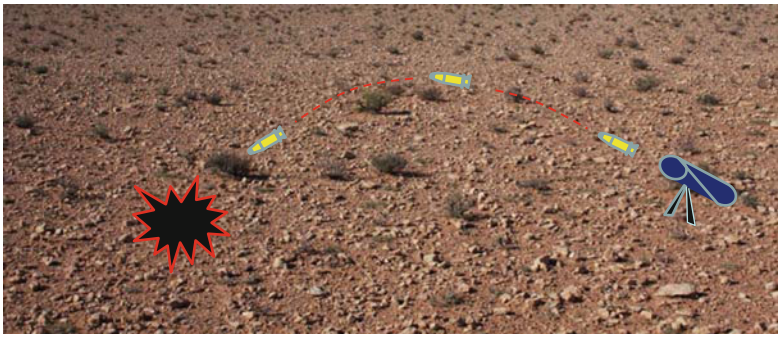


Fig. 9.1 Mortar launch and detonation

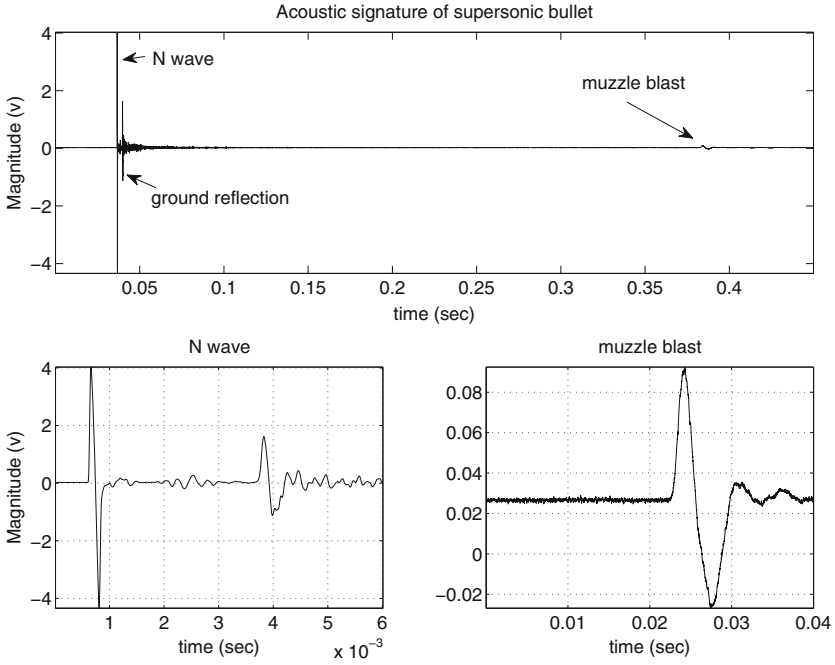


Fig. 9.2 Waveform generated by a supersonic bullet

distance. At distances above 200 m, the muzzle blast signal is barely visible above the noise floor (see Fig. 9.2). Some of the techniques described in Chap. 3 can be used in detecting the muzzle blast signal where the noise and signal statistics are used to estimate the likelihood ratio.

9.2 Estimation of Time Delay

For localization of events, one of the most widely used technique is localization using time difference of arrival (TDOA). This relates to estimating the time difference between two signals arriving at two spatially separated sensors, as shown in Fig. 9.3. The signals can be modeled as

$$\begin{aligned} x_i(t) &= s(t) + n_i(t) \\ x_j(t) &= \alpha s(t+\delta) + n_j(t) \end{aligned} \tag{9.1}$$

where $s(t)$ is the signal transmitted by the source, α the scaling factor, 'δ' is the time delay, and $n_i(t)$ and $n_j(t)$ are the random noises assumed uncorrelated. It is also assumed that the signal $s(t)$ is uncorrelated with the noise $n_i(t)$ and $n_j(t)$. Estimation of the delay δ is the subject of this section. A simple technique is to compute the cross correlation $R_{x_i x_j}$ as discussed in Sect. 2.4 of Chap. 2:

$$R_{x_i x_j}(\tau) = E\left[x_i(t) x_j(t-\tau)\right], \tag{9.2}$$

where E denotes expectation. The argument τ that maximizes (9.2) provides an estimate of delay between two signals $x_i(t)$ and $x_j(t)$. A simple example is presented in Fig. 9.4, where signal $x_i(t)$ and $x_j(t)$ are two pulses plus noise. The signal length is 512 samples long. The delay between the two signals is 30 samples long. $R_{x_i x_j}(\tau)$ in Fig. 9.4 peaks at 482. The time delay is $512 - 482 = 30$. The expectation in (9.2) is meant to be done over an infinitely long signals; however, the signal observation times are short, and hence, $R_{x_i x_j}(\tau)$ can only be estimated. For ergodic processes, an estimate of the cross correlation is given by

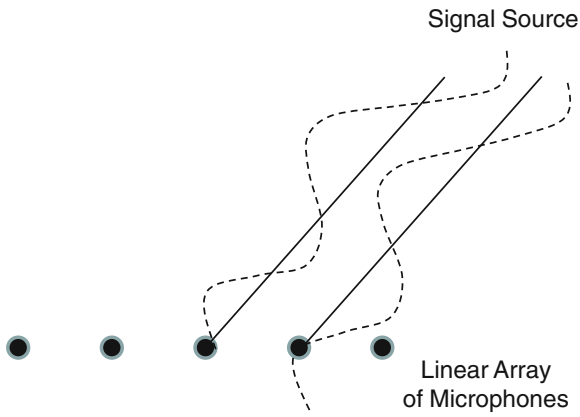


Fig. 9.3 Signal arriving at two different sensors

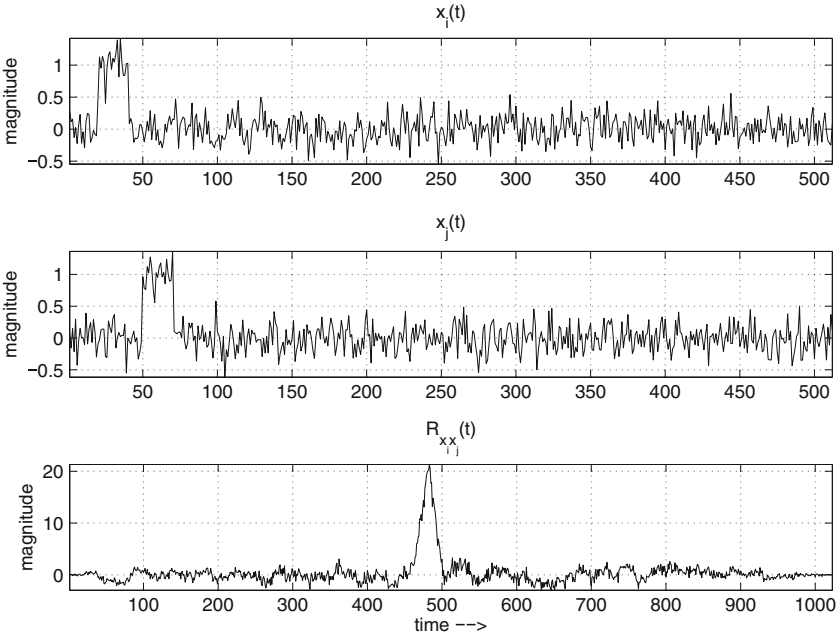


Fig. 9.4 Signal arriving at two different sensors and their cross correlation

$$\hat{R}_{x_i x_j}(\tau) = \frac{1}{T-\tau}\int_{\tau}^{T} x_i(t)x_j(t-\tau)dt, \tag{9.3}$$

where T represents the observation interval.

For better localization of transient or sound sources, it is important that the delay δ is accurately estimated. The accuracy is affected by the noise. In order to reduce the effect of the noise, one often filters the signals x_i and x_j. The process used to estimate the time delay with pre-filtering of the data is called the generalized correlation method (GCM).

9.2.1 Generalized Correlation Method

The general process used for estimation of cross correlation is shown as a block diagram in Fig. 9.5. As shown, each signal x_i is filtered through H_i to generate an output y_i with reduced noise. One of the y_i's is delayed and multiplied, and integrated and squared for a range of delays τ and a maximum peak is found that gives an estimate of the time delay δ in (9.1).

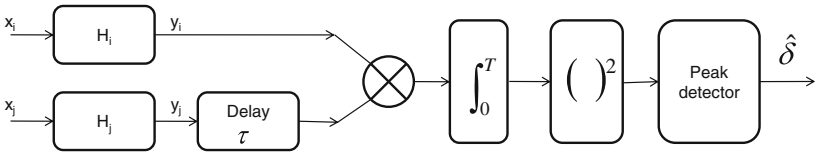


Fig. 9.5 Process used in estimating the delay using GCM

It is well known that the cross correlation $R_{x_i x_j}$ and the cross power spectral density $S_{x_i x_j}$ are related [90] by

$$R_{x_i x_j}(\tau) = \int_{-\infty}^{\infty} S_{x_i x_j}(f)\, e^{j2\pi f \tau}\, df. \tag{9.4}$$

The cross power spectral density $S_{y_i y_j}$ of the filtered outputs is

$$S_{y_i y_j} = H_i(f) H_j^{'}(f) S_{x_i x_j}, \tag{9.5}$$

where $H_j^{'}$ is the complex conjugate of H_j. Then the generalized correlation between x_i and x_j is

$$R_{y_i y_j}(\tau) = \int_{-\infty}^{\infty} \psi(f) S_{x_i x_j}(f)\, e^{j2\pi f \tau}\, df, \tag{9.6}$$

where

$$\psi(f) = H_i(f) H_j^{'}(f) \tag{9.7}$$

which is called the general frequency weighting. Note that $R_{y_i y_j}$ and $S_{x_i x_j}$ are related by Fourier transform. We now use different weighting functions $\psi(f)$ to multiply $S_{x_i x_j}$ and take an inverse Fourier transform to obtain $R_{y_i y_j}$ and estimate the location of the peak to find the time delay. Each technique is used on sample speech data denoted by s_1 and the delayed signal denoted by s_2, as shown in Fig. 9.6. A random Gaussian noise is added to the signals to obtain $\mathfrak{s}_1 = ns_1$ and $\mathfrak{s}_2 = ns_2$. The cross correlations $R_{s_1 s_2}$ and $R_{\mathfrak{s}_1 \mathfrak{s}_2}$ are shown in the Fig. 9.6. The delays are estimated corresponding to their peaks. We now present three different approaches.

Simple Generalized Cross Correlation (GCC): In this case, the fast Fourier transforms of signals $\mathfrak{s}_1$ and $\mathfrak{s}_2$ are taken to obtain the cross power spectral density

$$S_{\mathfrak{s}_1 \mathfrak{s}_2} = F_{\mathfrak{s}_1} F_{\mathfrak{s}_2}^{'}, \tag{9.8}$$

where $F_{\mathfrak{s}_1}$ is the Fourier transform of $\mathfrak{s}_1$. Take the inverse Fourier transform of $S_{\mathfrak{s}_1 \mathfrak{s}_2}$ to obtain $R_{\mathfrak{s}_1 \mathfrak{s}_2}$. The time delay corresponds to the location of the $R_{\mathfrak{s}_1 \mathfrak{s}_2}$ peak.

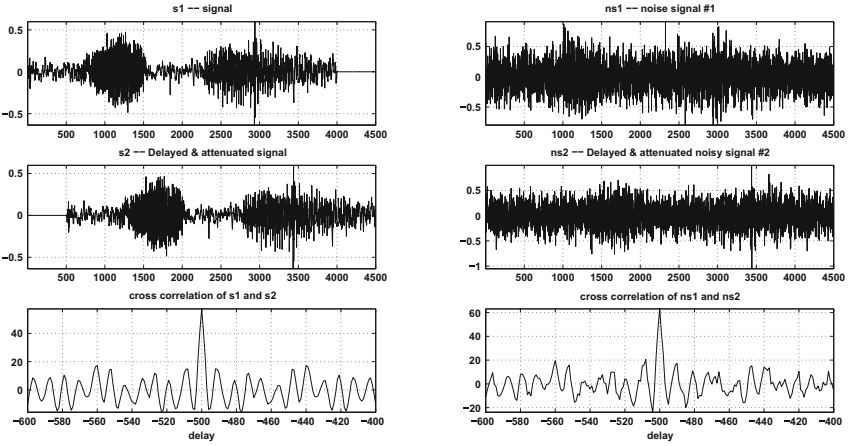


Fig. 9.6 Time delay estimation **a** noise-free case and **b** noisy case

The Phase Transform (PHAT): In order to sharpen the cross correlation peak, one can whiten the input signals by using the appropriate weighting function. In this case, the weighting function used is

$$\psi(f) = \frac{1}{\left|S_{\mathfrak{s}_1\mathfrak{s}_2}\right|} \tag{9.9}$$

The algorithm for estimating time delay using PHAT is given below:

1. Find the cross correlation of the signals

$$R_{\mathfrak{s}_1\mathfrak{s}_2}(\tau) = E\left[\mathfrak{s}_1(t)\mathfrak{s}_2(t-\tau)\right] \tag{9.10}$$

2. Estimate the cross power spectral density of the signals by taking a fast Fourier transform of $R_{\mathfrak{s}_1\mathfrak{s}_2}(\tau)$, that is,

$$S_{\mathfrak{s}_1\mathfrak{s}_2} = FFT\left(R_{\mathfrak{s}_1\mathfrak{s}_2}(\tau)\right). \tag{9.11}$$

3. Divide $S_{\mathfrak{s}_1\mathfrak{s}_2}$ by $\psi = \left|S_{\mathfrak{s}_1\mathfrak{s}_2}\right|$

$$\mathfrak{S}_{\mathfrak{s}_1\mathfrak{s}_2} = \frac{S_{\mathfrak{s}_1\mathfrak{s}_2}}{\left|S_{\mathfrak{s}_1\mathfrak{s}_2}\right|} \tag{9.12}$$

4. Take an inverse Fourier transform of $\mathfrak{S}_{\mathfrak{s}_1\mathfrak{s}_2}$ to get the modified estimate of the $R_{\mathfrak{s}_1\mathfrak{s}_2}(\tau)$, that is,

$$\mathfrak{R}_{\mathfrak{s}_1\mathfrak{s}_2}(\tau) = \mathrm{IFFT}\left(\mathfrak{S}_{\mathfrak{s}_1\mathfrak{s}_2}\right). \tag{9.13}$$

5. Find the delay using the peak detector.

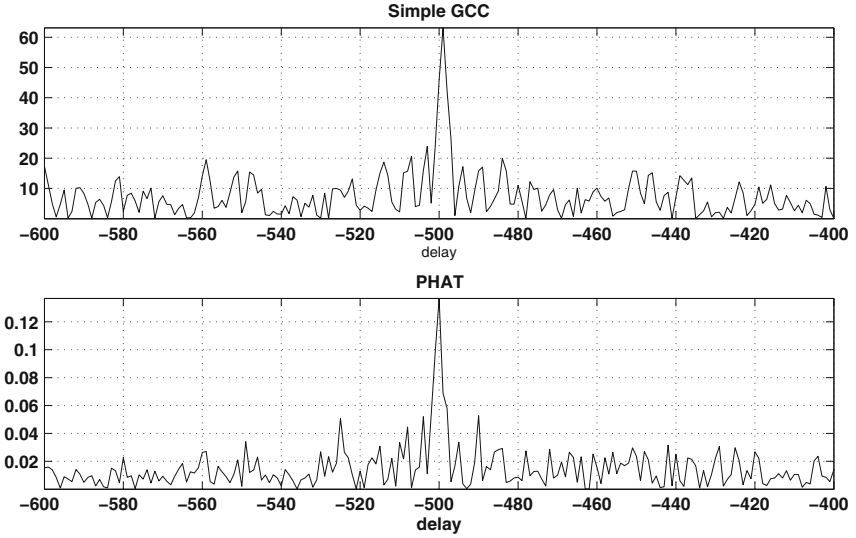


Fig. 9.7 Time delay estimation **a** simple GCC and **b** PHAT

Note from (9.12) only the phase information is preserved. If there is no noise, this technique would result in a delta function centered at the correct delay. In any case, it minimizes the spreading of the peak when noise is present. Figure 9.7 shows the time delay estimation using the simple GCC and PHAT methods.

Maximum Likelihood (ML) Method: This method provides the maximum likelihood solution to the time delay estimation (TDE) problem. The weighting function is selected such that it attenuates the signals that have a low SNR. For uncorrelated signals and noise, the time delay estimate is unbiased and it is efficient if the observation intervals are long. The weighting function is given by

$$\psi(f) = \frac{1}{\left|S_{\mathfrak{s}_1\mathfrak{s}_2}\right|} \frac{\left|\gamma_{\mathfrak{s}_1\mathfrak{s}_2}(f)\right|^2}{1 - \left|\gamma_{\mathfrak{s}_1\mathfrak{s}_2}(f)\right|^2}, \tag{9.14}$$

where

$$\left|\gamma_{\mathfrak{s}_1\mathfrak{s}_2}(f)\right|^2 = \frac{\left|S_{\mathfrak{s}_1\mathfrak{s}_2}\right|^2}{S_{\mathfrak{s}_1\mathfrak{s}_1} S_{\mathfrak{s}_2\mathfrak{s}_2}} \tag{9.15}$$

is the squared coherency magnitude. Since, the weighting function has the term $\gamma(f)^2/1 - \gamma(f)^2$, greater weight is given to the frequencies where coherency is high and de-emphasizes those frequencies with low coherency. The ML process for estimating the time delay weights the cross spectral phase according to the estimated phase when the variance of the estimated phase error is the least. The algorithm for

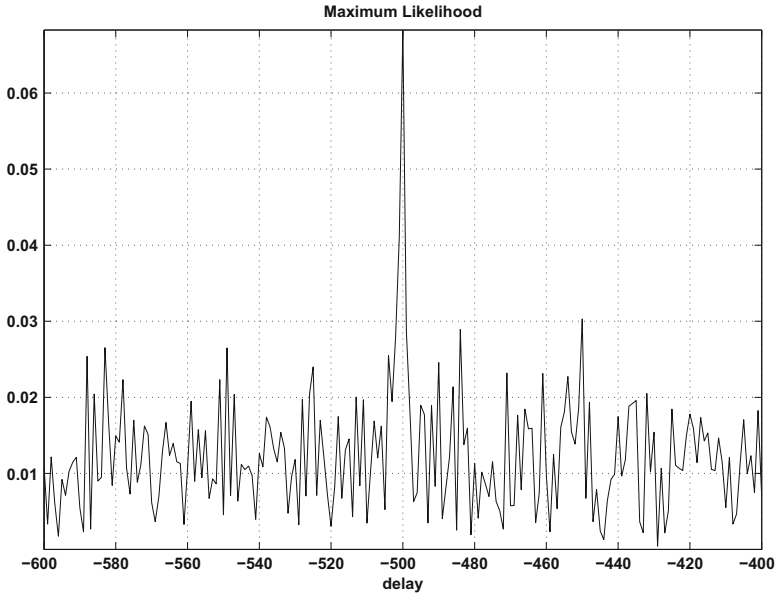


Fig. 9.8 Time delay estimation using maximum likelihood

estimating the time delay using the ML method is similar to the one given for the PHAT method with a weighting function given by (9.14). Figure 9.8 shows the time delay estimation plot using the ML technique. There are other techniques [59, 100] that could be used. If the SNR is high a simple cross correlation may be sufficient. However, if the noise is high, that is, if the SNR is low, then one of the techniques described here can be used to estimate the time delay with reasonable accuracy.

9.3 Sniper Localization

Sniper localization is a study of finding the location of the sniper (shooter) using an acoustic sensor array and the properties of the bullet if it is traveling at supersonic speeds. Sniper localization is a vital part of battlefield acoustics. The enemy may be firing from a location that is in plain sight, from a bunker, through a window in a building, or some other location. It is not always possible to see the sniper. However, by employing acoustic sensor arrays or distributed microphones, one can estimate the location of the sniper with reasonable accuracy.

As mentioned, firearms often use a confined explosive charge to expel mortars, bullets, etc., out of cylindrical barrels to guide their trajectory. The rapidly expanding gases in the confined area result in acoustic blast that travels at the speed of sound 'c'. In the case of guns, this acoustic blast is called "muzzle blast." Often, guns use

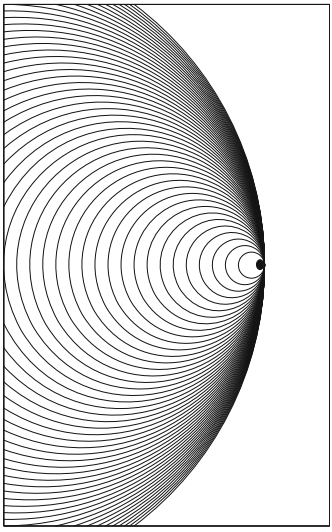


Fig. 9.9 Wavefront generated by a bullet

silencers to muffle the sound so those types of firearms may not be a good source for detection and analysis.

Unlike the mortars, bullets often travel at supersonic speeds, that is, speeds greater than the speed of sound. When bullets travel at supersonic speeds, their dynamics are different and the signals generated by the bullets are also different. Let us suppose that the bullet is traveling at speed of sound ($v = c$), then the wavefront generated by the bullet cannot be separated from the bullet as shown in Fig. 9.9.

As the bullet approaches the speed of sound, all the waves pile up and form a large amplitude "sound barrier." For airplanes traveling near or at the speed of sound, the sound barrier makes sustained flight at this speed risky and difficult. It was during World War II that the term "sound barrier" or "sonic barrier" came into existence as fighter pilots noticed tremendous drag when they dove at high speeds approaching the speed of sound [1]. When the bullet travels at speeds greater than the speed of sound ($v > c$), the wave fronts lag behind the source in a cone-shaped region as shown in Fig. 9.10. The surface of the cone forms a supersonic wave front with a high amplitude called "shockwave." When the shockwave reaches an observer, a sonic boom is heard. In the case of a bullet, the sonic boom sounds like the crack of a whip.

It is clear from the previous discussion that when an object travels at speeds greater than the speed of sound, a shockwave is generated and it expands as a cone behind the bullet, with a wave front traveling normal to the surface of the cone at velocity 'c' as shown in Fig. 9.11. The shockwave cone has an inner angle

$$\theta_M = \arcsin\left(\frac{1}{M}\right) \tag{9.16}$$

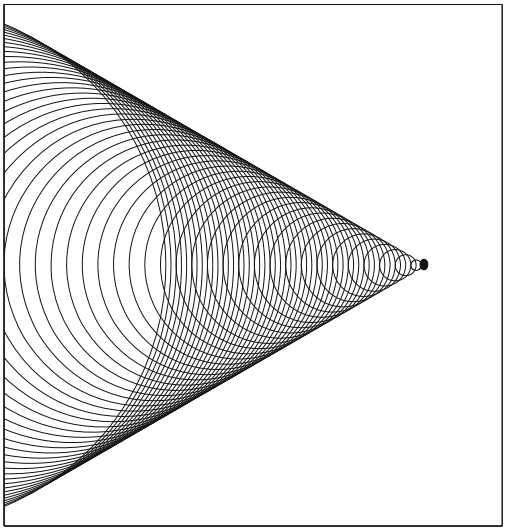


Fig. 9.10 Wavefront generated by a supersonic bullet

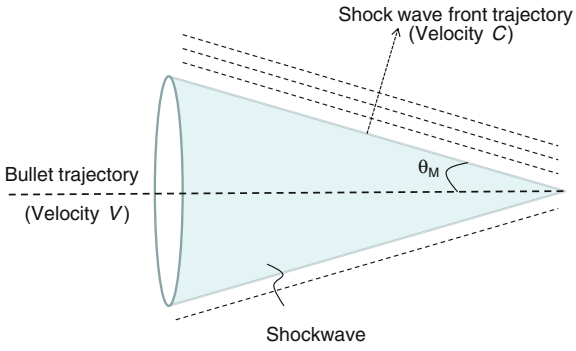


Fig. 9.11 Supersonic bullet wave front propagation

where M is the Mach number[1]

$$M = \frac{v}{c} \tag{9.17}$$

'v' is the speed of bullet. A typical signal generated by a supersonic bullet is shown in Fig. 9.12. The signature has three distinct features, namely, (a) a large N-wave due to shockwave, (b) a signal due to ground reflection of shockwave, and (c) a signal due to the muzzle blast. We also note that the signal strengths are different for each signal. The signal strength of the reflected shockwave depends on the ground absorption coefficient. The muzzle blast signal strength is $\propto \frac{P_0}{d}$, where P_0 is the signal strength at the origin (at the gun) and d is the distance to the microphone.

[1] In honor of Ernst Mach (1838-1916), the Moravian physicist, psychologist, and philosopher who studied sound and ballistics.

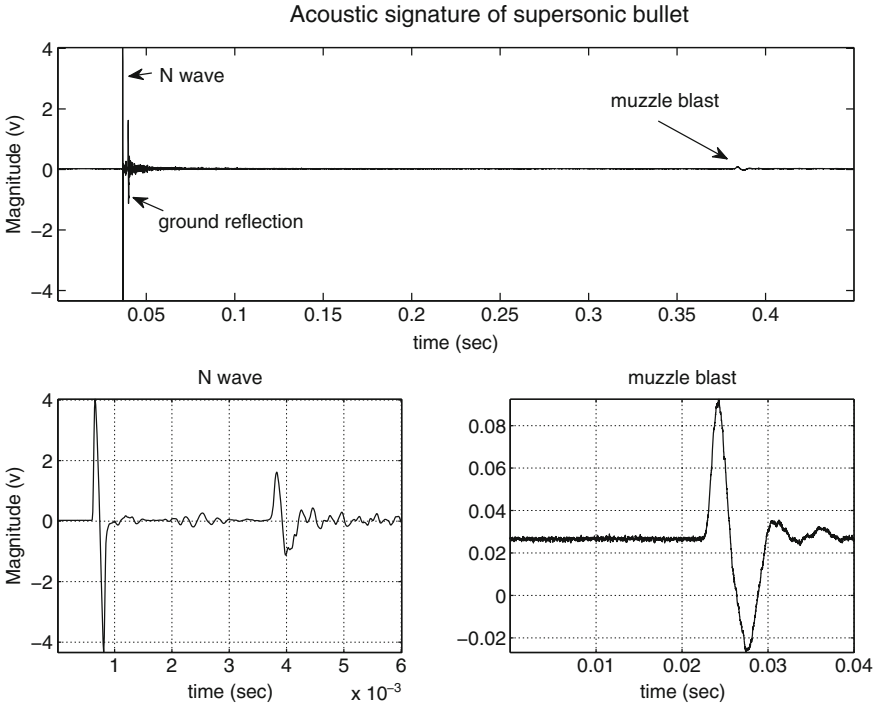


Fig. 9.12 Waveform generated by a supersonic bullet

***Nonlinear distortion of ballistic shockwave in the far-field*:**

The shockwave manifests as an N-shaped waveform as shown in Fig. 9.13. However, it distorts as it propagates as shown in Fig. 9.14. The amplitude decreases and at the same time the N-wave duration increases.

The far-field models of shockwave show that the overpressure decreases proportionally to the miss distance according to $r^{-3/4}$, and the N-wave duration increases proportionally to the miss distance as $r^{1/4}$ [37, 97]. Whitham [97] has shown that the overpressure P_{fs} (peak pressure amplitude) varies according to

$$p_{max} = \frac{0.53 P_0 \left(M^2 - 1\right)^{1/8}}{r^{3/4}} \frac{d}{l^{1/4}} \tag{9.18}$$

$$T = \frac{1.82\ M\ r^{1/4}}{c \left(M^2 - 1\right)^{3/8}} \frac{d}{l^{1/4}} \tag{9.19}$$

where l and d denote the length and diameter of the bullet, respectively, r the miss distance, M mach number, c speed of sound and P_0 the ambient pressure.

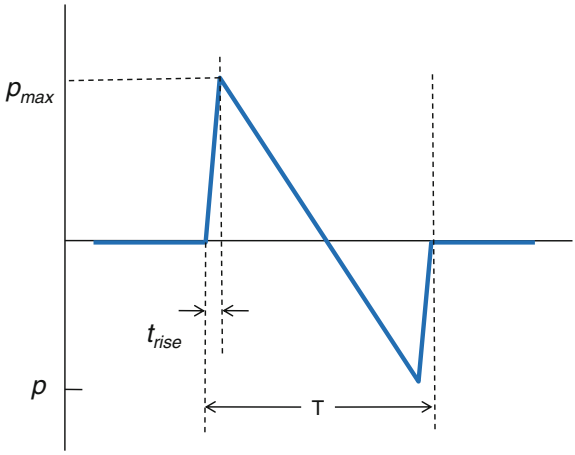


Fig. 9.13 Ideal shockwave with raise time

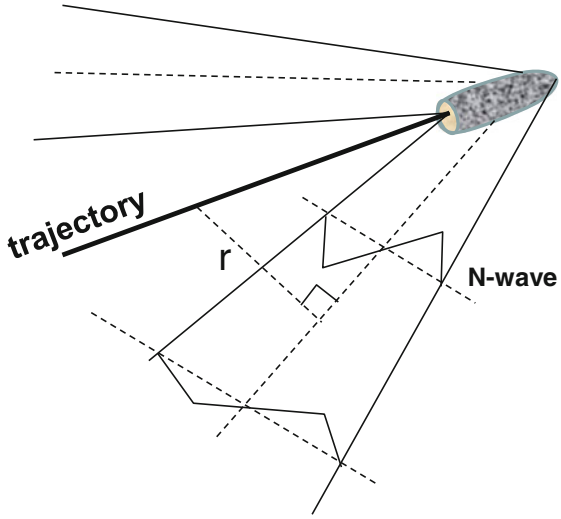


Fig. 9.14 Distortion of shockwave with miss distance *r*

In order to find the location of a sniper (shooter), several acoustic sensors are used. These acoustic sensors could be arrays or distributed microphones. In this chapter, we investigate both the cases.

There are several commercial sniper localization systems by various vendors [36, 88]. Vanderbilt University developed a soldier-wearable shooter localization system [95]. The commercial systems employ an array of microphones in some geometrical configurations, for example, (a) circular array, (b) tetrahedral [36, 74] array, etc., at each location to determine the angle of arrival (AoA) of the muzzle blast and the shockwave. If the Mach number of the bullet is known, one can determine the trajectory of the bullet using the AoA of the shockwave. With the knowledge

of the difference between the muzzle blast and shockwave arrival times, it is easy to determine the sniper location [36, 95]. In the following sections, some of the techniques are presented.

9.3.1 Localization Using Time of Arrival (TOA)

The time of arrival (TOA) is defined as the time it takes a signal (such as a gun-shot) to arrive at a sensor (such as a microphone). Let the instant the gun's muzzle blast is emitted be t_0 and the time the signal arrived at the sensor S_i be t_i, then the distance the sound traveled is

$$r_i = (t_i - t_0)\, c, \tag{9.20}$$

where c denotes the speed of sound. Since the both the times t_0 and t_i are known, the distance r_i can be computed using (9.20). If there are at least three sensors able to detect the sound signal, then these distances $r_i,\ \in \{r_1, r_2, \ldots, r_n\}$, and $n \geq 3$ can be used to determine the location of the gunfire. Each distance r_i describes a sphere of radius r_i. It takes three spheres to intersect at a point (the intersection of two spheres result in a circle) giving the location of the gunshot as shown in Fig. 9.15.

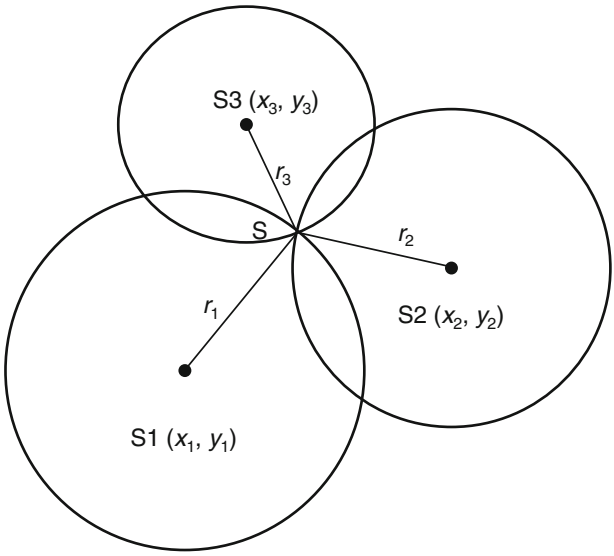


Fig. 9.15 Localization using TOAs

In the case of finding the location on a plane, the spheres are replaced by circles. Let the location of the sound source (gunshot) be $S = (x, y)$, then the squared distances are

$$\begin{aligned} r_1^2 &= (x_1 - x)^2 + (y_1 - y)^2 = x^2 + y^2 + x_1^2 + y_1^2 - 2xx_1 - 2yy_1 \\ r_2^2 &= (x_2 - x)^2 + (y_2 - y)^2 = x^2 + y^2 + x_2^2 + y_2^2 - 2xx_2 - 2yy_2 \\ r_3^2 &= (x_3 - x)^2 + (y_3 - y)^2 = x^2 + y^2 + x_3^2 + y_3^2 - 2xx_3 - 2yy_3 \end{aligned} \tag{9.21}$$

Subtracting r_1^2 from r_2^2 gives

$$r_2^2 - r_1^2 = \left(x_2^2 - x_1^2\right) + \left(y_2^2 - y_1^2\right) - 2x\,(x_2 - x_1) - 2y\,(y_2 - y_1)\,.$$

Similarly, subtracting r_1^2 from r_i^2, $\forall i > 1$ and rearranging them in a matrix form gives

$$\begin{bmatrix} (x_2 - x_1) & (y_2 - y_1) \\ (x_3 - x_1) & (y_3 - y_1) \\ \vdots & \vdots \\ (x_n - x_1) & (y_n - y_1) \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \frac{1}{2} \begin{bmatrix} \left(x_2^2 - x_1^2\right) + \left(y_2^2 - y_1^2\right) - r_2^2 + r_1^2 \\ \left(x_3^2 - x_1^2\right) + \left(y_3^2 - y_1^2\right) - r_3^2 + r_1^2 \\ \vdots \\ \left(x_n^2 - x_1^2\right) + \left(y_n^2 - y_1^2\right) - r_n^2 + r_1^2 \end{bmatrix} \tag{9.22}$$

Equation (9.22) can be written as

$$H \begin{bmatrix} x \\ y \end{bmatrix} = B \tag{9.23}$$

This is a linear equation. The least-squares solution can be found as

$$H^T H \begin{bmatrix} x \\ y \end{bmatrix} = H^T B,$$

or

$$\begin{bmatrix} \hat{x} \\ \hat{y} \end{bmatrix} = \left(H^T H\right)^{-1} H^T B, \tag{9.24}$$

where $\left(H^T H\right)^{-1} H^T$ is called the pseudo inverse of H. Note that (9.24) gives the approximate estimates of x and y in the least-squares sense and that is the reason for using $\hat{x}$ instead of x, similarly $\hat{y}$ for y.

To recapitulate, the TOAs are converted to distances, which are then fused along with the known coordinates of the sensors to obtain the location of the sound transmitting source such as a gunfire, mortar launch, siren, etc. The technique is general, in the sense that it can be used to localize any source as long as the distances r_i can be estimated.

Even though the technique described above for localization is elegant, it requires the reference time t_0 corresponding to the beginning of the gunfire. However, to obtain t_0 one should record the signal immediately after the gunfire at its location. In most of the cases, the gunfire is due to hostile activity, hence the location and time when the guns are fired are not known. However, we can still detect and localize the source using TDOA, which is presented in the next section.

9.3.2 Localization Using Time Difference of Arrival (TDOA)

Since t_0 in (9.20) of TOA is hard to estimate, TDOA is the most widely used approach in localization of targets. Using (9.20) the TDOA between two sensors S_i and S_j is given by

$$t_{ij} = (t_i - t_0) - \left(t_j - t_0\right) = t_i - t_j \ , \tag{9.25}$$

and the difference in the distances

$$r_{ij} = (t_i - t_0)\, c - \left(t_j - t_0\right) c = \left(t_i - t_j\right) c = r_i - r_j = t_{ij} c. \tag{9.26}$$

The signal source must lie on the locus, which keeps the difference r_{ij} constant. This locus defines a hyperbola. If, there are at least three sensors, the intersection of hyperbolas gives the location of the signal source as shown in Fig. 9.16. The following formulation gives the procedure to find the point of intersection of hyperbolas [14, 63, 83]. Let $S = (x, y)$ denotes the location of the sound source to be estimated; the locations of the sensors $S_i = (x_i, y_i)$ are known. Without loss of generality, the location of S_1 is set at $(0, 0)$ (just subtract the coordinates of all sensors with the coordinates of S_1 to make $S_1 = (0, 0)$). Now, from Eq. (9.26)

$$\begin{aligned} r_{i1} &= r_i - r_1 \\ \text{or} \quad r_{i1} + r_1 &= r_i = \sqrt{(x_i - x)^2 + (y_i - y)^2} \\ \text{or} \quad (r_{i1} + r_1)^2 &= K_i - 2x_i x - 2y_i y + x^2 + y^2 = K_i - 2x_i x - 2y_i y + r_1^2 \end{aligned} \tag{9.27}$$

where

$$K_i = x_i^2 + y_i^2.$$

Equation (9.27) can be rewritten as

$$x_i x + y_i y = -r_{i1} r_1 + \frac{1}{2}\left(K_i - r_{i1}^2\right) \tag{9.28}$$

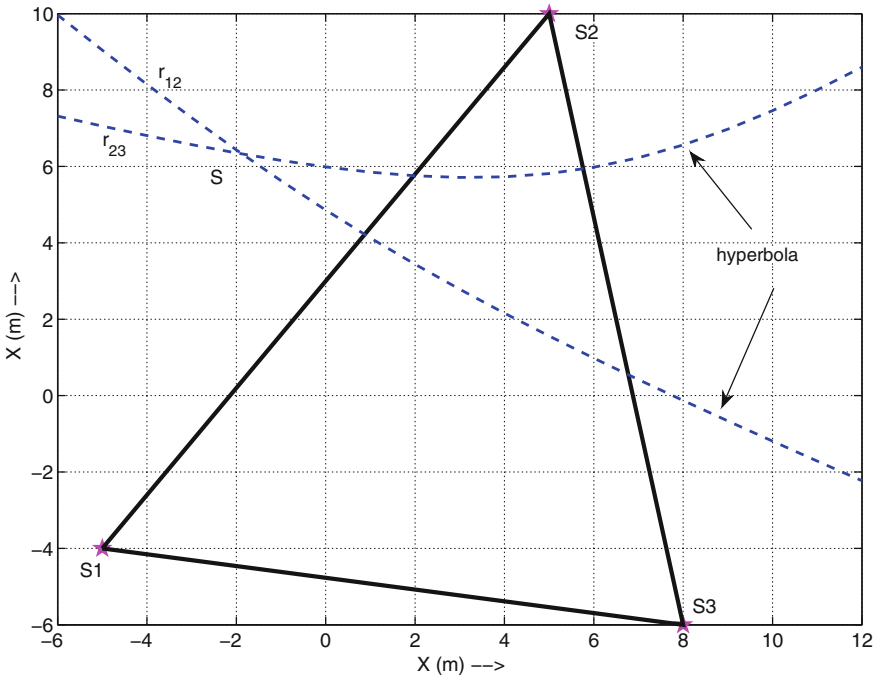


Fig. 9.16 Localization using TDOAs

Explicitly writing for all sensors the above equation becomes

$$\begin{bmatrix} x_2 & y_2 \\ x_3 & y_3 \\ \vdots & \vdots \\ x_n & y_n \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = r_1 \begin{bmatrix} -r_{21} \\ -r_{31} \\ \vdots \\ -r_{n1} \end{bmatrix} + \frac{1}{2} \begin{bmatrix} K_2 - r_{21}^2 \\ K_3 - r_{31}^2 \\ \vdots \\ K_n - r_{n1}^2 \end{bmatrix} \tag{9.29}$$

This is in the form of a linear equation

$$HX = r_1 G + D \tag{9.30}$$

where $X = \begin{bmatrix} x \\ y \end{bmatrix}$. The least-squares solution in terms of r_1 yields

$$\hat{X} = \left(H^T H\right)^{-1} H^T \left(r_1 G + D\right). \tag{9.31}$$

Substituting this intermediate result into

$$r_1^2 = x^2 + y^2,$$

leads to a quadratic equation in r_1. Solving for r_1 and substituting the positive root back into (9.31) yields the final solution for X. This method is called spherical-interpolation [14, 86].

Once, the approximate solution for X is obtained, it may be further improved using the second-order statistics of the TDOA measurement errors.

The above two methods, namely, TOA and TDOA, for localization are quite general and can be applied for locating the origin of any sound source using individual microphones to detect sounds.

9.3.3 Sniper Localization of Subsonic Gunfire Using Acoustic Array

Traditionally, the sniper localization is done using an acoustic array. When the aperture of the array is small, using the TOA or TDOA methods described in previous sections would result in poor accuracy. Hence, it is a common practice to use several arrays with small apertures to estimate the direction of arrival (DOA) angles and triangulate them to find the location of the source. The angles of arrival at different arrays distributed over an area can be triangulated to locate the sniper's position using the techniques described in Sect. 8.1 of Chap. 8.

9.3.4 Sniper Localization of Supersonic Gunfire with a Single Array of Microphones

In the case of the supersonic gunfire, there is a shockwave apart from the muzzle blast, as shown in Fig. 9.12. We assume that the location and the orientation of the sensor array is known. Let us assume that a supersonic gun is fired from a location Z as shown in Fig. 9.17. The bullet travels at a supersonic speed v with a Mach number $M = v/c > 1$, where c is the propagation speed of sound. The bullet's path is marked as "Bullet Trajectory" in the figure. As the bullet travels at supersonic speed, a shockwave cone with cone angle θ radiates from the bullet and the relation between Mach number M and θ is

$$\sin\theta = \frac{1}{M}. \tag{9.32}$$

While the bullet travels at supersonic speed, the shockwave travels perpendicular to the cone at the speed of sound c and reaches an array of microphones denoted by S in Fig. 9.17. The muzzle blast radiates from the gun and travels a distance R from the gun to reach the sensor array S.

In order to find the location, we need the DOA of the muzzle blast and the distance between the sensor location S and sniper location Z. The DOA of the muzzle blast

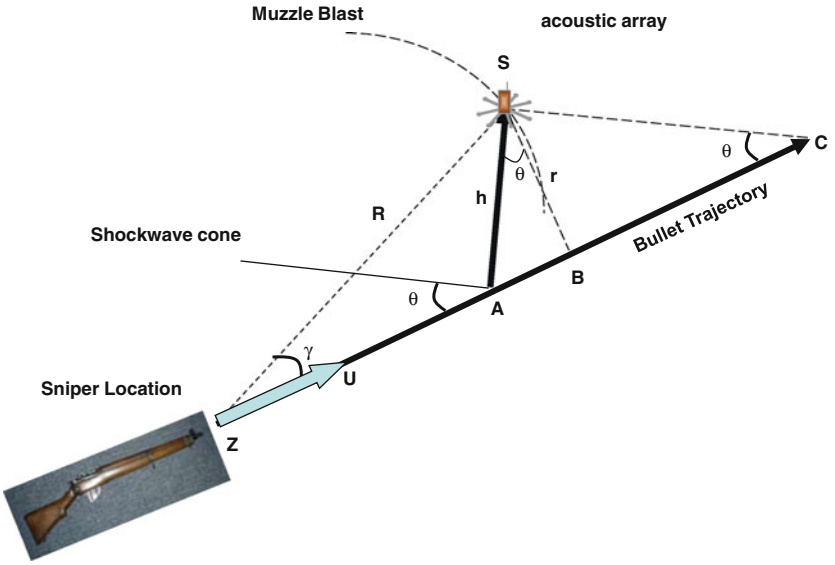


Fig. 9.17 Geometry of a supersonic bullet path and different wave-fronts

can be estimated by processing the signals captured by the sensor array. Now, we show how to estimate the distance $d_{Z,S}$ from the TOA measurements of the muzzle blast and shockwave.

The time of arrival of muzzle blast t_m is the time taken by the muzzle blast to travel from the gun to the sensor at the speed of sound and is given by

$$t_m = t_0 + \frac{d_{Z,S}}{c} = t_0 + \frac{R}{c}, \tag{9.33}$$

where t_0 is the instant of time the gun is fired. Since the shockwave originated at the point A on the bullet's trajectory, the time of arrival of the shockwave has two components: one corresponds to the time the bullet takes to travel the distance $d_{Z,A}$ at speed v and the time the shockwave takes to travel the distance $d_{A,S}$ at speed of sound c. Hence, the time of arrival of shockwave t_s is given by

$$t_s = t_0 + \frac{d_{Z,A}}{v} + \frac{d_{A,S}}{c} = t_0 + \frac{d_{Z,A}}{v} + \frac{h}{c}. \tag{9.34}$$

Assuming that the bullet is traveling in a straight line, by the time shockwave reaches the sensor array from A to S, the bullet travels from A to C as shown in Fig. 9.17. From the geometry shown in Fig. 9.17, the angle $\angle ASC$ is a right angle and the triangles $\triangle ASC$ and $\triangle ASB$ are similar. Hence, the angle $\angle ASB = \theta$. The miss distance, which is denoted by 'r' on Fig. 9.17, can be calculated by measuring the shockwave duration T using (9.19), which is repeated below:

$$T = \frac{1.82\ M\ r^{1/4}}{c\left(M^2-1\right)^{3/8}}\ \frac{d}{l^{1/4}} \approx \frac{1.82d}{c}\left(\frac{Mr}{l}\right)^{1/4} \tag{9.35}$$

by knowing the Mach number M, diameter d, and length l of the projectile. Note that M, d and l are known for most of the bullets used by various guns. The shockwave duration is measured from the waveform captured by the microphones of the sensor array S as shown in Figs. 9.12 and 9.13.

From (9.33) to (9.34), the TDOA between the muzzle blast and shockwave is given by

$$t_{ms} = t_m - t_s = \frac{R}{c} - \frac{d_{Z,A}}{v} - \frac{h}{c}. \tag{9.36}$$

Note that the t_{ms} is measured from the received signals at the microphones of the sensor array S. The distance R is estimated using the known values of the other distances in (9.36) and the estimated miss distance r from (9.35). From the geometry, the distance h is given by

$$h = \frac{r}{\cos\theta} \tag{9.37}$$

and

$$d_{Z,A} = d_{Z,B} - d_{A,B} \tag{9.38}$$

where

$$d_{Z,B} = \sqrt{R^2 - r^2} \tag{9.39}$$

and

$$d_{A,B} = h\tan\theta. \tag{9.40}$$

Equation (9.36) can be solved for the distance R between the sniper location and the sensor array, resulting in a closed-form solution [80]

$$R = \frac{1}{2\left(c^4 - v^4\right)}\left(P - 2\sqrt{Q}\right) \tag{9.41}$$

where P and Q are defined as

$$P = -2v^3\ r\sqrt{v^2+c^2} - 2t_{ms}\ c^3\ v^2 + 2c^2\ r\ v\sqrt{v^2+c^2} - 2t_{ms}\ c\ v^4$$

$$\begin{aligned} Q = &-2c^4\ v^4\ r^2 + 2\ t_{ms}^2\ c^6\ v^4 + t_{ms}^2\ c^4\ v^6 - 2c^7\ r\ t_{ms}\ v\ \sqrt{v^2+c^2} \\ &+ c^8\ t_{ms}^2\ v^2 + 2c^8\ r^2 + 2v^5\ r\sqrt{v^2+c^2}\ t_{ms}\ c^3 \end{aligned}$$

Since we know the DOA of muzzle blast and the range R, we know the location of the sniper. In this section, we showed how to determine the location of a sniper using an array of microphones provided the bullet is supersonic. In the next section, we use distributed microphones to determine the origin of a supersonic bullet.

9.3.5 Sniper Localization of Supersonic Gunfire with Distributed Microphones

When the sensors are distributed, time synchronization among the sensors is critical [23]. If the sniper localization sensors use individual data collection systems, synchronization among the data collection units is not guaranteed. This is evident in Fig. 9.18, which shows the recordings of the same gunshot data on five different personal digital assistants (PDAs) showing the time delays due to the lack of time

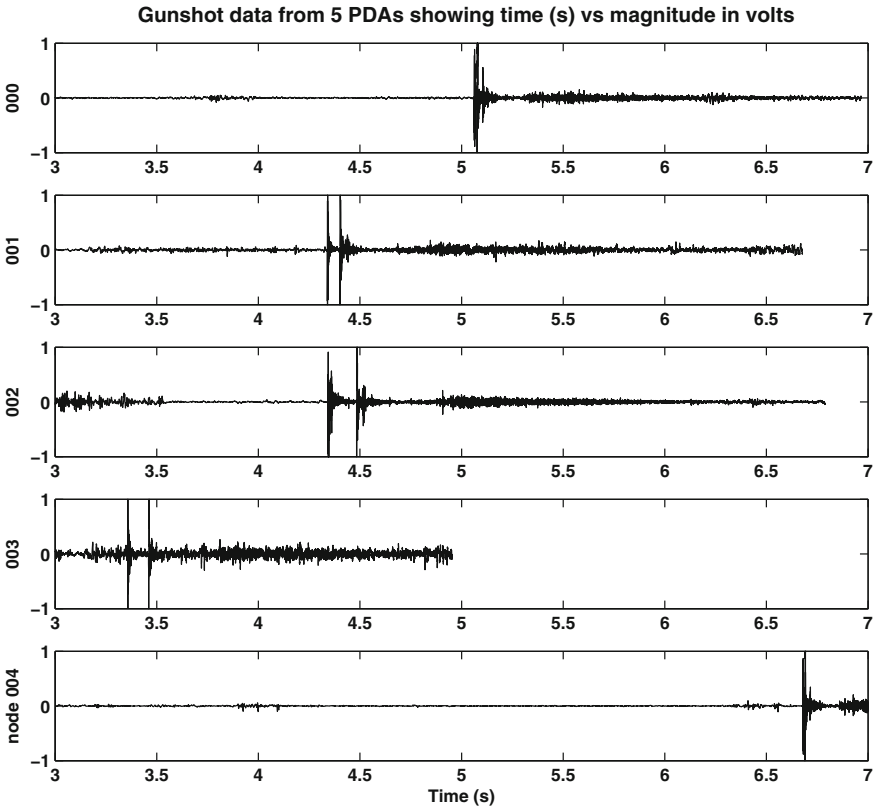


Fig. 9.18 Acoustic data of gunshot collected by five PDAs

synchronization. As a result, the TOA estimates of different events could be off by several seconds making localization impossible. However, the TDOA between the muzzle blast and shockwave can be measured accurately since the internal clocks of the data collection systems are stable during short periods of time. The algorithm presented here uses the TDOAs between the muzzle blast and shockwave. This requires that the events, namely, the muzzle blast and shockwave, be detected at each microphone. Standard algorithms presented earlier for the detection of transient events (shockwave and muzzle blast) in this chapter can be used. From these detections, the TOA of the muzzle blast and shockwave are estimated. The difference in TOAs of the muzzle blast and shockwave gives the TDOA between them. Since the TOA of the shockwave depends on the bullet's velocity, known as Mach number, one has to estimate it using (9.35), which uses the known parameters of the bullet, often provided in charts by the bullet manufacturer. It is well known from ballistic data that the bullet looses its speed due to atmospheric friction as it moves away from the gun. It is assumed here that the velocity of the bullet is constant and does not decrease during flight and the trajectory of the bullet is a straight line—which is a valid assumption for the distances up to 300 m [6].

***Derivation of Sniper Equations*:** Fig. 9.19 shows the geometry of the bullet trajectory and the shockwave cone. In Fig. 9.19, Z denotes the location of the sniper and U is the unit vector in the direction of the bullet. As the bullet travels at supersonic speed, the shockwave generates a cone with angle θ, where $\sin\theta = 1/M$, M is the Mach number. The shockwave propagates perpendicular to the cone surface and reaches sensor S_k. The point where the shockwave radiates towards the sensor is denoted

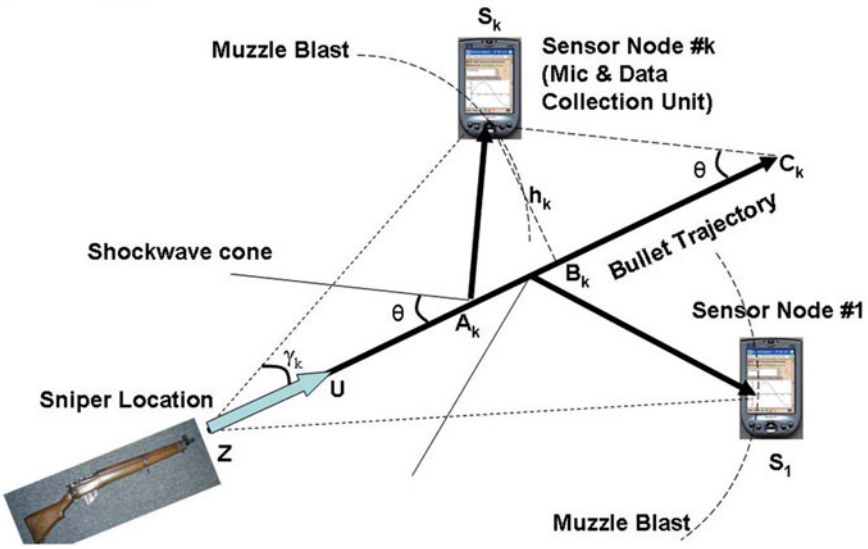


Fig. 9.19 Geometry of the bullet's trajectory and the shockwave cone

by A_k. By the time the shockwave reaches sensor S_k, the bullet has traveled from A_k to C_k and the miss distance is given by $h_k = \|S_k - B_k\|$, where $\|B\|$ denotes the norm of the vector B. Let γ_k be the angle between the trajectory of the bullet and the line joining sniper location Z and sensor location S_k. Let us denote the TOA of the muzzle blast and shockwave as T_k and t_k respectively,

$$T_k = \frac{\|S_k - Z\|}{c}; \quad t_k = \frac{\|A_k - Z\|}{Mc} + \frac{\|S_k - A_k\|}{c} \tag{9.42}$$

where c denotes the propagation velocity of the sound. Note that in the time the shockwave propagates from A_k to S_k, the bullet travels from A_k to C_k; thus, the bullet travels from Z to C_k during the time period t_k. Then t_k can be re-written as

$$\begin{aligned} t_k &= \frac{\|A_k - Z\|}{Mc} + \frac{\|S_k - A_k\|}{c} = \frac{\|C_k - Z\|}{Mc} \\ &= \frac{\|B_k - Z\|}{Mc} + \frac{\|C_k - B_k\|}{Mc} \\ &= \frac{1}{Mc}\left((S_k - Z)^T\, U + h_k \cot\theta\right) \end{aligned} \tag{9.43}$$

where 'T' is the transpose and $\|B_k - Z\|$ is the projection of vector $S_k - Z$ onto the trajectory of the bullet denoted by the unit vector U. Use of the trigonometric relations that $\|B_k - Z\| = \|S_k - Z\| \cos\gamma_k$ and $h_k = \|S_k - Z\| \sin\gamma_k$ and $\sin\theta = 1/M$, results in

$$\begin{aligned} t_k &= \frac{\|S_k - Z\|}{c}(\sin\theta\cos\gamma_k + \cos\theta\sin\gamma_k) \\ &= \frac{\|S_k - Z\|}{c}\sin(\theta + \gamma_k) \end{aligned} \tag{9.44}$$

and

$$\frac{(S_k - Z)^T\, U}{\|S_k - Z\|} = \cos\gamma_k \tag{9.45}$$

Then from (9.42) to (9.44), the TDOA is given by

$$\tau_k = T_k - t_k = \frac{\|S_k - Z\|}{c}\left[1 - \sin(\theta + \gamma_k)\right] \tag{9.46}$$

Re-arranging (9.46) results in linear equations

$$\|S_k - Z\| = \frac{d_k}{q_k}, \quad \forall k = \{1, 2, \ldots, n\} \tag{9.47}$$

where $d_k = c\,(T_k - t_k)$, $q_k = [1 - \sin(\theta + \gamma_k)]$ and 'n' is the number of sensors. Eqs. (9.47) and (9.45) are used to solve for the sniper location and the trajectory of the bullet, respectively. The TDOA 'τ_k' is estimated from sensor data for each sensor 'k' and the parameters c and θ are assumed to be known.

9.3.5.1 Implementation of Sniper Localization Algorithm

There are two parts to this algorithm development. The first part consists of data collection in order to understand the nuances of the data. In particular, sniper data collected at different times of day and on different days vary significantly due to atmospheric effects. One of the variations observed in the data are the bullet's velocity at different times of day. Another variation observed is in the propagation velocity of the sound. The second part consists of developing the algorithm. A good algorithm should be robust enough to handle the data irrespective of when they are collected.

Any set of distributed microphones could be used to determine a sniper's position. These microphones could be ones in the cell phones of different people in an area or the ones mounted on the helmets of soldiers, etc. In Fig. 9.20 we show eight microphones distributed in an area. They are mounted on tripods, as live bullets are used in the data collection. Mounting of single microphone sensor used for data collection is also shown. The microphones used for data collection should have a frequency response >100 kHz so that the fast rising supersonic 'N' wave front can be

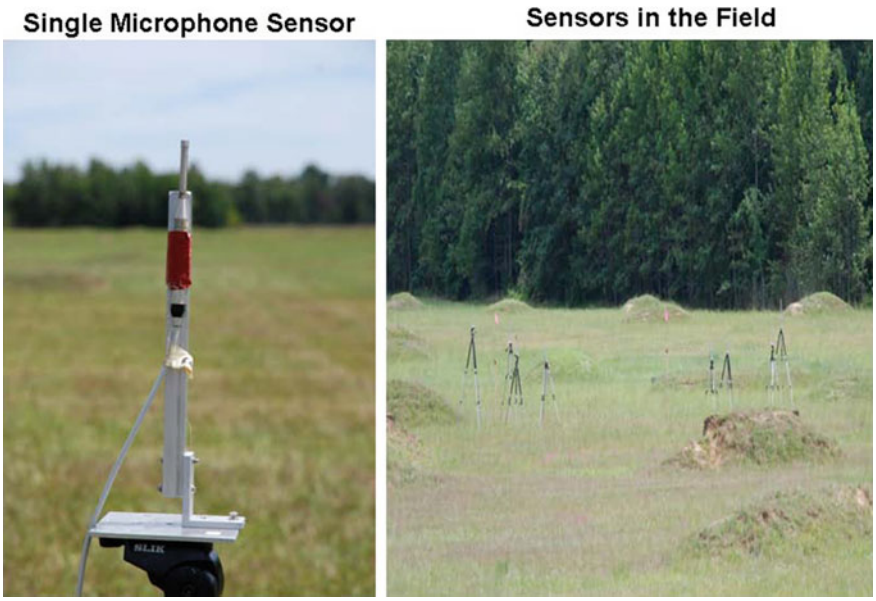


Fig. 9.20 Single microphone sensor and their placement in the field

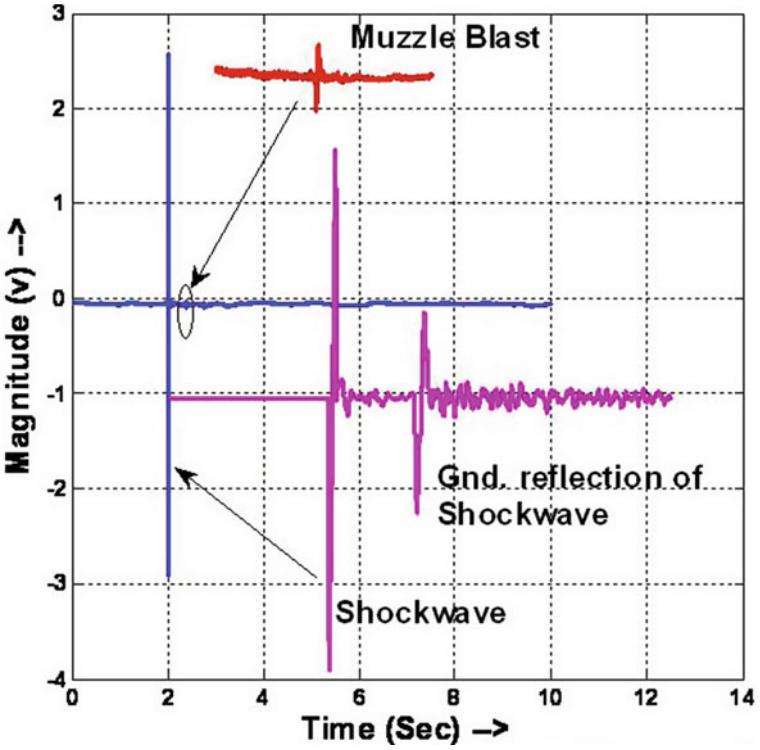


Fig. 9.21 Acoustic signature of supersonic bullet

captured properly. In order to help firing the gun, three aim points were selected with one aim point at the center of the sensor field and the other two aim points to either side from the center. These aim points help the shooter to aim while firing so that the data collected are consistent. The sensors were located about 250 m downrange. The data were collected at 100 k samples per second in order to capture the fast rising shockwave.

Figure 9.21 shows the shockwave and muzzle blast waveforms captured by one of the sensors. The shockwave is a characteristic 'N' wave, which is expanded and shown as an insert. Notice that the sensor also captured the ground reflection of the shockwave where the 'N' wave is clearly visible. The signal from the muzzle blast gets attenuated as it propagates through atmosphere and, as a result, it is much smaller in amplitude compared to the shockwave. The insert on the top of Fig. 9.21 shows the muzzle blast waveform.

In order to solve for the sniper position using the TDOAs, one needs to determine the TDOA at each sensor very accurately from the acoustic data captured by each sensor. Often this is the source of the error, as determining the points of reference on the 'N' wave and muzzle blast (often the raising edges of shockwave and the muzzle blast) are subject to noise. This measurement error in TDOAs results in error in

estimating the sniper position. Now the technique used for measurement of TDOA is discussed, although any other techniques that can measure the time difference between the muzzle blast and the shockwave can be used.

***Estimation of TDOA*:** In order to determine the TDOA, first the peaks associated with the shockwave and muzzle blast signals are detected. Then, for each event, a portion of the signal prior to its peak and onset of the 'N' wave is selected for collection of mean and variance. The TDOA is determined as the time difference between the onset of the shockwave and the muzzle blast. Detection of the onset of the N-wave and muzzle blast are determined if the amplitude of these events are above certain thresholds. The threshold for each event is set as the noise mean plus twice the sigma value of noise prior to the event.
Two other parameters are required for solving the sniper localization problem, namely, the propagation velocity of sound 'c' and the Mach number 'M' for the bullet. It is well known [89] that the velocity of the sound is dependent on the atmospheric conditions such as air temperature, humidity, etc. The propagation velocity 'c' is estimated using the meteorological data, i.e., the temperature, using the formula [89]

$$c = 331.3\sqrt{1 + \frac{\tau}{273.15}} \text{ m/s} \tag{9.48}$$

where the temperature τ is in Celsius.
Determining the bullet's speed could be done by measuring the raise and fall time of the N-wave as shown in [36]. Most bullet manufacturers provide ballistic data on the bullet. However, the bullet speed varies depending on the gun used, type and length of barrel of the gun, etc. Other major contributing factors that affect the velocity of the bullet are (a) the charge used for the bullet and (b) atmospheric effects such as temperature. Figure 9.22 shows the time of day each shot was fired, the propagation velocity of sound and the measured TDOA for each shot. The shots are fired on two different days; some shots are fired during the morning while others are fired during the afternoon. The temperature difference from morning to afternoon and its effect is reflected in the propagation velocity of the sound. The measured TDOAs also show three different mean values. The TDOA changes depending on the propagation velocity of the sound and the bullet's speed, which also varies with atmospheric conditions. For the sake of this book, the bullet's speed is estimated using the ground truth of the sensors, measured TDOAs, and the sniper location.

9.3.5.2 Estimation of Sniper Position

We now present the algorithm for determining a sniper's position using a constant velocity model for bullet speed. The algorithm is primarily concerned in solving for the sniper position Z iteratively, using (9.45) and (9.47). Linearization of (9.47) gives

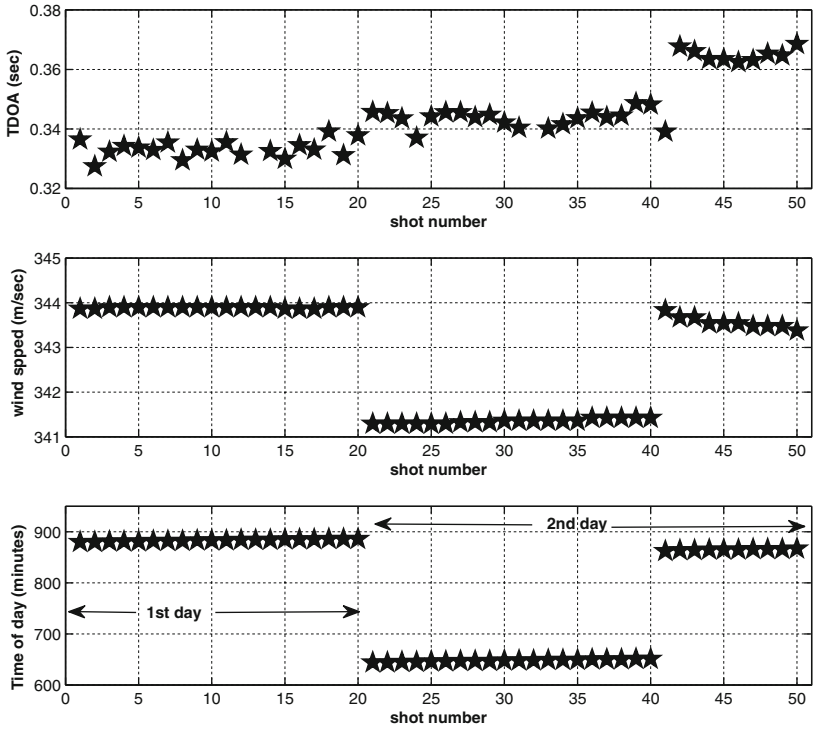


Fig. 9.22 TDOA, propagation velocity, time of the day

$$(S_k - Z)^T \ (S_k - Z) = \frac{d_k^2}{q_k^2}$$

or

$$\|S_k\|^2 - 2S_k Z + \|Z\|^2 = \frac{d_k^2}{q_k^2} \tag{9.49}$$

Subtracting (9.49) for ith sensor from jth sensor for all i and $j \neq i$, gives

$$2\begin{bmatrix} (S_1 - S_2)^T \\ (S_1 - S_3)^T \\ \vdots \\ (S_{n-1} - S_n)^T \end{bmatrix} \tilde{Z} = \begin{bmatrix} (d_2/q_2)^2 - (d_1/q_1)^2 - S_2^2 + S_1^2 \\ (d_3/q_3)^2 - (d_1/q_1)^2 - S_3^2 + S_1^2 \\ \vdots \\ (d_n/q_n)^2 - (d_{n-1}/q_{n-1})^2 - S_n^2 + S_{n-1}^2 \end{bmatrix} \tag{9.50}$$

where 'n' is the number of sensors. To solve (9.50) for $\tilde{Z}$, one would require the values of d_k, and q_k for all k. While $d_k = c\,(T_k - t_k)$ can be computed using the TDOA measurements and the propagation velocity of the sound, $q_k = 1 - \sin(\theta + \gamma_k)$

requires the miss angle γ_k for each sensor k. These miss angles are not known a priori—in fact, these need to be estimated. The procedure used to estimate the initial values of γ_k is presented later. Assuming initial set of values for these angles are known; they are used in (9.50) to solve for $\tilde{Z}$ which is then used to estimate the values of γ_k denoted by $\hat{\gamma}_k$ using (9.47), that is,

$$\hat{\gamma}_k = \sin^{-1}\left(1 - \frac{d_k}{\left\|S_k - \tilde{Z}\right\|}\right) - \theta \tag{9.51}$$

Using the values of $\hat{\gamma}_k$'s and $\tilde{Z}$, we can estimate the unit vector in the direction of the trajectory by linearizing the (9.45) given by

$$\begin{bmatrix} \left(S_1 - \tilde{Z}\right)^T \\ \left(S_2 - \tilde{Z}\right)^T \\ \vdots \\ \left(S_n - \tilde{Z}\right)^T \end{bmatrix} \tilde{U} = \begin{bmatrix} \left\|S_1 - \tilde{Z}\right\| \cos\hat{\gamma}_1 \\ \left\|S_2 - \tilde{Z}\right\| \cos\hat{\gamma}_2 \\ \vdots \\ \left\|S_n - \tilde{Z}\right\| \cos\hat{\gamma}_n \end{bmatrix} \tag{9.52}$$

which can be solved for $\tilde{U}$ using regression. These estimated values of Z, and U are used to re-estimate the values of γ_k using the (9.45), that is,

$$\breve{\gamma}_k = \cos^{-1}\left(\frac{\left(S_k - \tilde{Z}\right)^T \tilde{U}}{\left\|S_k - \tilde{Z}\right\|}\right) \tag{9.53}$$

Ideally if the values of $\tilde{Z} = Z$ and $\tilde{U} = U$, the values of $\hat{\gamma}_k$ and $\breve{\gamma}_k$ calculated using the (9.51) and (9.53), respectively, should be equal, resulting in a determination of the sniper location and bullet trajectory. Figure 9.23 shows the flowchart for the algorithm. As shown in the flow chart, initially one needs to estimate reasonable values for the γ_k for all k. Next few paragraphs discuss how the initial values for the gammas are determined.

Prior to estimating the values of the gammas, it would be useful to understand the nature of gammas in terms of how they change with distance while keeping the TDOA constant. Figure 9.24 shows the gamma versus distance for different TDOAs measured at different sensors. The curves in Fig. 9.24 are obtained from (9.51) keeping θ and $d_k = (T_k - t_k) * c$ constant and changing the values of $D_k = \|S_k - Z\|$. From Fig. 9.24, it can be seen that: (a) as the distance between the sniper location and sensor D_k increases to ∞, the gamma asymptotically reaches the value $\pi/2 - \theta$, (b) for any given D_k, the value of gamma increases with TDOA, hence, d_k and (c) for practical cases, the magnitudes of the gammas are greater than zero.

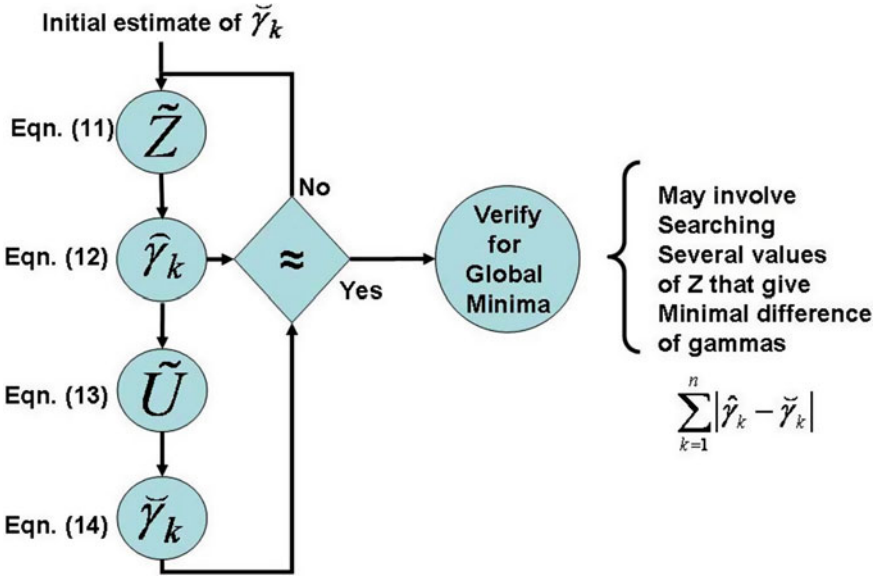


Fig. 9.23 Flowchart for the sniper localization algorithm

***Initial estimate of gammas*:** Generate the gamma versus distance for each measured TDOA as in Fig. 9.24. Since the gammas need to be positive, find the minimum distance D_{min} where all gammas are positive. Initial estimates of the gamma values correspond to the values of the gammas in Fig. 9.24 when $D_k \geq D_{min}$ such that $\gamma_k \geq 0$ for all k.

Once the initial values of the gammas for all sensors are estimated as above, they can be used in the algorithm shown in Fig. 9.23 to estimate the location of the sniper. Unfortunately, the solution given by the algorithm in Fig. 9.23 corresponds to one of the local minima. It is also clear from Fig. 9.24 that there are many local minima, as for any given set of TDOAs there are several D_k's. However, the solution obtained by the above algorithm approximately provides the correct direction of the sniper location provided the error $\sum_{k=1}^{n} \left\| \breve{\gamma}_k - \hat{\gamma}_k \right\|$ is small. In order to find the global solution, it is necessary to search the area in the direction provided by the above solution. From Fig. 9.24 it is clear that the search can extend until $D_k \approx \infty$. The lower limit on the distance is already established by D_{min}. In the next few paragraphs, an upper bound on the maximum distance D_{max} for the search is presented based on the sensor geometry. In the event the error is not small, a search has to be performed between two circles with radii D_{min} and D_{max}.

For all practical situations, the sensors are closely located (probably within few meters from one another), which means that the difference between the gamma values is also small. From Fig. 9.24, it is easy to see that the distance D_k increases with an increase in gamma value. The difference $D_i - D_j$ is estimated as a function of the sensor positions. Based on this relation, an upper bound on D_k, that is, D_{max}, cab be established. Figure 9.25 shows two different configurations of the sensor placements.

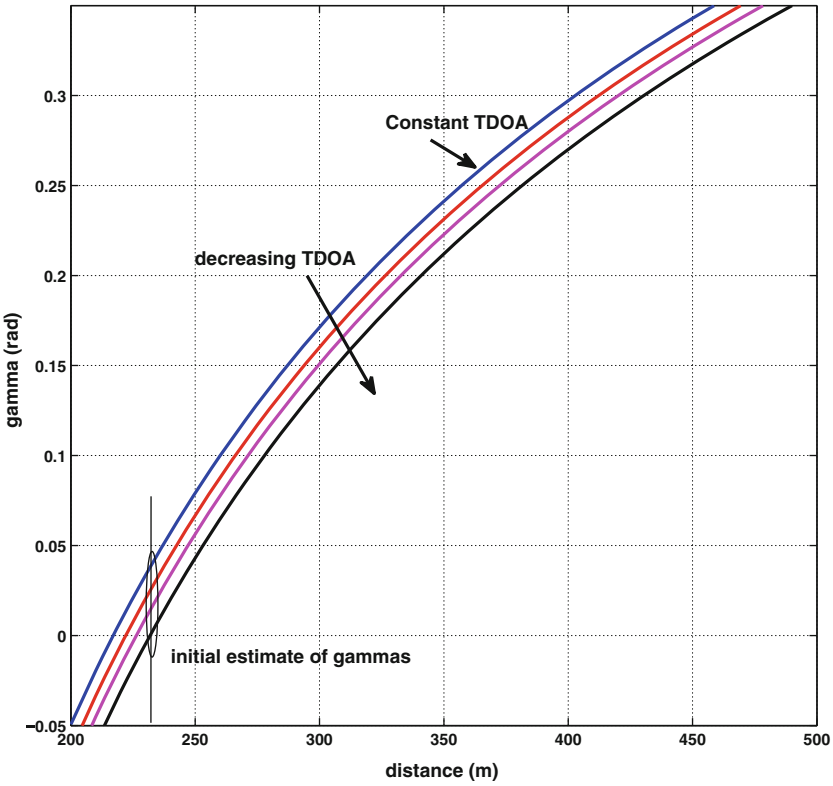


Fig. 9.24 Gamma versus distance for fixed TDOA

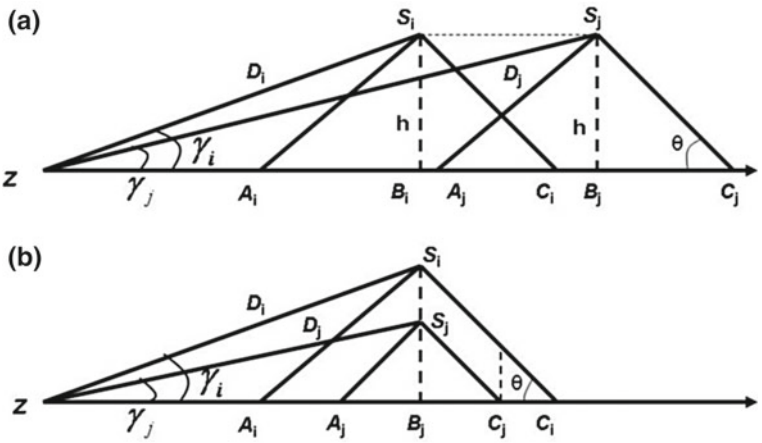


Fig. 9.25 Sensor placements **a** *horizontal*, **b** *vertical*

***Horizontal placement of sensors*:** From Fig. 9.25, it can be seen that the time difference between the TOA of the shockwave at sensor S_i and S_j is given by

$$t_i - t_j = \frac{\|C_i - C_j\|}{Mc} = \frac{\|S_i - S_j\|}{Mc}$$

Similarly, the difference in the TOA of muzzle blasts at the sensors is given by

$$T_i - T_j = \frac{\|D_i - D_j\|}{c}$$

Then the difference in the distances $D_i - D_j$ can be expressed using the TDOAs $\tau_i = T_i - t_i$ and $\tau_j = T_j - t_j$ as

$$D_i - D_j = (\tau_i - \tau_j)\, c + \frac{\|S_i - S_j\|}{M} \tag{9.54}$$

***Vertical placement of sensors*:** From Fig. 9.25b, the difference in TOA of the shockwaves at sensors S_i and S_j is estimated as the time the bullet takes to travel the distance $C_i - C_j$ and is given by

$$t_j - t_i = \frac{\|C_j - C_i\|}{Mc} = \frac{\|S_j - S_i\| \cot\theta}{Mc}.$$

Then the difference in TDOAs is given by

$$\tau_j - \tau_i = \frac{D_j - D_i}{c} - \frac{\|S_j - S_i\| \cot\theta}{Mc}$$

Then the difference in distances $D_i - D_j$ is given by

$$D_j - D_i = (\tau_j - \tau_i)\, c + \frac{\|S_j - S_i\| \cot\theta}{M} \tag{9.55}$$

***Determination of* $\mathbf{D_{max}}$:** First generate the gamma versus distance plots as shown in Fig. 9.24 for each measured TDOA for all sensors S_k, $k \in 1, 2, \ldots, n$. Find the largest estimate of $D_i - D_j$ using (9.54) and (9.55). Now from Fig. 9.24, find the gamma value that spans all the curves with a horizontal spread equal to $D_i - D_j$. Then the distance D_{max} is given by the x-coordinate corresponding to the value of the gamma for the lowest TDOA.

Now the final sniper localization algorithm is as follows:

Sniper Localization Algorithm 1:

Step 1. Determine the initial values of the gammas
Step 2. Find the initial estimate of the sniper location using the algorithm in Fig. 9.23
Step 3. Determine the values of D_{min} and D_{max} for the global search for the sniper location
Step 4. Search the area bounded by D_{min} and D_{max} in the direction of the initial estimate of the sniper location for a global solution that gives the minimum error of $\sum_{k=1}^{n} \left\| \hat{\gamma}_k - \breve{\gamma}_k \right\|$.

Chapter 10
Classifiers

This chapter deals with machine learning. Machine learning assist with classifying or sorting data. Humans routinely relegate visual and auditory signals into various categories. Humans look at various objects, categorizing them as a chair, a table, a person, an animal, etc. This ability is a learned skill developed and practiced from childhood. As with humans, machines too can be trained to recognize and sort items into convenient and useful categories. For instance, these days weather prediction is done routinely using a number of machine-based tools and sensors, and then is broadcast on television. In order to do weather prediction, several measurements such as atmospheric pressure, pressure changes, cloud formation and the rate at which the clouds are traveling, etc., are made at several geographical places. Based on these measurements, the data are placed into a category: for example, (a) those data that indicate rain tomorrow and (b) those data that do not. Let us assume that the data pertaining to the category "rain tomorrow" lie in a region R_1 and the those data in the "do not rain tomorrow" lie in the region R_2, as shown in Fig. 10.1, then the problem of classifying the data becomes finding out whether the newly measured data belongs in region R_1 or R_2.

Thus, weather prediction can be cast as a pattern classifier. The pattern classifier receives a vector of numbers $X = \{x_1, x_2, \ldots, x_m\}$ as an input pattern, where x_i corresponds to ith measurement, say, atmospheric pressure, and it generates an output $y \in \{1, 2, \ldots, R\}$ indicating that the input pattern belongs to one of the R categories. In the case of weather prediction mentioned earlier $R = 2$. A pattern classifier is depicted in Fig. 10.2. There are several issues associated with pattern classification:

1. Measurements
2. Feature extraction
3. Decision surfaces and regions using discriminant functions

We discuss each aspect in turn.

T. Damarla, *Battlefield Acoustics*, DOI 10.1007/978-3-319-16036-8_10

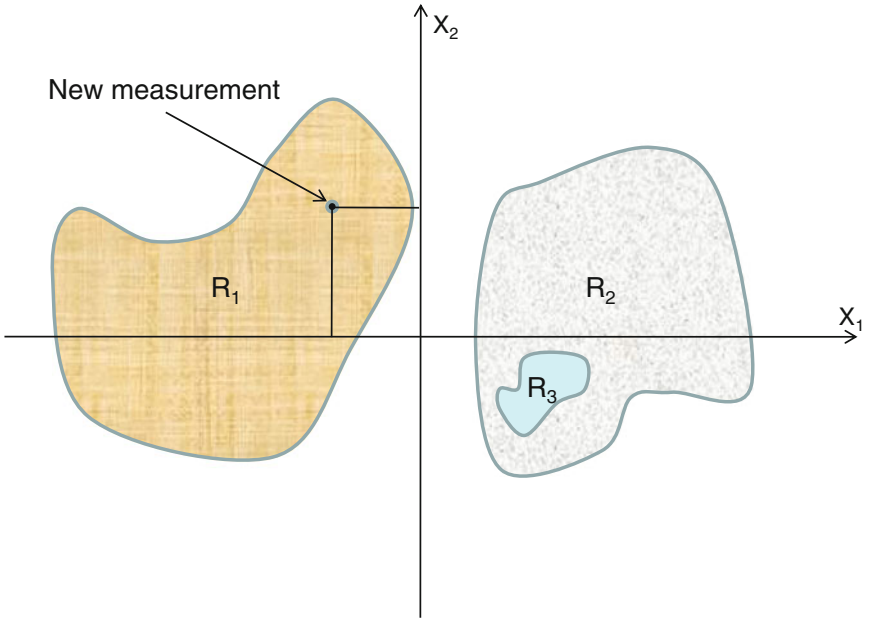


Fig. 10.1 Different regions

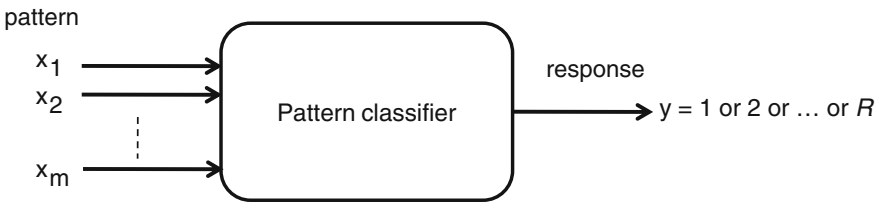


Fig. 10.2 Pattern classifier

10.1 Measurements

In order to classify data, one must first measure the phenomena generated by the targets we are trying to classify. For example, to classify motor vehicles (car, tank, helicopter, airplane) or musical instruments (flute, clarinet, guitar), one must first collect the sound created by these targets using data acquisition systems. For example, one could use a computer to record the sound generated by the targets. Several commercial companies sell data collection equipment with varying capabilities.

It is important to keep in mind the type of data that is being collected, particularly, in terms of minimum and maximum frequency. For example, if we are trying to collect the output of music instruments, the frequency response is normally high if they are string instruments, whereas the drums have a lower frequency response. Similarly,

if the data collected are from a supersonic bullet, which produces a shockwave, the data should be collected at a high sampling rate (say 100 k samples/sec) in order to capture the fast rising N-wave. In general, the sampling rate should be greater or equal to the Nyquist rate, that is, twice the highest frequency produced by the target.

10.2 Feature Extraction

Feature extraction is an essential part of the classification process. The raw data measurements are often large in size and majority of them are redundant. Hence, a small dimensional feature vector may be sufficient to classify a target. For example, few spectral coefficients may be sufficient to distinguish between string instruments and drums. It is not necessary to apply the complete data to the pattern classifier for classification. The features selected as input patterns $X = \{x_1, x_2, \ldots, x_m\}$ to the classifier need to be selected carefully and should be based on the physics governing the target output. The computational complexity of the pattern classifier depends on the number of features m. Use of good features makes the classification robust and results in a classifier that produces fewer misclassifications. On the other hand, use of poorly selected features with little discriminant capability results in lot of false alarms and eventually one would loose confidence in the classifier. We now discuss some of the features used in acoustic signal processing.

- Spectral features
- Cepstral coefficients
- Mel-frequency cepstral coefficients
- Temporal features

Spectral features: Spectral features are some of the most widely used features in acoustic signal processing. Figure 10.3 shows a spectrogram of typical helicopter data. From the figure we notice that there are several harmonics of the fundamental frequency due to main rotor of the helicopter. The magnitudes of these harmonics can be used as a feature vector for determining whether the pattern belongs to a helicopter or a car, etc. Only a few spectral coefficients are sufficient for classification. In a majority of cases, the phase information is discarded resulting in data reduction. Higher-order spectral coefficients are also discarded as they do not contribute to the classification much. This further reduces the number of features to a reasonable size.

Cepstral coefficients: The origin of the word "cepstral" is obtained by reversing "spec" in spectral and joining it with "tral" in spectral. Let us first understand the motivation for the cepstral analysis of the signals. In general, the output of a system can be represented as a convolution of the system response with the excitation signal as shown in Fig. 10.4.

$$s(t) = e(t) * h(t) \tag{10.1}$$

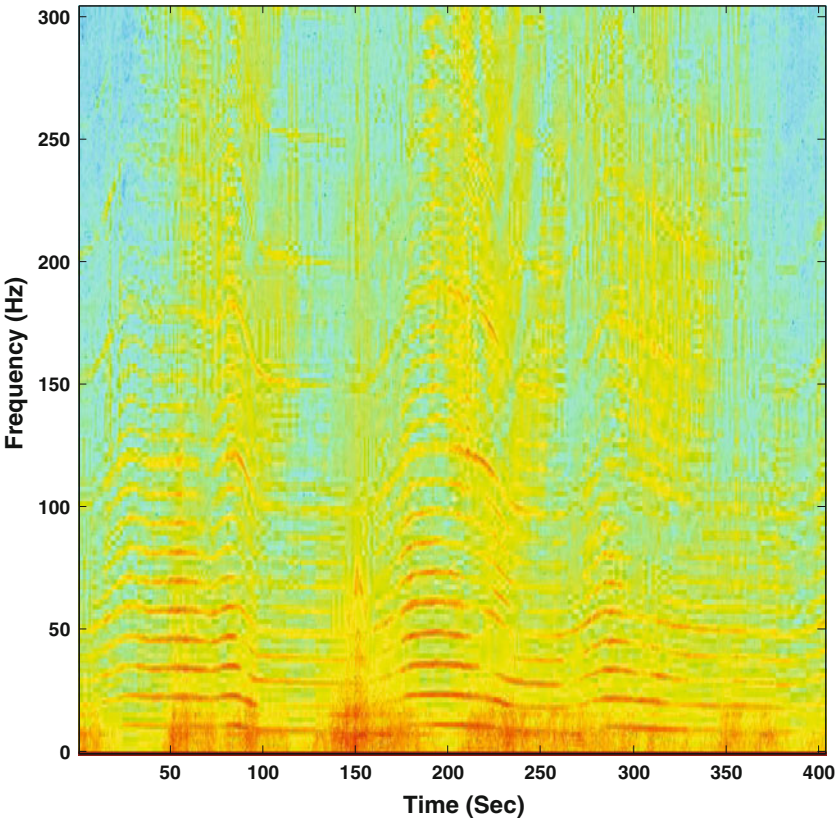


Fig. 10.3 Spectrogram of typical helicopter data

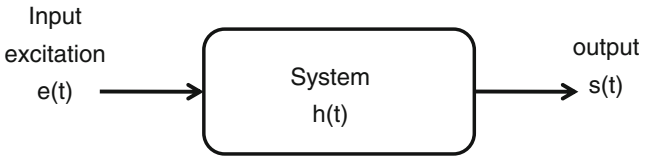


Fig. 10.4 System response

In frequency domain, the above equation can be represented as

$$\begin{aligned} S(w) &= E(w).H(w) \\ \text{Or} \qquad |S(w)| &= |E(w)|.|H(w)| \end{aligned} \tag{10.2}$$

where the convolution in the time domain becomes a product in the frequency domain. If the excitation signal is known, the system response can be obtained by deconvolution. If the system response is known, the excitation signal can be obtained by a linear prediction process. If the input excitation signal and system response are

unknown, then another type of deconvolution is used. In general, a speech signal is generated by some excitation and vocal cord response. These two components may have to be separated in order to model the excitation signals. This is where cepstral analysis helps. In cepstral analysis, the multiplied source and system components in the frequency domain are transformed as a linear combination of the two in the cepstral domain. This is done by a logarithmic operation

$$\log|S(w)| = \log|E(w)| + \log|H(w)| \tag{10.3}$$

The separation of the excitation and system responses can be done by taking the inverse discrete Fourier transformation (IDFT)

$$c(n) = \text{IDFT}\left(\log|S(w)|\right) = \text{IDFT}\left(\log|E(w)| + \log|H(w)|\right). \tag{10.4}$$

This inverse transform results in time domain-like data, and the new domain is called the quefrency domain or cepstral domain. If the responses of $E(w)$ and $H(w)$ are distinct, they can be windowed, and individual excitation and system responses can be extracted. In general, for classification, only few cepstral coefficients are used. A typical number is $m = f_s/2000$, where f_s is the sampling rate.

Mel frequency cepstral coefficients (MFCC): These are used to improve the scaling effect of the frequency. The process used in generating MFCC is shown in Fig. 10.5. The Mel frequency scale is given by

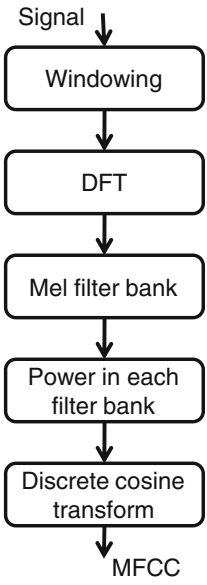


Fig. 10.5 Generation of Mel frequency cepstral coefficients

$$\text{Mel}(f) = 2595 \log_{10}(1 + f/700) \tag{10.5}$$

The success of the MFCC is mainly due to a log of the magnitudes resulting in a better perceptual change.

Temporal Features: Some of the temporal features used are the following:

- Rise time
- Amplitude modulation
- Frequency modulation

Training:

In general, a set of patterns (features) with their known classification is called a training set, in which all the data are applied to a training algorithm corresponding to one of the classifiers and the weights/parameters of the classifier are adjusted to result in the proper classification. The following table shows a sample training set for a two-class problem.

Test pattern	x_1 ... x_m	Class
X_1	0.12 ··· 2.35	1
X_2	1.32 ··· 5.45	1
X_3	6.32 ··· 0.75	2
⋮	⋮ ⋮ ⋮	⋮
X_{N-1}	0.53 ··· 3.79	2
X_N	1.02 ··· 3.94	1

In the next section, we present the general theory of pattern classification, which shows how to separate various regions by discriminant functions.

10.3 Decision Surfaces and Regions Using Discriminant Functions

Let us consider the regions R_1 and R_2 in Fig. 10.1. The point sets in the two regions are separated from each other by surfaces. In general, the entire Ecludian space E^m is divided into R regions, which we call decision regions [71]. The kth region R_k comprises all the points (patterns), which map into the kth category. In practice, the regions are generated based on the training sets. The training sets are selected from the measurements, which belong to a known category. A reasonable set of points is used to define each region.

Let $g_i(X), \ \forall i \in \{1, 2, \ldots, R\}$ be a set of single-valued functions such that $g_i(X) > g_j(X)$ for $i, j = 1, 2, \ldots, R; \ j \neq i$ for all $X \in R_i$. These functions are called discriminant functions and are continuous on the decision surfaces. The decision surface separating the contiguous regions R_i and R_j is given by

$$g_i(X) - g_j(X) = 0 \tag{10.6}$$

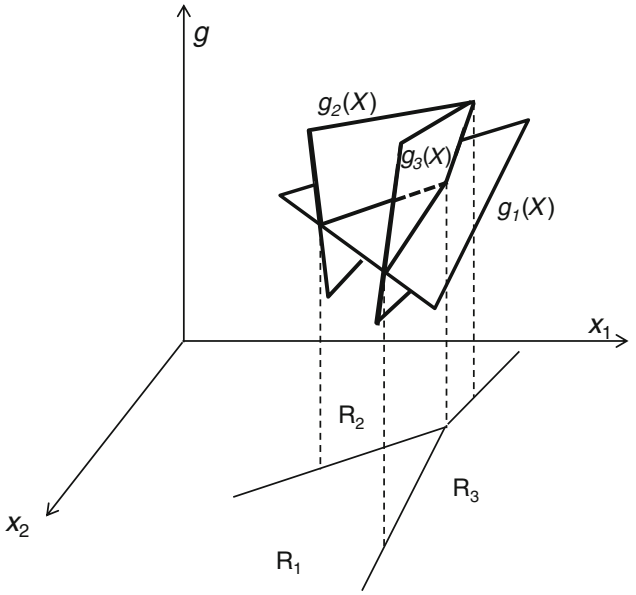


Fig. 10.6 Examples of discriminant functions

Figure 10.6 shows the discriminant functions and the regions they define in 2-D plane. Once the discriminant functions are selected, then classification is performed by computing $g_i(X)$, $\forall i = 1, 2, \ldots, R$ and determining $g_i(X)$ such that $g_i(X) > g_j(X)$ for all $j \neq i$, as shown in Fig. 10.7. The outputs of the discriminant functions are called discriminants. Often selection of the discriminant function is done via training. There are two types of training methods: (a) parametric and (b) non-parametric. Parametric methods are used if a set of parameters can be identified a priori for all categories. The values of these parameters can be estimated using the training data.

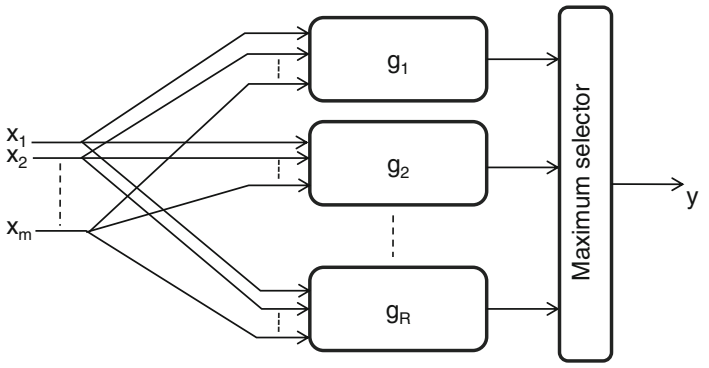


Fig. 10.7 Classification of patterns

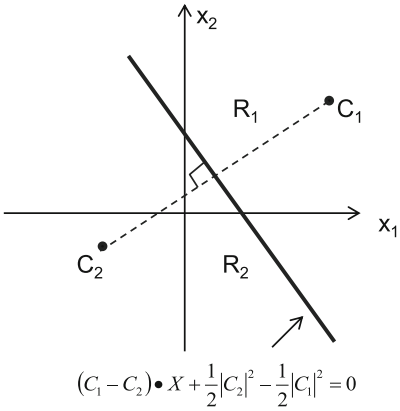


Fig. 10.8 A linear decision surface

Non-parametric methods are used if there are no assumptions that can be made about characterizing the parameters.

As an example of the parametric method, consider the case where there are two categories whose points are distributed in two regions R_1 and R_2. It is also known that the points in each category are clustered around some point C_1 for category # 1 and C_2 for category # 2. The values of C_1 and C_2 can be estimated using the training data. Once C_1 and C_2 are estimated, one can use a hyperplane to divide the two regions. Such a hyperplane would be the plane that bisects and is normal to the line joining C_1 and C_2. Such a hyperplane equation (discriminating function) is given by

$$g(X) = (C_1 - C_2) \bullet X + \frac{1}{2}|C_2|^2 - \frac{1}{2}|C_1|^2 = 0 \tag{10.7}$$

where '•' denotes the dot product. Figure 10.8 shows the hyperplane decision surface for a two-dimensional case. The function in (10.7) is also called the linear discriminant function. Linear discriminant functions are used for those regions that are separable in the domain of operation. For example, there is no linear discriminant function that can separate regions R_2 and R_3 in Fig. 10.1 as the latter is embedded in R_2. For the case in Fig. 10.8, the function $g(X) > 0$ if an input pattern X is in the region R_1, while $g(X) < 0$ if X lies in the region R_2. So a simple threshold detector applied to the output of the discriminator would be sufficient.

$$g(X) = \begin{cases} > 0 & \text{if } X \in R_1 \\ < 0 & \text{if } X \in R_2 \end{cases} \tag{10.8}$$

For non-parametric case, there are no assumptions that can be made about the parameters. One might use

$$g(X) = w_1 x_1 + w_2 x_2 + \cdots + w_m x_m + w_{m+1} \tag{10.9}$$

for a two-category problem. The training involves finding the proper weights.

One of the most important classifiers is a multivariate Gaussian classifier in parametric training methods. We now present the multivariate Gaussian classifier.

10.4 Multivariate Gaussian Classifier

The underlying assumption for this classifier is that there are R different categories. Each category is characterized by a set of parameters, for example, mean and variance. These categories are characterized by sets of variables governed by R distinct probability functions $p(X|i),\ i = 1, 2, \ldots, R$. The $p(X|i)$ is the probability of the occurrence of pattern X given that it belongs to category i. If X can take continuous values, the $p(X|i)$ is the probability density function. Clearly, each category is characterized by the parameter set: $p(i)$ and the parameters of $p(X|i)$, that is, mean and variance. Training involves estimating $p(i)$ and the parameters of $p(X|i)$ using the training data.

Suppose, $\lambda(i|j)$ denotes the cost (penalty) incurred by assigning a given pattern X from category 'j' to category 'i'. The construction of the discriminant functions is to minimize the average cost of misclassification. Let $L_X(i)$ be the conditional average loss,

$$L_X(i) = \sum_{i=1}^{R} \lambda(i|j)\ p(j|X), \tag{10.10}$$

where $p(j|X)$ is the probability given X, category is j. For a given X, one can always calculate $L_X(i)$ for all $i = 1, 2, \ldots, R$. Suppose, for a given X, the loss function is the minimum for some $i = i_m$, that is, $L_X(i_m) \leq L_X(i)$ for $i = 1, 2, \ldots, R$. The average loss is minimized if the classifier always assigns this X to i_m. Hence, the algorithm for the classifier using the probabilities is given by:

Algorithm:

1. Given pattern X, compute $L_X(i)$ for all $i = 1, 2, \ldots, R$.
2. The classifier decides that X belongs to category i_m for which $L_X(i_m) \leq L_X(i)$ for $i = 1, 2, \ldots, R$.

We now show how the loss function is computed. Using the Bayes' rule

$$p(j|X) = \frac{p(X|j)p(j)}{p(X)} \tag{10.11}$$

where $p(j)$ is the a priori probability and $p(X)$ is the probability that X occurs irrespective of its category. Substituting (10.11) in (10.10), we get

$$L_X(i) = \frac{1}{p(X)} \sum_{i=1}^{R} \lambda(i|j)\ p(X|j)\ p(j) \tag{10.12}$$

Since $\frac{1}{p(X)}$ is common, a modified loss function $l_X(i)$ can be used

$$l_X(i) = \sum_{i=1}^{R} \lambda(i|j)\ p(X|j)\ p(j). \tag{10.13}$$

One of the most widely used loss functions $\lambda(i|j)$ is the symmetric loss function and it is given by

$$\lambda(i|j) = 1 - \delta_{ij} \tag{10.14}$$

where δ_{ij} is the Kronecker delta function, that is, $\delta_{ij} = 1$ if $i = j$, else $\delta_{ij} = 0$ if $i \neq j$. Substituting (10.14) in (10.13), we get the conditional average loss

$$l_X(i) = p(X) - p(X|i)\ p(i). \tag{10.15}$$

From (10.15) it is clear that the minimization of average loss implies the maximization of $p(X|i)p(i)$. For a special case where $p(i) = 1/R$ for all $i = 1, 2, \ldots, R$, the maximization of $p(X|i)p(i)$ implies maximizing $p(X|i)$. Such a decision is called the maximum likelihood decision. Hence the discriminant functions can be expressed as

$$g_i(X) = p\,(X|i)\ p(i), \quad \text{for } i = 1, 2, \ldots, R. \tag{10.16}$$

These discriminant functions are often expressed using log likelihoods as

$$g_i(X) = \log\ p\,(X|i) + \log\ p(i), \quad \text{for } i = 1, 2, \ldots, R. \tag{10.17}$$

So we have expressed the conditional average loss in terms various probability functions. A special class of probability that is often used in literature and easy to deal with is the normal density function. Since $X = [x_1, x_2, \ldots, x_m]^T$, then the m-variate normal probability density function $p(X|i)$ is given by

$$p(X|i) = \frac{1}{(2\pi)^{m/2}|\Sigma_i|^{1/2}}\ \exp\left\{-\tfrac{1}{2}\,(X - M_i)^T\ \Sigma_i^{-1}\,(X - M_i)\right\} \quad \text{for } i = 1, 2, \ldots, R \tag{10.18}$$

where

$$M_i = E\,[\mathcal{X}_i] = \left[m_1^i, m_2^i, \ldots, m_m^i\right]^T\ ;\ \text{where } \forall \mathcal{X}_i \in X \text{ belonging to category } i$$

and

$$\Sigma_i = E\left[(\mathcal{X}_i - M_i)\,(\mathcal{X}_i - M_i)^T\right] = \begin{bmatrix} \sigma_{11}^i & \sigma_{12}^i & \cdots & \sigma_{1m}^i \\ \sigma_{21}^i & \sigma_{22}^i & \cdots & \sigma_{2m}^i \\ \vdots & \vdots & \vdots & \vdots \\ \sigma_{m1}^i & \sigma_{m2}^i & \cdots & \sigma_{mm}^i \end{bmatrix}.$$

Now, the discriminant function $g_i(X)$ given by (10.17) can be computed by substituting the expression for $p(X|i)$ given by (10.18) and it is given by

$$g_i(X) = \log p_i - \tfrac{m}{2}\log 2\pi - \tfrac{1}{2}\log|\Sigma_i| - \tfrac{1}{2}\left[(X - M_i)^T \Sigma_i^{-1}(X - M_i)\right] \quad i = 1, 2, \ldots, R \tag{10.19}$$

where $p_i = p(i)$. In the above equation $\frac{m}{2}\log 2\pi$ is a constant and does not affect the performance of the discriminator. Then

$$g_i(X) = b_i - \frac{1}{2}\left[(X - M_i)^T \Sigma_i^{-1}(X - M_i)\right], \quad i = 1, \ldots, R \tag{10.20}$$

where

$$b_i = \log p_i - \frac{1}{2}\log|\Sigma_i|.$$

Now we will give the multivariate Gaussian classifier algorithm:

Multivariate Gaussian Classifier Algorithm:

1. Let $\mathcal{X}_i$ be the training set of all patterns belonging to category i.
2. Compute the mean M_i and Σ_i for all $i = 1, 2, \ldots, R$

$$M_i = E[\mathcal{X}_i] = \left[m_1^i, m_2^i, \ldots, m_m^i\right]^T$$

$$\Sigma_i = E\left[(\mathcal{X}_i - M_i)(\mathcal{X}_i - M_i)^T\right] = \begin{bmatrix} \sigma_{11}^i & \sigma_{12}^i & \cdots & \sigma_{1m}^i \\ \sigma_{21}^i & \sigma_{22}^i & \cdots & \sigma_{2m}^i \\ \vdots & \vdots & \vdots & \vdots \\ \sigma_{m1}^i & \sigma_{m2}^i & \cdots & \sigma_{mm}^i \end{bmatrix}.$$

3. For each test pattern X, compute

$$g_i(X) == b_i - \frac{1}{2}\left[(X - M_i)^T \Sigma_i^{-1}(X - M_i)\right], \quad i = 1, \ldots, R$$

4. Declare the test pattern X belonging to category 'i' for which

$$g_i(X) \geq g_j(X), \quad \text{for all } i, j \in \{1, 2, \ldots, R\}, i \neq j$$

We can apply the multivariate Gaussian (MVG) classifier in classifying vehicles into tracked, wheeled, airborne, etc., and we use it to classify targets as humans and animals in the target discrimination Chap. 11. The MVG classifier is one of the simplest classifiers and this is a widely used algorithm. There is even a MATLAB program called "classify," which is based on MVG algorithm.

In the MVG classifier, all the data are assumed to have normal distribution. In general, that is not the case.

10.5 Gaussian Mixture Model (GMM)

The goal here is to classify a given pattern $X = \{x_1, x_2, \ldots, x_m\}$ into several classes $C_j,\ j \in \{0, 1, \ldots, R\}$. Using the Bayes' rule

$$p\left(C_j|X\right) = \frac{p\left(X|C_j\right) p(C_j)}{p(X)} \tag{10.21}$$

where $p\left(C_j|X\right)$ is the probability of class C_j given the observation vector X. The probability $p\left(X|C_j\right)$ can be learned from the training set. Since $p(X)$ is same for all the classes $C_j, j \in \{0, 1, \ldots, R\}$, a maximum a posteriori classification can be obtained by maximizing

$$C_j = \max \arg_j \ p\left(X|C_j\right) p(C_j) \tag{10.22}$$

Hence the task is to parameterize and learn $p\left(X|C_j\right)$ and the prior probabilities $p(C_j)$. The GMM is a convenient way to represent $p\left(X|C_j\right)$.

$$p\left(X|C_j\right) = \sum_q w_{j,q}\, \mathcal{N}\left(X; \mu_{j,q}, \sigma_{j,q}\right) \tag{10.23}$$

where $\mathcal{N}\left(X; \mu_{j,q}, \sigma_{j,q}\right)$ is the Gaussian distribution with mean $\mu_{j,q}$ and variance $\sigma_{j,q}$ and $w_{j,q}$ is the weight associated with each distribution. So basically, in (10.23), the probability $p\left(X|C_j\right)$ is represented as a weighted sum of Gaussians. The parameters $w_{j,q}$, $\mu_{j,q}$ and $\sigma_{j,q}$ can be estimated using the Expectation Maximization (EM) algorithm. Figure 10.9 shows the GMM concept. In Chap. 4, we showed how the EM algorithm can be used to find the parameters for data collected using two

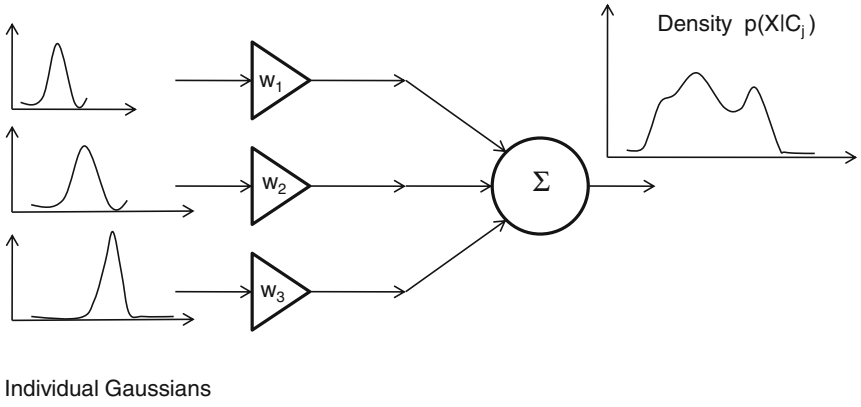


Fig. 10.9 Density function as a mixture of Gaussians

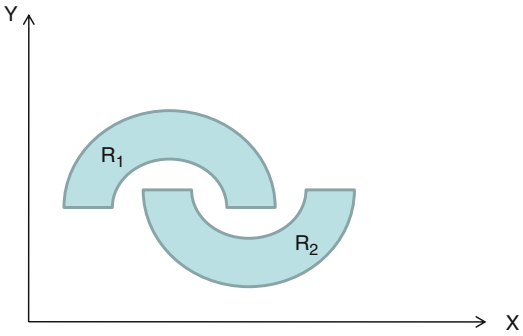


Fig. 10.10 Example of two regions intermingled

biased coins. The same EM algorithm can be used to find the parameters of all the distributions needed for the GMM.

The classifiers discussed so far assumed that a decision boundary can be a hyperplane that separates two regions clearly belonging to two classes. Some times it is not possible to use a simple hyperplane to separate two regions, as shown in Fig. 10.10. One may be able to apply a transformation of the data so that in the transform domain the data becomes separable by a hyperplane. Even for linearly separable data, it is not always clear where the hyperplane should be placed. Support vector machines (SVMs) present a technique for placing a hyperplane at an optimal location. In the next section, we present the theory behind the SVMs.

10.6 Support Vector Machines (SVM)

We have seen that the pattern classification is a process of mapping a given patten $\mathbf{x}_i = \left(x_1^i, x_2^i, \ldots, x_n^i\right)$ onto an output label

$$f(\mathbf{x}_i) = y_i. \tag{10.24}$$

Let us assume that there are 'l' number of training patterns. A classifier should be capable of learning the underlying probability distribution $P(X, y)$ from the training data consisting of patterns $X = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_l\}$ and its class label y_i, $i = 1, 2, \ldots, l$. Let $y_i = 1$ if it belongs to the class or $y_i = -1$. If the classifier is not designed properly, the classifier may not be able to generalize well for the unseen patterns. In general, there needs to be a balance between the training set and the capacity of the machine in order for the machine to learn without error. If the capacity of the machine is large, the machine memorizes the patterns (not learning) and misclassifies a pattern if it does not match one of the training patterns. If the capacity of the machine is too small, the machine will classify a new pattern as being part of the class with the maximum correlation. Thus neither machine generalizes well.

Let $f(X, w)$ be a mapping function [12] with a set of weights w that maps $X \longrightarrow y$, and in particular, for each individual vector $\mathbf{x}_i \longrightarrow y_i$. The training using the training set is the process of learning this mapping, that is, finding the right set of weights w. The expected error in classification is given by

$$R(w) = \frac{1}{2} \int |y - f(X, w)| \, dP(X, y) \tag{10.25}$$

While $R(w)$ is the theoretical expectation, in practice the machine is trained with a limited training data, which, in turn, results in empirical error

$$R_{\text{emp}}(w) = \frac{1}{2l} \sum_{i=1}^{l} |y_i - f(\mathbf{x}_i, w)| \tag{10.26}$$

The $\frac{1}{2} |y_i - f(\mathbf{x}_i, w)|$ is the loss and can take the values 0 or 1. Now, choose some η such that $0 \leq \eta \leq 1$. Then Vapnik [93] showed that for losses taking these values with probability $1 - \eta$, the following bound holds

$$R(w) \leq R_{\text{emp}}(w) + \sqrt{\left(\frac{h(\log(2l/h) + 1) - \log(\eta/4)}{l} \right)} \tag{10.27}$$

where h is the Vapnik-Chervonenkis (VC) dimension and is a measure of the capacity of the machine. The term in the square-root in the above equation is called the VC confidence. The goal of the training is for a chosen η minimizing the right side of (10.27). This approach results in a robust machine. Thus one has to come up with a class of functions whose capacity can be computed. Support vector classifiers are based on the class of hyperplanes.

10.6.1 Linearly Separable Data

As an example, let us consider a linearly separable data and the data need to be classified as '+1' if the data are to the right of the decision boundary given by

$$w^T X + b = 0; \quad w \in R^n, \; b \in R, \tag{10.28}$$

and classified as '−1' if the data lies to the left of the boundary as shown in Fig. 10.11. The corresponding decision functions are

$$f(X) = sign\left(w^T X + b\right). \tag{10.29}$$

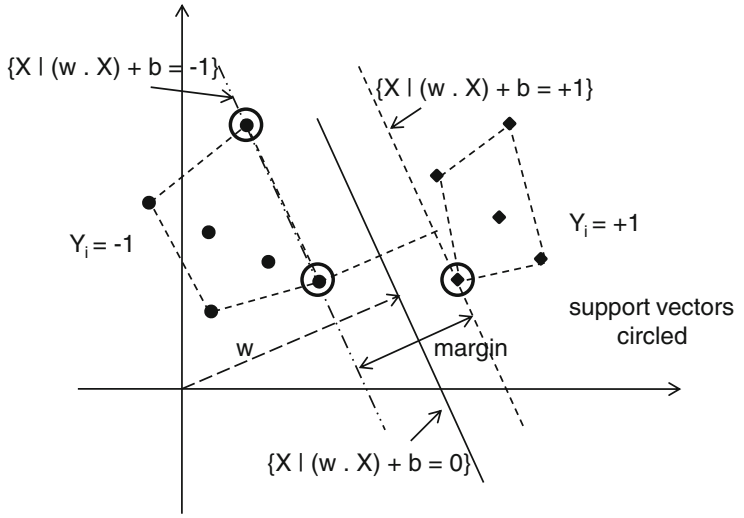


Fig. 10.11 Support vectors and margin

Thus checking whether the data were classified correctly or not implies checking

$$y\left(w^T X + b\right) \geq 1\ ,$$

since both $y > 0$ and $w^T X + b > 0$ for the data points on the right side of the decision boundary and $y < 0$ and $w^T X + b < 0$ for those on the left of the decision boundary. In general, there is some space between the decision boundary and the nearest points of either class. Let us rescale the data such that anything on or to the right of the boundary

$$w^T X + b = 1 \tag{10.30}$$

is of one class with label '1' and all the points on or to the left of the boundary

$$w^T X + b = -1 \tag{10.31}$$

is of other class with label '-1'. We now estimate the distance between the two boundaries given by (10.30) and (10.31). Notice that these two boundaries are parallel to each other. Let $\mathbf{x}_1$ is a point on the left boundary given by (10.31). Then the closest point on the line given by (10.30) is the point $\mathbf{x}_2 = \mathbf{x}_1 + \lambda w$, where λ is some scalar, since the closest point must lie on the perpendicular. Note that w is perpendicular to both the boundaries. Hence, λw is the segment that is connecting both points $\mathbf{x}_1$ and $\mathbf{x}_2$ and $\lambda \|w\|$ is the distance between the two points $\mathbf{x}_1$ and $\mathbf{x}_2$ and is the shortest distance.

Let us now solve for λ: since $\mathbf{x}_2$ is on the right boundary, we have

$$w^T \mathbf{x}_2 + b = 1$$

but $\mathbf{x}_2 = \mathbf{x}_1 + \lambda w$, then the above equation becomes

$$\begin{aligned} w^T (\mathbf{x}_1 + \lambda w) + b &= 1 \\ w^T \mathbf{x}_1 + b + \lambda w^T w &= 1. \end{aligned}$$

Substituting $w^T \mathbf{x}_1 + b = -1$, in the above equation gives

$$\lambda w^T w = 2$$

Hence

$$\lambda = \frac{2}{w^T w} = \frac{2}{\|w\|^2},$$

and the distance $\lambda \|w\|$ is

$$\lambda \|w\| = \frac{2}{\|w\|^2} \|w\| = \frac{2}{\|w\|} = \frac{2}{\sqrt{w^T w}}. \tag{10.32}$$

In order to make sure that no classification errors occur, we would like the distance between the two boundaries to be as large as possible. This distance is called the margin and we prefer to obtain the *maximal margin*.

Hence, we want to maximize the distance $\frac{2}{\sqrt{w^T w}}$ or minimize $\frac{\sqrt{w^T w}}{2}$, which is same as minimizing $\frac{w^T w}{2}$. This is a constrained optimization problem expressed as

$$\begin{aligned} &\min_{w,b} \ \frac{w^T w}{2} \\ &\text{subject to :} \ \ y_i \left(w^T \mathbf{x}_i + b\right) \geq 1; \quad \forall \mathbf{x}_i \end{aligned} \tag{10.33}$$

We now solve the constrained optimization problem. This is done by using the Lagrange multiplier technique. The Lagrangian is

$$\mathcal{L} = \frac{w^T w}{2} + \sum_{i=1}^{l} \lambda_i \left(1 - y_i \left(w^T \mathbf{x}_i + b\right)\right) \tag{10.34}$$

where λ_i are the Lagrangian multipliers. Setting the gradient of $\mathcal{L}$ with respect to w and b, we have

$$\frac{\partial \mathcal{L}}{\partial w} = w + \sum_{i=1}^{l} \lambda_i \left(-y_i\right) \mathbf{x}_i = 0.$$

Hence

$$w = \sum_{i=1}^{l} \lambda_i y_i \mathbf{x}_i . \tag{10.35}$$

Now taking the derivative with respect to b, we have

$$\frac{\partial \mathcal{L}}{\partial b} = \sum_{i=1}^{l} \lambda_i y_i = 0,$$

hence

$$\sum_{i=1}^{l} \lambda_i y_i = 0. \tag{10.36}$$

Now substituting (10.35) in (10.34) we get

$$\begin{aligned}
\mathcal{L} &= \frac{1}{2} \sum_{i=1}^{l} \lambda_i y_i \mathbf{x}_i^T \sum_{j=1}^{l} \lambda_j y_j \mathbf{x}_j + \sum_{i=1}^{l} \lambda_i \left(1 - y_i \left(\sum_{j=1}^{l} \lambda_j y_j \mathbf{x}_j^T \mathbf{x}_i + b \right) \right) \\
&= \frac{1}{2} \sum_{i=1}^{l} \sum_{j=1}^{l} \lambda_i \lambda_j y_i y_j \mathbf{x}_i^T \mathbf{x}_j + \sum_{i=1}^{l} \lambda_i - \sum_{i=1}^{l} \lambda_i y_i \sum_{j=1}^{l} \lambda_j y_j \mathbf{x}_j^T \mathbf{x}_i - b \sum_{i=1}^{l} \lambda_i y_i \\
&= -\frac{1}{2} \sum_{i=1}^{l} \sum_{j=1}^{l} \lambda_i \lambda_j y_i y_j \mathbf{x}_i^T \mathbf{x}_j + \sum_{i=1}^{l} \lambda_i
\end{aligned} \tag{10.37}$$

where we used the fact that $\sum_{i=1}^{l} \lambda_i y_i = 0$. Equation (10.37) is a function of λ_i only. We started with the problem of maximizing $\frac{w^T w}{2}$. This can be achieved by maximizing

$$\begin{aligned}
&\max \quad w(\lambda) = \textstyle\sum_{i=1}^{l} \lambda_i - \frac{1}{2} \sum_{i=1}^{l} \sum_{j=1}^{l} \lambda_i \lambda_j y_i y_j \mathbf{x}_i^T \mathbf{x}_j \\
&\text{subject to} \quad \lambda_i \geq 0, \quad \textstyle\sum_{i=1}^{l} \lambda_i y_i = 0
\end{aligned} \tag{10.38}$$

This is called the dual problem, that is, if we know w, we know all λ_i and vice versa. Equation (10.38) is a quadratic programming problem. A global maximum of λ_i can always be found and w can be found by (10.35), that is,

$$w = \sum_{i=1}^{l} \lambda_i y_i \mathbf{x}_i .$$

It turns out that the majority of λ_i are zeroes, leaving few that are not. The corresponding $\mathbf{x}_i$ with non-zero λ_i are called support vectors. An example of support vectors is shown in Fig. 10.11. The decision boundary is determined only by the support vectors.

Let $V \subset X$ be the set of support vectors where $V = \{\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_q\}$ is the set of q support vectors and the corresponding $\lambda^v = \left\{\lambda_1^v, \lambda_2^v, \ldots, \lambda_q^v\right\}$, $y = \left\{y_1^v, y_2^v, \ldots, y_q^v\right\}$, then the boundary is determined by

$$w = \sum_{i=1}^{q} \lambda_i^v y_i^v \mathbf{v}_i \tag{10.39}$$

Now to classify a new data $\mathbf{z}$, we compute

$$w^T \mathbf{z} + b = \sum_{i=1}^{q} \lambda_i^v y_i^v \left(\mathbf{v}_i^T \mathbf{z}\right) + b = \begin{cases} > 0 & \text{Class 1} \\ \text{otherwise} & \text{Class 2} \end{cases} \tag{10.40}$$

The classification is done using support vectors alone and w is not formed explicitly.

The Non-separable Case

Now consider the case where the data are not perfectly linearly separable as shown in Fig. 10.12. It is possible that some of the data points are not labeled correctly, so it is necessary to allow some data points of one class to appear in the other class. Moreover, if the points are not separable, then estimation of λ_i using the dual Lagrangian would grow arbitrarily and does not converge. We can relax the constraints given by (10.30) and (10.31) by introducing further cost. This is done by introducing slack variables ϵ_i, $i = 1, 2, \ldots, l$ in the constraints [12].

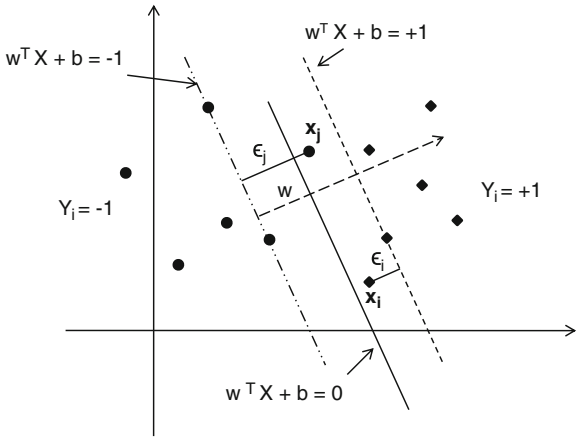


Fig. 10.12 Non-separable data

$$w^T \mathbf{x}_i + b \geq +1 - \epsilon_i \quad \text{for } y_i = +1 \tag{10.41}$$

$$w^T \mathbf{x}_i + b \leq -1 + \epsilon_i \quad \text{for } y_i = -1 \tag{10.42}$$

$$\epsilon_i \geq 0 \quad \forall i. \tag{10.43}$$

In order for an error to occur, the corresponding ϵ_i must exceed unity, so $\sum_i \epsilon_i$ is an upper bound on the number of training errors. The number of errors can be controlled by changing the objective function given in (10.33) to

$$\begin{aligned} &\min_{w,b} \ \frac{w^T w}{2} + C \sum_{i=1}^{l} \epsilon_i \\ &\text{subject to :} \ \ y_i \left(w^T \mathbf{x}_i + b \right) \geq 1 - \epsilon_i; \quad \epsilon_i \geq 0 \end{aligned} \tag{10.44}$$

where C is a parameter selected to control the penalty assigned to the error. A higher value of C implies a higher penalty for the errors. The dual representation of this new constrained optimization problem is

$$\begin{aligned} &\max \ \ w(\lambda) = \sum_{i=1}^{l} \lambda_i - \frac{1}{2} \sum_{i=1}^{l} \sum_{j=1}^{l} \lambda_i \lambda_j y_i y_j \mathbf{x}_i^T \mathbf{x}_j \\ &\text{subject to} \ \ C \geq \lambda_i \geq 0, \quad \sum_{i=1}^{l} \lambda_i y_i = 0 \end{aligned} \tag{10.45}$$

The solution can be found using a quadratic programming solver. Once again, w can be obtained using (10.39).

10.6.2 Nonlinear Decision Boundary

In real-world applications, the data often are not clearly separable, as in Fig. 10.11. However, it is possible to project the data into a higher dimensional dot product space where it becomes linearly separable. This higher dimensional space is called the feature space. Let $\Phi(X)$ be one such function that maps a given vector X into the feature space. Figure 10.13 shows the mapping. The formulation of quadratic programming in the feature space is similar to the one with the slack variables and is given by

$$\begin{aligned} &\max \ \ w(\lambda) = \sum_{i=1}^{l} \lambda_i - \frac{1}{2} \sum_{i=1}^{l} \sum_{j=1}^{l} \lambda_i \lambda_j y_i y_j \Phi(\mathbf{x}_i)^T \Phi\left(\mathbf{x}_j\right) \\ &\text{subject to} \ \ C \geq \lambda_i \geq 0, \quad \sum_{i=1}^{l} \lambda_i y_i = 0 \end{aligned} \tag{10.46}$$

Note that $\Phi(\mathbf{x}_i)^T \Phi\left(\mathbf{x}_j\right)$ appears in the above equation. Estimation of $\Phi(\mathbf{x}_i)$ is computationally costly. Now if there exists a "kernel function" K such that $K(\mathbf{x}_i, \mathbf{x}_j) = \Phi(\mathbf{x}_i) \cdot \Phi\left(\mathbf{x}_j\right) = \Phi(\mathbf{x}_i)^T \Phi\left(\mathbf{x}_j\right)$, then we only need to use K in

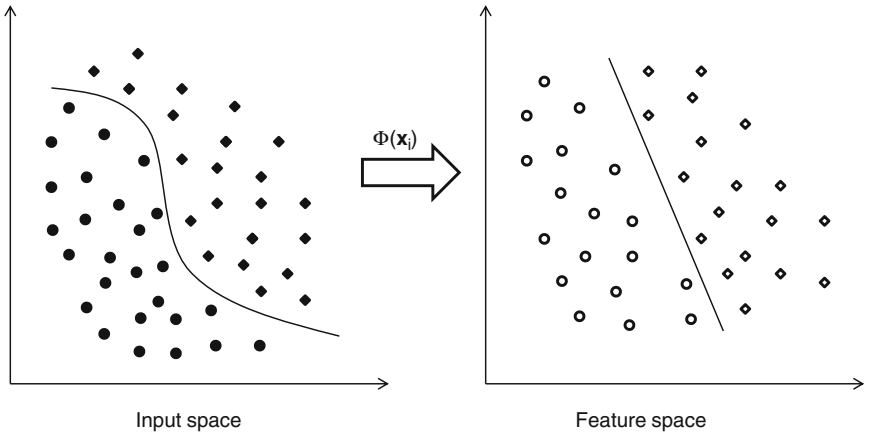


Fig. 10.13 Non-separable data transformed to separable data

the training algorithm, that is, in (10.46) and it does not need to know what Φ is explicitly [12, 47, 94]. One such mapping for a 2-D case is

$$\Phi(\mathbf{X}) = \begin{bmatrix} x_1^2 \\ \sqrt{2}x_1x_2 \\ x_2^2 \end{bmatrix} \tag{10.47}$$

and the kernel function

$$K(\mathbf{X}, \mathbf{Y}) = \Phi(\mathbf{X})^T \Phi(\mathbf{Y}) = (x_1y_1 + x_2y_2)^2 \tag{10.48}$$

The use of kernel function to avoid implementing Φ is known as the *kernel trick*. Using the kernel function, the optimization problem in (10.46) becomes

$$\begin{aligned} &\max \quad w(\lambda) = \textstyle\sum_{i=1}^{l}\lambda_i - \frac{1}{2}\sum_{i=1}^{l}\sum_{j=1}^{l}\lambda_i\lambda_j y_i y_j K\left(\mathbf{x}_i, \mathbf{x}_j\right) \\ &\text{subject to} \quad C \geq \lambda_i \geq 0, \quad \textstyle\sum_{i=1}^{l}\lambda_i y_i = 0 \end{aligned} \tag{10.49}$$

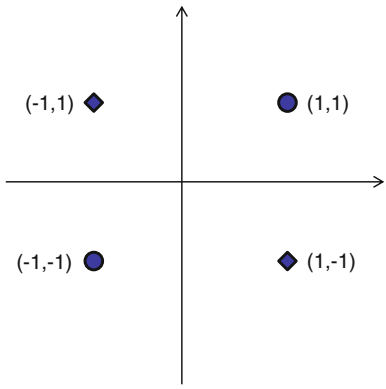
In order to classify new data, $\mathbf{z}$, compute as in the case of linear separable case using (10.39) and (10.40), that is,

$$w = \sum_{i=1}^{q} \lambda_i^v y_i^v \mathbf{v}_i \tag{10.50}$$

$$w^T\Phi(\mathbf{z}) + b = \sum_{i=1}^{q} \lambda_i^v y_i^v K(\mathbf{v}_i, \mathbf{z}) + b = \begin{cases} > 0 & \text{Class 1} \\ \text{otherwise} & \text{Class 2} \end{cases} \tag{10.51}$$

where $\mathbf{v}_i$ are the support vectors.

Fig. 10.14 XOR problem



Some of the kernel functions used are

- Polynomial kernel with degree d

$$K(\mathbf{x}, \mathbf{y}) = \left(\mathbf{x}^T \mathbf{y} + 1\right)^d \tag{10.52}$$

- Radial basis function kernel with width σ

$$K(\mathbf{x}, \mathbf{y}) = \exp\left(-\|\mathbf{x} - \mathbf{y}\|^2 / \left(2\sigma^2\right)\right) \tag{10.53}$$

- Sigmoid kernel function with κ and θ

$$K(\mathbf{x}, \mathbf{y}) = \tanh\left(\kappa \mathbf{x}^T \mathbf{y} + \theta\right) \tag{10.54}$$

Example: Let us consider the XOR problem shown in Fig. 10.14 that is given by

X	x_1	x_2	y_i
$\mathbf{x}_1$	-1	-1	-1
$\mathbf{x}_2$	-1	1	1
$\mathbf{x}_3$	1	-1	1
$\mathbf{x}_4$	1	1	-1

This is a classic problem and cannot be solved by a linear machine, that is, we cannot draw a line that separates the two classes. Here we have to project the problem into a higher dimensional space. Let us use the polynomial kernel of degree 2

$$K(\mathbf{x}, \mathbf{y}) = \left(\mathbf{x}^T \mathbf{y} + 1\right)^d$$

and a $C = 100$ for solving the constrained optimization problem, which is given by

$$\begin{aligned} &\max \ \ \textstyle\sum_{i=1}^{4}\lambda_i - \frac{1}{2}\lambda_i\lambda_j y_i y_j \left(\mathbf{x}_i\mathbf{x}_j + 1\right)^2 \\ &\text{subject to} \ \ C \geq \lambda_i \geq 0, \quad \textstyle\sum_{i=1}^{l}\lambda_i y_i = 0 \end{aligned}$$

We used a quadratic programming optimization tool "fmincon" in MATLAB [64] that performs constrained optimization and found that

$$\{\lambda_1, \lambda_2, \lambda_3, \lambda_4\} = \{0.99,\ 0.99,\ 0.99,\ 0.99\}$$

corresponding the four support vectors $\{\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, \mathbf{x}_4\}$. The discrimination function is given by (10.51) and is re-written here

$$\begin{aligned} f(\mathbf{z}) &= \textstyle\sum_{i=1}^{4}\lambda_i y_i K\left(\mathbf{x}_i, \mathbf{z}\right) + b \\ &= \lambda_1 y_1(-z_1 - z_2 + 1)^2 + \lambda_2 y_2(-z_1 + z_2 + 1)^2 \\ &\quad + \lambda_3 y_3(z_1 - z_2 + 1)^2 + \lambda_4 y_4(z_1 + z_2 + 1)^2 + b \\ &= \left(z_1^2 + z_2^2 + 1\right)\textstyle\sum_{i=1}^{4}\lambda_i y_i - 2z_1 z_2 \sum_{i=1}^{4}\lambda_i + 2z_1\left(\lambda_1 - \lambda_2 + \lambda_3 - \lambda_4\right) \\ &\quad + 2z_2\left(\lambda_1 + \lambda_2 - \lambda_3 - \lambda_4\right) + b \\ &= -2z_1 z_2 \textstyle\sum_{i=1}^{4}\lambda_i + b \end{aligned}$$

In order to solve for the variable "b", we solve the equations

$$\begin{aligned} f\left(\mathbf{z} = (-1, -1)\right) &= -2 * -1 * -1 * \textstyle\sum_{i=1}^{4}\lambda_i + b = -2 * \sum_{i=1}^{4}\lambda_i + b = -1 \\ f\left(\mathbf{z} = (-1, 1)\right) \ \ &= -2 * -1 * 1 * \textstyle\sum_{i=1}^{4}\lambda_i + b = 2 * \sum_{i=1}^{4}\lambda_i + b = 1 \end{aligned}$$

This gives the value $b = 0$. In order to test a given vector $\mathbf{z}$, compute

$$f(\mathbf{z}) = -k z_1 z_2 = \begin{cases} > 0 \text{ assign } 1 \\ < 0 \text{ assign } -1 \end{cases}$$

where $k = 2\sum_{i=1}^{4}\lambda_i$. Note that the constant k can be any value greater than zero, in other words, the product $-z_1 z_2$ determines the class.

There are several MATLAB programs available on SVMs for downloading on the internet. MATLAB also has its version on the SVM in one of the toolboxes they support.

10.7 Neural Networks

In late 1980s neural networks reemerged as a new tool in machine learning with principles based on the neural systems of biological entities. Ever since, there had been a tremendous growth in neural network architectures and new paradigms governing them to solve various problems. Neural networks are emerging as the primary tool for big data analysis in the decade of 2010, as other standard techniques used for processing big data are extremely slow.

It is known that an adult human brain consists of ∼86 billion neurons. Each neuron is interconnected with others through a synaptic contact. The synaptic contact weight determines the learning. Figure 10.15 shows a typical neuron with a receiving and transmitting dendrite. Each neuron transmits a series of pulses depending the input it receives. It assigns more importance by increasing the synaptic weight if a particular cell's input is to be given more weight or reduces the synaptic weight if the information is to be given less consideration. The synaptic weights are constantly modified to reflect the learning process over time.

Neuron Model

Based on our understanding of the biological neuron, it can be modeled as the system shown in Fig. 10.16. Hence, the output of the neuron is a function of the weighted sum of inputs. The function is usually a nonlinear function. Quite often a sigmoid function is used as an activation function of a neuron and it is given by

$$\varphi(v) = \frac{1}{1 + \exp(-\alpha(v - \theta))}, \tag{10.55}$$

where v input to the neuron, θ threshold and α is the parameter that controls the slope of the sigmoid function.

$$v = \sum_{i=1}^{n} w_{ki} x_i .$$

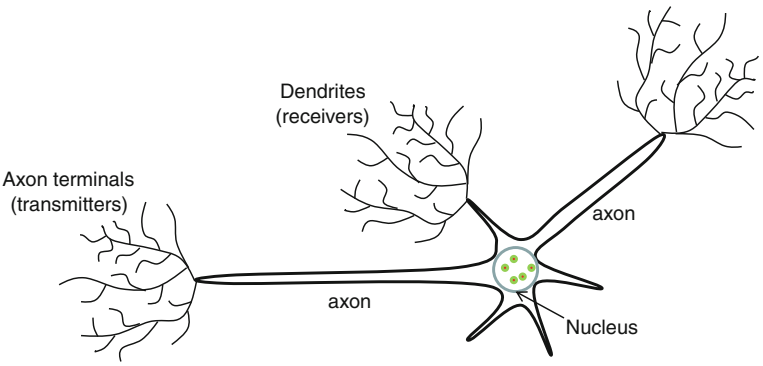


Fig. 10.15 Typical neuron

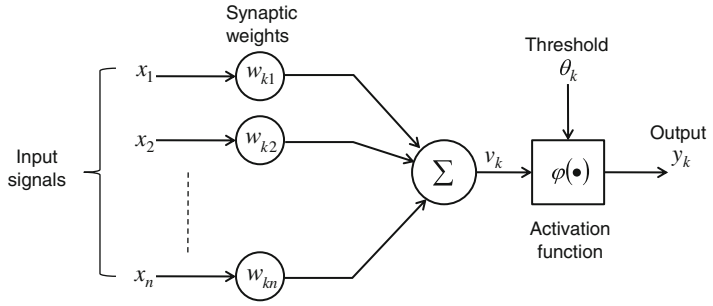


Fig. 10.16 Neuron model

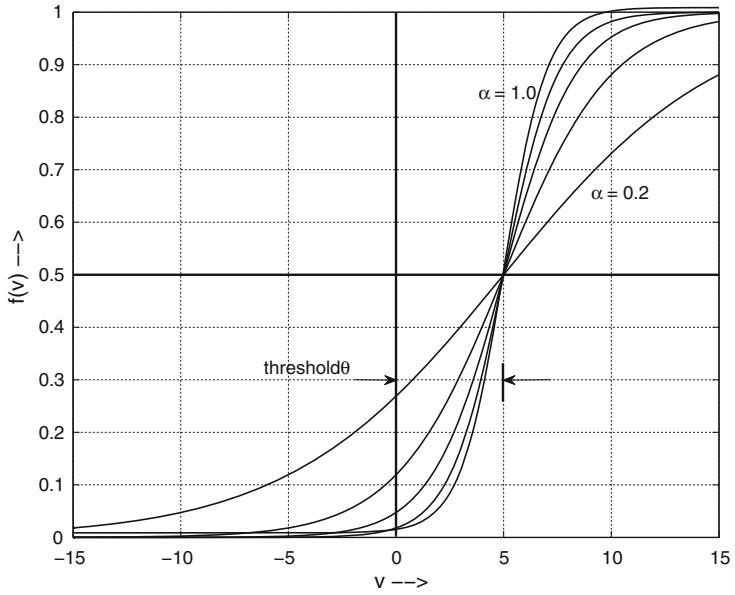


Fig. 10.17 Sigmoid function output

The output of the sigmoid function for different values of α are shown in Fig. 10.17. Clearly, the sigmoid function is a nonlinear function and its output lies in the range [0, 1]. Such a model is the binary sigmoid. In some applications, it is necessary that the activation function take both "$-ve$" and "$+ve$" values. One such function is the bipolar sigmoid function.

$$\varphi(v) = \frac{2}{1 + \exp(\alpha v)} - 1 \tag{10.56}$$

In general several nonlinear functions can be used as activation functions. For example, tanh is also used in some applications.

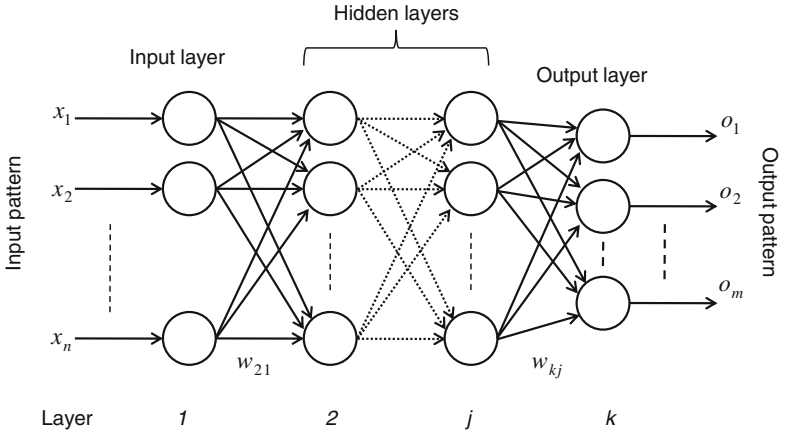


Fig. 10.18 Neural network architecture

$$\tanh\left(\frac{v}{2}\right) = \frac{1-\exp(-v)}{1+\exp(v)} \tag{10.57}$$

Now that we have a model for neurons, we can connect several neurons in some fashion and train them to perform pattern classification. One such a network is shown in Fig. 10.18 with an input layer, several hidden layers and an output layer. Each layer has several neurons. The number of neurons in a layer depends on the complexity of the problem. There are several books [40, 48, 72, 79] that allude to the issue of complexity in a problem, the required number of hidden layers and the number of neurons in each layer.

Training

The goal is to train a neural network to generate an expected known output at the output layer for a given input pattern. This is done in an iterative fashion, that is, each input pattern from a set of training patterns is applied one by one and the weights are adjusted to yield the expected output patterns. Although, there are several algorithms available for pattern classification in neural networks, in this chapter we concentrate on "backpropagation" algorithm. The backpropagation algorithm is similar to the algorithm developed by Bernard Widrow and his student Marcian Hoff [98] called "delta rule".

For the backpropagation algorithm development, we assume the following:

1. Neurons in the input layer have the activation function

$$\varphi(v) = v.$$

2. All other neurons have the sigmoid activation function given by (10.55).
3. The threshold is applied to a neuron with unity weight.

In order to train the network, a set of input vectors $\{X^1, X^2, \ldots, X^N\}$, with $X^i = x_1^i, x_2^i, \ldots, x_n^i$, where N is the number of patterns in the training set, are applied and the training is continued until the desired outputs $\{T^1, T^2, \ldots, T^N\}$, with $T^i = t_1^i, t_2^i, \ldots, t_m^i$, where n and m are the number of input and output neurons, respectively. The goal for training is to minimize the square of the error for qth pattern

$$E_q = \frac{1}{2}\sum_{j=1}^{m}\left(t_j^q - o_j^q\right)^2, \tag{10.58}$$

where $O^q = o_1^q, o_2^q, \ldots, o_m^q$ is the actual output of the net when X^q is applied and the average overall system error is given by

$$E = \frac{1}{2N}\sum_{q=1}^{N}\sum_{j=1}^{m}\left(t_j^q - o_j^q\right)^2. \tag{10.59}$$

We adjust the weights of the neurons to minimize the error in (10.58). This is done by taking the derivative of E_q with respect to the weight w_{kj} of the jth neuron connected to the kth neuron in the next layer. The change in the weight should be proportional to

$$\Delta w_{kj} = -\rho\frac{\partial E}{\partial w_{kj}} \tag{10.60}$$

where ρ is the proportionality constant, also called the learning rate. It is possible to control the convergence of the network by using different values of ρ.

Let y_k be the output of node k, then from Fig. 10.16, we have

$$y_k = \varphi(v_k), \tag{10.61}$$

where

$$v_k = \sum_j w_{kj} o_j$$

where o_j's are the outputs of the lower level layer. Then the term $\frac{\partial E}{\partial w_{kj}}$ can be evaluated as

$$\frac{\partial E}{\partial w_{kj}} = \frac{\partial E}{\partial v_k}\frac{\partial v_k}{\partial w_{kj}} = \frac{\partial E}{\partial v_k} o_j$$

Substituting the above equation in (10.60), we get

$$\Delta w_{kj} = -\rho\frac{\partial E}{\partial v_k} o_j. \tag{10.62}$$

Clearly, there are two cases exist for the kth neuron: (1) kth neuron belongs to the output layer and (2) kth neuron belongs to the hidden layer.

Case 1: Output Layer For the output layer, we can write

$$\frac{\partial E}{\partial v_k} = \frac{\partial E}{\partial o_k}\frac{\partial o_k}{\partial v_k} = -\left(t_k - o_k\right)\frac{\partial o_k}{\partial v_k}. \tag{10.63}$$

Since $o_k = \varphi(v_k)$,

$$\frac{\partial o_k}{\partial v_k} = \varphi'\left(v_k\right)$$

Then (10.63) becomes

$$\frac{\partial E}{\partial v_k} = -\left(t_k - o_k\right)\varphi'\left(v_k\right), \tag{10.64}$$

and from (10.62) the change in weight will be

$$\Delta w_{kj} = \rho\left(t_k - o_k\right)\varphi'\left(v_k\right)o_j. \tag{10.65}$$

If we incorporate the θ in (10.55) in the input v_k, and ignoring the constant α, then the $\varphi'\left(v_k\right)$ becomes

$$\varphi'\left(v_k\right) = \frac{\partial\varphi\left(v_k\right)}{\partial v_k} = \frac{\partial}{\partial v_k}\frac{1}{1+\exp\left(-v_k\right)} = \varphi\left(v_k\right)\left(1-\varphi\left(v_k\right)\right). \tag{10.66}$$

Then from (10.65) the weight connecting to the output layer adjusted as

$$\Delta w_{kj} = \rho\left(t_k - o_k\right)\varphi\left(v_k\right)\left(1-\varphi\left(v_k\right)\right)o_j. \tag{10.67}$$

All the variables in (10.67) are calculated and the whole expression is evaluated to get an update to the weight. This process is done for all the output neurons. From (10.67) it is clear that the change in w_{kj} is proportional to the following:

1. Difference between the expected output and the actual output $(t_k - o_k)$
2. Output of the hidden layer neuron o_j
3. Derivative of the activation function $\varphi'\left(v_k\right)$.

Now let us look at the hidden layer:

Hidden Layer

Let us assume the neuron is in the jth layer and the feeding layer is ith and the weight we are trying to adjust is w_{ji}. Then the change in the weight is given by (10.60)

$$\begin{aligned}\Delta w_{ji} &= -\rho\frac{\partial E}{\partial w_{ji}} = -\rho\frac{\partial E}{\partial v_j}\frac{\partial v_j}{\partial w_{ji}} = -\rho\frac{\partial E}{\partial v_j}o_i \\ &= -\rho\left(\frac{\partial E}{\partial o_j}\frac{\partial o_j}{\partial v_j}\right)o_i = -\rho\left(\frac{\partial E}{\partial o_j}\right)\varphi'\left(v_j\right)o_i\end{aligned} \tag{10.68}$$

Since, $\frac{\partial E}{\partial o_j}$ cannot be evaluated directly, we write it as

$$\frac{\partial E}{\partial o_j} = \sum_k \frac{\partial E}{\partial v_k} \frac{\partial v_k}{\partial o_j} \tag{10.69}$$

But

$$v_k = \sum_q w_{kq}\, o_q$$

hence,

$$\frac{\partial v_k}{\partial o_j} = w_{kj}.$$

Substituting this in (10.69), we get

$$\frac{\partial E}{\partial o_j} = \sum_k \frac{\partial E}{\partial v_k}\, w_{kj}. \tag{10.70}$$

Then (10.68) becomes

$$\Delta w_{ji} = -\rho \left(\frac{\partial E}{\partial o_j}\right) \varphi'\left(v_j\right) o_i = -\rho\, \varphi'\left(v_j\right) \sum_k \frac{\partial E}{\partial v_k}\, w_{kj}\, o_i. \tag{10.71}$$

We can substitute (10.64) for $\frac{\partial E}{\partial v_k}$ to evaluate Δw_{ji}. Hence, the error is backpropagated from the output layer to the lower level layers. To summarize the algorithm for updating the weights, we have

$$\Delta w_{kj} = \rho\,(t_k - o_k)\,\varphi\left(v_k\right)\left(1 - \varphi\left(v_k\right)\right) o_j; \quad \text{for output layer} \tag{10.72}$$

$$\Delta w_{ji} = -\rho\,\varphi\left(v_j\right)\left(1 - \varphi\left(v_j\right)\right) \sum_k \frac{\partial E}{\partial v_k}\, w_{kj}\, o_i; \quad \text{for hidden layer} \tag{10.73}$$

For each training pattern, the weights are updated according to the (10.72) and (10.73) until the error criteria are satisfied.

Prior to training, the neural network weights should be randomly assigned some values. It can be easily shown that if all the weights in the network are assigned same value, it is not possible to train the the network. It is shown in [62] that the single layer neurons provide the hyper planes, the hidden layer neurons group the hyper planes into closed regions, and subsequent layers cluster the closed regions into various classes for classification.

Although, here we presented the backpropagation algorithm there are several neural network architectures with their corresponding learning algorithms. Those interested in this should read some of the references provided in the book.

Chapter 11
Target Discrimination for Situational Awareness

Situational awareness is of paramount importance in a battlefield for successful operations. It is well known that whoever has the informational advantage tends to win the battles, mainly because they are in a better position to plan properly to counter the enemy's plans. In Chap. 9, we showed how mortar fire and sniper activity can be detected and localized to determine where the enemy's firepower is located. In this chapter, we focus on in detecting the enemy's vehicles and classifying them. We also discuss how to detect people with fewer false alarms and higher confidence. In particular, we examine personnel detection in detail, because it has become one of the main intelligence, surveillance and reconnaissance (ISR) requirements in urban operations, protection of secured facilities, border patrol, forward operating bases and perimeter protection. Traditionally, imaging sensors are used to detect targets as they provide an image of the target. However, surveillance of a facility or border over a prolonged period requires sensors to consume less energy, giving, non-imaging sensors an edge over the imaging sensors. Quite often, a suite of sensors of multiple modalities are packaged into a system called an unattended ground sensor (UGS). Then several of these UGSs are deployed in the field of operations in order to gather signatures for further analysis to determine the types of targets present.

Often, more than one sensor modality is required to detect targets and reduce false alarms. For example, if we use only microphones to listen for sounds to determine the presence of people in an area of surveillance, it fails if the people are silent or produce sounds sparingly. However, if they are walking around, the footfalls can be captured by seismic sensors. Some of the multimodal sensors used for surveillance are acoustic, seismic, passive infrared, magnetic, E-field, ultrasonic, micro-radar and profiling sensors, as shown in Fig. 11.1. Use of these sensors requires an understanding of the sensor phenomenology and the target signatures for proper algorithmic development and signal processing. Detections from all the sensors can be fused using fusion algorithms to achieve a high probability of detection with fewer false alarms and high confidence.

The majority of the detections in the field using existing commercial equipment result in nearly 20 % false alarms. As a result, development of robust algorithms using physics-based phenomenology is key to solving the problem. These algorithms

T. Damarla, *Battlefield Acoustics*, DOI 10.1007/978-3-319-16036-8_11

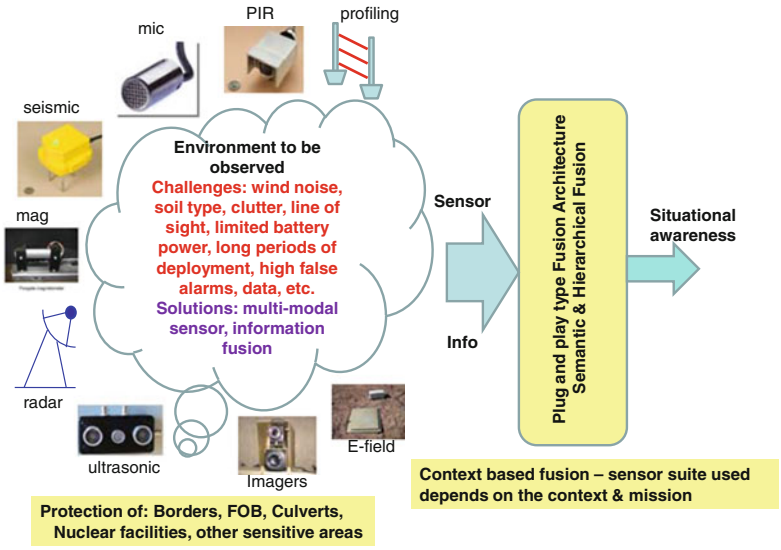


Fig. 11.1 Sensor suite and fusion to detect people

must take into consideration the power requirements, as well. Fusion algorithms and architectures that are appropriate for sensor networks to minimize the bandwidth and prolong the longevity of the deployed systems need to be developed. The following are descriptions of the types of sensors used in the sensor suite shown in Fig. 11.1.

Seismic: Seismic sensors are similar to acoustic sensors and are used to capture the vibrations in the ground such as (1) footfalls when people or animals are walking and (2) vibrations caused by vehicles traveling in the vicinity, to name just a few. Seismic sensors measure vibrations in all three axes, namely, the X, Y, and Z axes. The sensitivity of the sensors depends on the magnetic core and the number of turns used in a coil that moves in the core. The frequency response also varies from device to device. Figure 11.2 shows a picture of a seismic sensor. Since, seismic sensors are

Fig. 11.2 3-axis seismic sensor

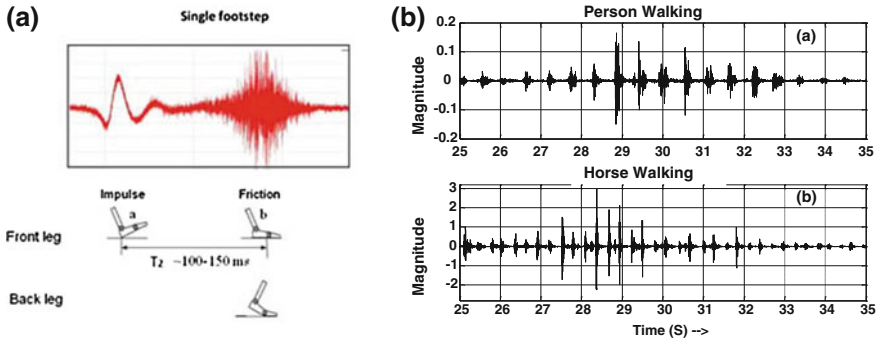


Fig. 11.3 **a** Single step signature, and **b** signatures of human and animal footfalls

primarily used to detect the footfalls of people in a personnel detection application, we present the signature of a single footfall of a person in Fig. 11.3a. Notice, when a person walks, the heel strikes the ground first giving the impulse shown in Fig. 11.3a, then the front of the foot slaps the ground and slides generating the high frequency part of the signature shown in Fig. 11.3a. Understanding of the phenomenology is key to generating robust algorithms for detecting people. Figure 11.3b shows the footfall signatures of a person and a horse. Animals have a different walking mechanism and the spectral content of animal footfall signatures are different and should be exploited in algorithmic development.

Seismic sensors can also be used to detect and classify vehicles since vehicles in motion produce vibrations. The vehicle signatures are similar to those of acoustic signatures and can be analyzed in a similar fashion.

Passive Infrared Sensor (PIR): These sensors use pyroelectric materials for sensing the thermal radiation emitted by warm bodies. A PIR sensor is shown in Fig. 11.4 with a Fresnel lens in front of it. Figure 11.5 shows a PIR sensor in protective packaging. The circuit diagram used with the sensor elements to observe the voltage generated by the pyroelectric sensing elements provided by one of the vendors is shown in Fig. 11.6. The pyroelectric sensing elements are connected back to back as shown in Fig. 11.6. Connecting the sensors back to back results in output that is proportional to the difference between the two sensing elements. Hence, in effect, it cancels out the variations due to atmospheric temperature changes. These elements are marked as positive and negative as shown in Fig. 11.4.

There are many different Fresnel lens arrays available. One type of Fresnel lens array is shown in Fig. 11.7. That Fresnel lens array shows five double beams, each separated by approximately 20°. Each beam pattern has two sub-beams; one corresponding to the positive element and another corresponding to the negative element.

The human body is a source of thermal radiation. The blackbody radiation pattern of a person at a temperature 37 °C is shown in Fig. 11.8. The maximum radiation from a human body occurs around the 9500 nm wavelength. Similar radiation patterns can be observed for animals. The pyroelectric sensors are optimized to detect radiation

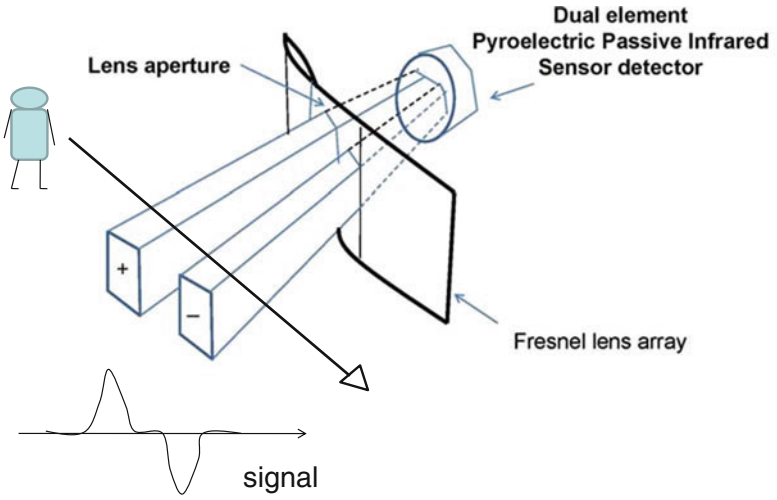


Fig. 11.4 Pyroelectric passive infrared detector with Fresnel lens array

Fig. 11.5 PIR Sensor in a protective case

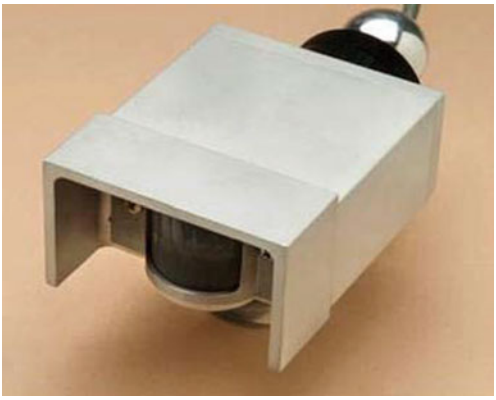
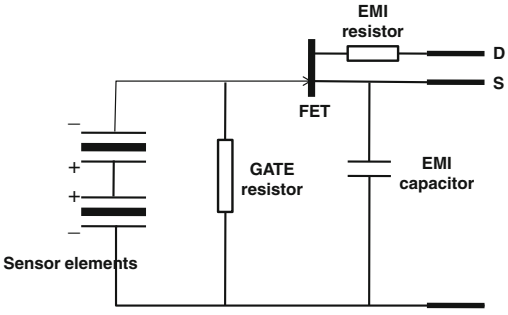


Fig. 11.6 Circuit diagram used to capture the output of the sensor



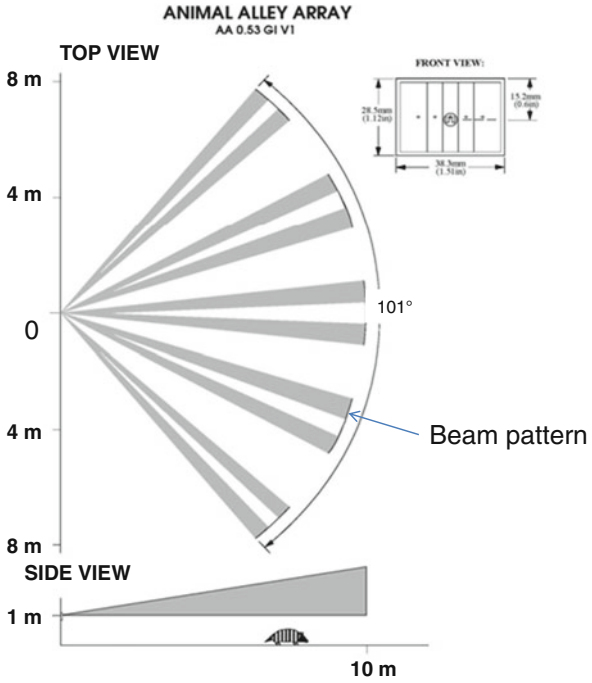


Fig. 11.7 A typical Fresnel array supplied by a vendor

from 8000 to 14,000 nm. When a heat source (source such as a human or an animal) moves across the beam patterns as shown in Fig. 11.4, an output voltage is generated. When the heat source moves into the positive element of the sensor, a positive voltage is generated at the output of the sensor. Similarly a negative voltage is generated when the heat source moves across the negative element of the sensor, as shown in Fig. 11.4.

Magnetic (B-field) Sensor: Magnetic sensors can be used to detect ferromagnetic materials carried by people, e.g., keys, firearms and knives. These sensors can also be used to detect vehicles. There are several types of magnetic sensor, namely, flux-gate magnetometer, coil type magnetic sensors, etc. The sensitivity of the sensors depends on the type of sensor. The flux-gate magnetometer has a sensitivity of 30 V/nT and is used to detect low frequency components up to 5 Hz. One can measure the magnetic field in all three axes. Figure 11.9 shows a picture of a flux-gate magnetometer.

Electrostatic (E-field) sensor: The E-field sensor consists of two parallel plates. One of the plates is grounded with a spike. The plates are charged, when a static field (a person has static field due to clothing rubbing together, etc.) comes near the plates, thus affecting the charge on the plates, which is, in turn, recorded. A D-dot E-field sensor is shown in Fig. 11.10.

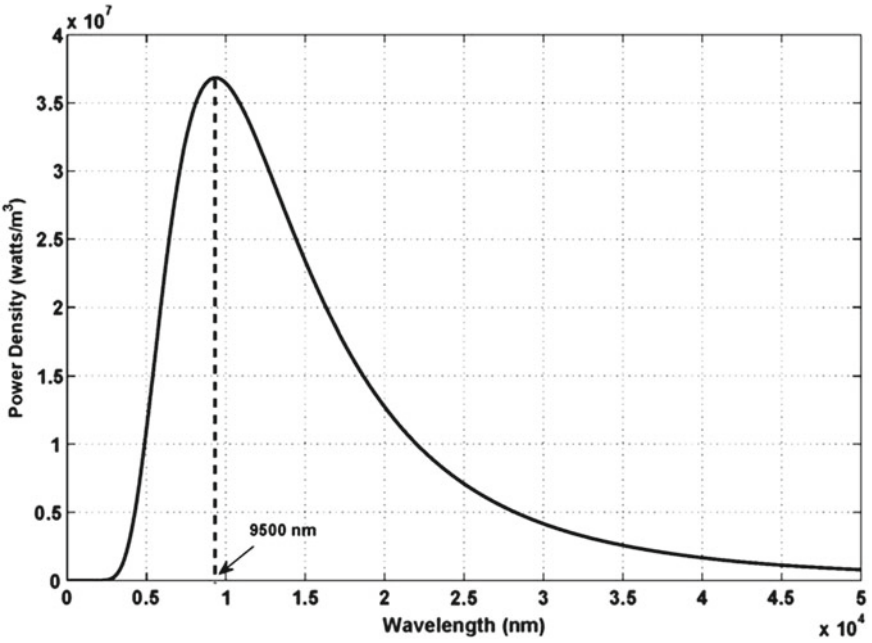
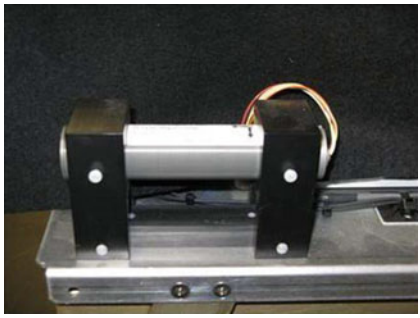


Fig. 11.8 Blackbody radiation curve of a human body at 37 °C

Fig. 11.9 Flux-gate magnetometer



Profiling Sensor: One of the disadvantages of an imaging sensor is that it captures several pixels (480 × 640) of an image requiring a large number of bytes of data to be transmitted to a command center for further analysis and action. This, in turn, requires a large communication bandwidth and drains more power to transmit the image. In order to determine if there is a person or not, one does not require the whole image; an outline of the image is sufficient. So capturing a silhouette of the target rather than the image of the target and transmitting that data to the command center reduces the bandwidth and also uses less power. A profiling sensor is shown in Fig. 11.11 with two vertical poles consisting of infrared (IR) transmitters mounted on one pole and receivers mounted on the other. A sample picture of a person generated by a profiling sensor is shown in Fig. 11.11.

Fig. 11.10 D-dot E-field sensor

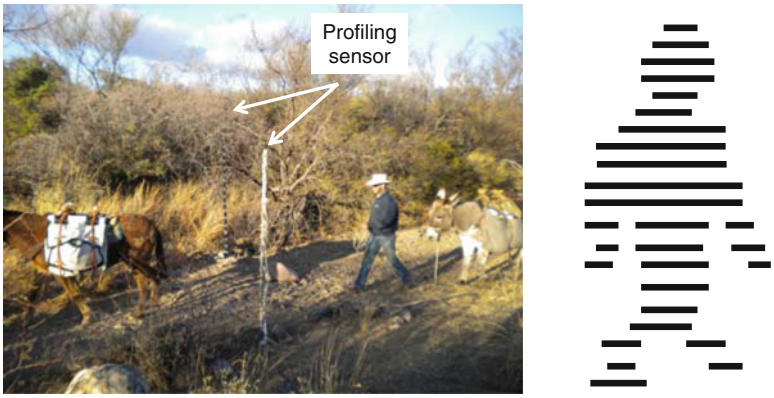


Fig. 11.11 Profiling sensor

Ultrasonic Sensor: The purpose of this sensor is to capture the micro-Doppler generated by various moving parts of a target. This sensor transmits an acoustic tone (ultrasound at 40 kHz) and receives signals scattered by the target just as in the case of radar. The received signal is analyzed to extract the Doppler due to the moving parts of the target. For example, human motion involves the motion of hands, legs and the torso. Their Doppler is normally different compared to the Doppler generated by the legs of animals. The sensor itself can be constructed using two 40 kHz ultrasonic transducers (MATSU/PAN EFR-RCB40K 54). These transducers have a typical bandwidth at −6 dB of 2 kHz and a directivity pattern at −6 dB of 55°. One of the transducers, acting as a transmitter, emits an ultrasonic wave, while the other acts as a receiver. Signals from the receiver and the transmitter can be recorded for processing. Figure 11.12 shows the ultrasonic sensor and the micro-Doppler extracted from the data. The figure also clearly shows the Doppler from various limbs of a person.

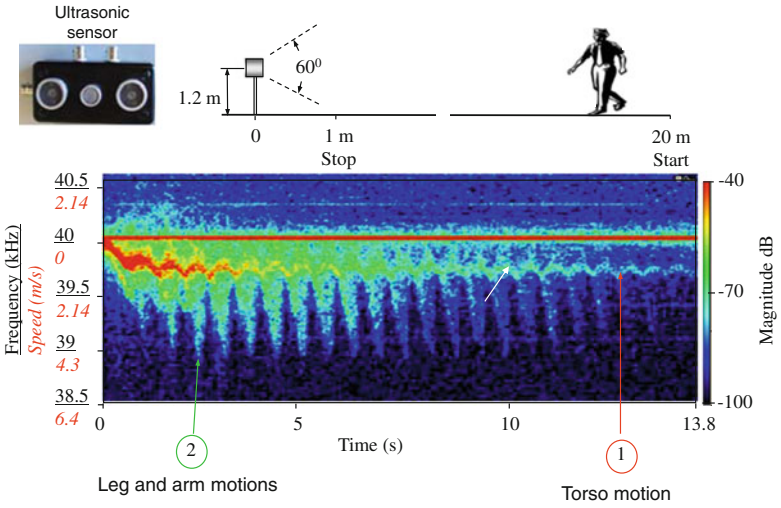


Fig. 11.12 Ultrasonic sensor

The other sensors in Fig. 11.1 are radar and imaging sensors. These are standard sensors whose descriptions are well documented in the literature.

Requirements for Situational Awareness:

Some of the requirements for better situational awareness are the following:

1. Detect, track and identify ground-based vehicles.
2. Detect, track and identify airborne vehicles.
3. Detect, localize and identify gun and mortar fire.
4. Detect people approaching forward operating bases.

In the previous chapters, we presented techniques to track vehicles and localize gun and mortar fire. In this chapter, we focus on detection and classification of vehicles. The techniques used for classifying ground as well as airborne vehicles are same, only the feature sets used are different. However, it is common practice to use radar for detection and tracking of airborne vehicles. Unlike vehicles, the detection of people is different and they are dealt separately. In the following section, we present the techniques for classifying ground vehicles using acoustic signatures. In this chapter, we address the issues involved in detecting and classification of vehicles and we also present the techniques used to detect people.

11.1 Vehicle Classification

For situational awareness, one should be able to detect and classify vehicles. Vehicle classification algorithms should be able to distinguish various types of vehicles. Classification of vehicles helps us understand the capability of the enemy and

Fig. 11.13 Different types of vehicles to be classified

strategize the responses. Some of the typical vehicle types to be classified are shown in Fig. 11.13 and include the following:

- Heavy vehicles
  - Tracked vehicles such as tanks
  - Wheeled vehicles
- Airborne vehicles such as helicopters and aeroplanes
- Light vehicles such as Humvees and civilian vehicles

It is clear that the list of vehicles to be classified is vast and their signatures also differ widely. In order to develop features for each type of vehicle, it is important to look into the physics-based phenomenology of the targets. The phenomenology would help us understand the type of signals emitted by each type of vehicle and hence help us extract a robust feature set that would result in correct classification with fewer miss detections and false alarms. To get a better feel for the phenomenology, first let us look at the spectrum of a typical acoustic signal generated by a tracked heavy vehicle in the Fig. 11.14. Clearly, there are harmonics present in the signal, which are seen as thick dark lines at some regular intervals of frequency. This is characteristic of a majority of vehicles with combustion engines with several pistons and rotating parts. Multiple pistons fire at regular intervals generating periodic impulses. This phenomenon results in multiple harmonics, which are seen in Fig. 11.14. The sound generated by the firing of the engine propagates through the body of the vehicle, which has its own resonance, and can act to suppress or enhance some other harmonics. The fundamental frequency clearly depends on the rate at which the pistons are fired and the number of cylinders. So the phenomenology we are looking for is the fundamental frequency and the harmonics of a given type of vehicle. The following table lists some of the characteristics to keep in mind while generating the feature sets: In Table 11.1, f_0 denotes the fundamental frequency. The number of harmonics to be selected depends on the vehicles, but 10 to 15 harmonics are a good number for a majority of vehicles. In the case of ground vehicles, the above feature set can be augmented with features extracted from seismic and magnetic sensor data for better classification. Using these features, one can easily classify the vehicles as: (a) heavy tracked vehicles, (b) light tracked vehicles, and (c) wheeled vehicles with 8, 6 or 4 cylinders.

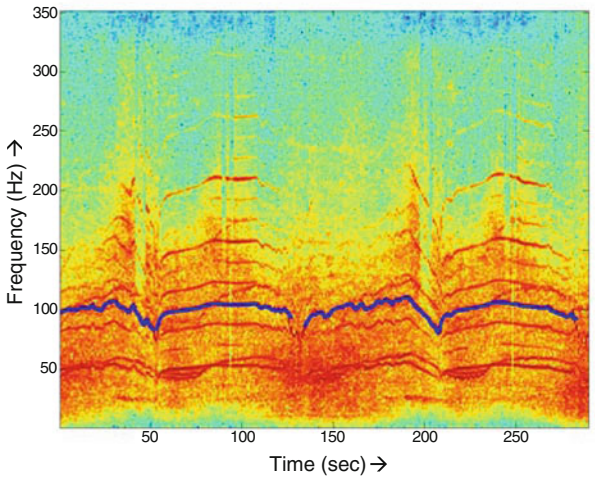


Fig. 11.14 Typical spectrogram of a heavy tracked vehicle

Table 11.1 Features from acoustic signatures of various vehicles

Type of vehicle	Feature	Features	Feature	Remarks
Heavy tracked vehicle	f_0	Magnitudes of 10–15 harmonics	Track slap freq.	Normalized for speed
Heavy wheeled vehicle	f_0	Magnitudes of 10–15 harmonics		Normalized for speed and load
Civilian vehicle	f_0	Magnitudes of 10–15 harmonics		Normalized for speed and load
Helicopters vehicle	f_0	Magnitudes of 10–15 harmonics	Tail rotor freq.	Normalized for speed and load

A spectrogram of helicopter data is shown in Fig. 11.15. The spectrogram is characterized by the harmonics that are generated by the main rotor blades rotating at a constant speed and the tail rotor. Analysis of helicopter data is similar to that of heavy tracked vehicle data.

In order to develop good algorithms, one should collect good data on all the vehicles to be classified. Some of the considerations that should be made while collecting the data are listed below.

Considerations for good Data Collection

In order for classification to be successful, one should collect good sets of data. In general, several precautions need to be taken while collecting the data.

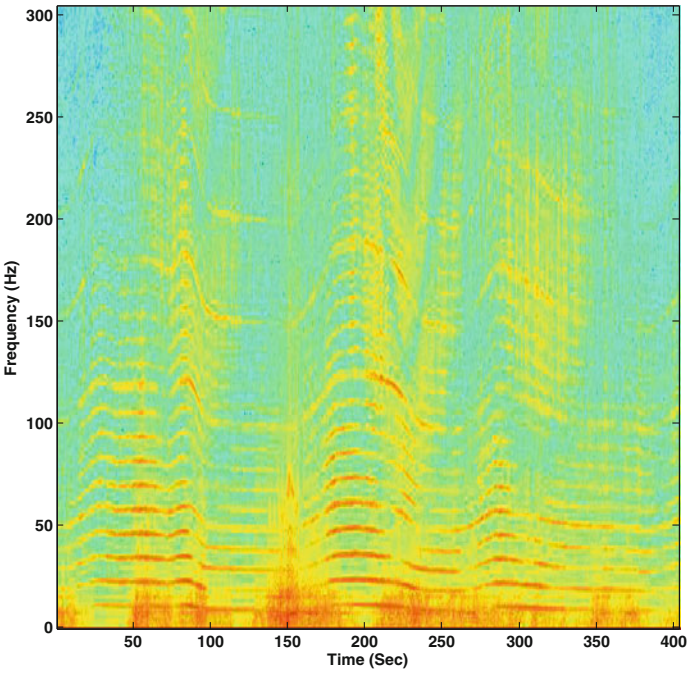


Fig. 11.15 Spectrogram of a typical helicopter data

- The sensor should be placed with the closest point of approach (CPA) of the vehicles at around 2–3 m.
- Whenever possible, one should use an array of microphones for data collection.
- Make sure the data are not saturated (especially for tracked and heavy wheeled vehicles.)
- Ensure a proper sampling rate.
- If multiple heterogenous sensor are used, they all should be collocated.
- All sensors should be properly calibrated prior to data collection.
- Collect ambient data to characterize the noise.
- Several sets of data should be collected for each type of vehicle at various speeds and loads to capture the variations that would naturally occur.

Since the vehicle data contains harmonics, a majority of the vehicle classifiers rely on harmonic analysis of the data in order to classify them. The magnitudes of various harmonics are used as features for classification. We now present the process involved in classifying vehicles. There are two phases: (a) training and (b) application.

Training a classifier:

In order to train a classifier, first select a classifier that is suitable for the task. Note that often one has to try several classifiers to find a suitable one for a given task. The classifier that is best suited is the one that best models the data [2, 25, 44]. A robust

training set should be generated for each type of vehicle to extract the parameters required by a classifier. Some of the tasks involved in generating training set are listed below.

Training Set Generation:

1. If an array of microphones is used, beamform (eg. delay and sum[1]) the data to improve SNR.
2. Use a sliding window of 1 s to get the 1 s data $x(t)$ and normalize.
3. Compute the fast Fourier transform of $x(t)$ to obtain $X(f)$.
4. Estimate and track the fundamental frequency f_0. The highlighted track in Fig. 11.14 is the 6th harmonic of the fundamental frequency. Note that it is not always easy to track the fundamental frequency. We track one of the strong harmonic frequencies and then infer the fundamental frequency from it.
5. Estimate the amplitudes of the harmonics $h_1, \ldots, h_m$, where h_k denotes the kth harmonic frequency. Then determine the feature vector $F = \{A_1, \ldots, A_m\}$, where A_k is the amplitude of harmonic frequency h_k.
6. Add the feature vector F to the training set pertaining to the target.

Repeat the above method for each vehicle. Make sure the data used for the training set generation has the relevant diversity to capture all the dynamics associated with the vehicle.

The parameter estimation needed for a classifier is generated from the training set. For example, for a multivariate Gaussian classifier presented in Chap. 10, we need to estimate the mean vectors and covariance matrices for each training set belonging to an individual vehicle. That is,

$$M_i = E\left[\mathcal{F}_i\right] = \left[m_1^i, m_2^i, \ldots, m_m^i\right]^T;$$

and

$$\Sigma_i = E\left[\left(\mathcal{F}_i - M_i\right)\left(\mathcal{F}_i - M_i\right)^T\right] = \begin{bmatrix} \sigma_{11}^i & \sigma_{12}^i & \cdots & \sigma_{1m}^i \\ \sigma_{21}^i & \sigma_{22}^i & \cdots & \sigma_{2m}^i \\ \vdots & \vdots & \vdots & \vdots \\ \sigma_{m1}^i & \sigma_{m2}^i & \cdots & \sigma_{mm}^i \end{bmatrix}.$$

Similarly, if one is using a SVM, one would generate the support vectors using quadratic programming optimization techniques. Next we discuss the application phase. Here, we generate a feature vector and apply it to the trained classifier for classification. The algorithm is given below.

Algorithm for vehicle classification using acoustic data:

1. If an array of microphones is used, beamform (eg. delay and sum) the data to improve SNR.
2. Use a sliding window of 1 s to get the 1 s data $x(t)$ and normalize.

[1] See Chap. 7 on Bearing Estimation using Acoustic Arrays.

3. Compute the fast Fourier transform of $x(t)$ to obtain $X(f)$.
4. Estimate and track the fundamental frequency f_0. The highlighted track in Fig. 11.14 is the 6th harmonic of the fundamental frequency. Note that, it is not always easy to track the fundamental frequency. We track one of the strong harmonic frequencies and then infer the fundamental frequency from it.
5. Estimate the amplitudes of the harmonics $h_1, \ldots, h_m$, where h_k denotes the kth harmonic frequency. Then determine the feature vector $F = \{A_1, \ldots, A_m\}$, where A_k is the amplitude of harmonic frequency h_k.
6. Use the feature vector F to classify the target using one of the classifiers discussed in the Chap. 10.

Notice that the procedures used for generating the training set and extracting the feature vector for testing are identical except for the last step. If they are not identical, the target models developed using the training set will be different from the model captured by the feature vector in the testing phase. We also note that the features extracted using harmonic analysis are widely used in classification of the target such as vehicles, helicopters and even airplanes.

There are several paper published on vehicle detection and classification in the literature. The survey paper [2] presents classification techniques and their capabilities.

Next we learn the techniques used in detecting people.

11.2 Detection of People Using Acoustic Signals

There are several ways one can detect the presence of a person depending on what the person is doing. If the person is talking, it is natural to use the voice signature of that person to detect the presence. However, there are other sources that generate various sounds, for example, natural sounds such as birds chirping and manmade sounds such as machine noises. So, it is imperative to clearly distinguish human voice from those generated by other sources. If the person is walking, we can use the sounds generated by the person walking or the combination of sounds generated by walking and talking. Using several approaches to detect a person's presence reduces the number of false alarms.

A typical human voice signature is shown in Fig. 11.16a and its spectrum in Fig. 11.16b.

11.2.1 Detection of Personnel Using Formants and Voice Modulation Characteristics

The mechanism of sound generation in humans is accomplished by an elaborate system called the "larynx". A typical picture of larynx is shown in Fig. 11.17.

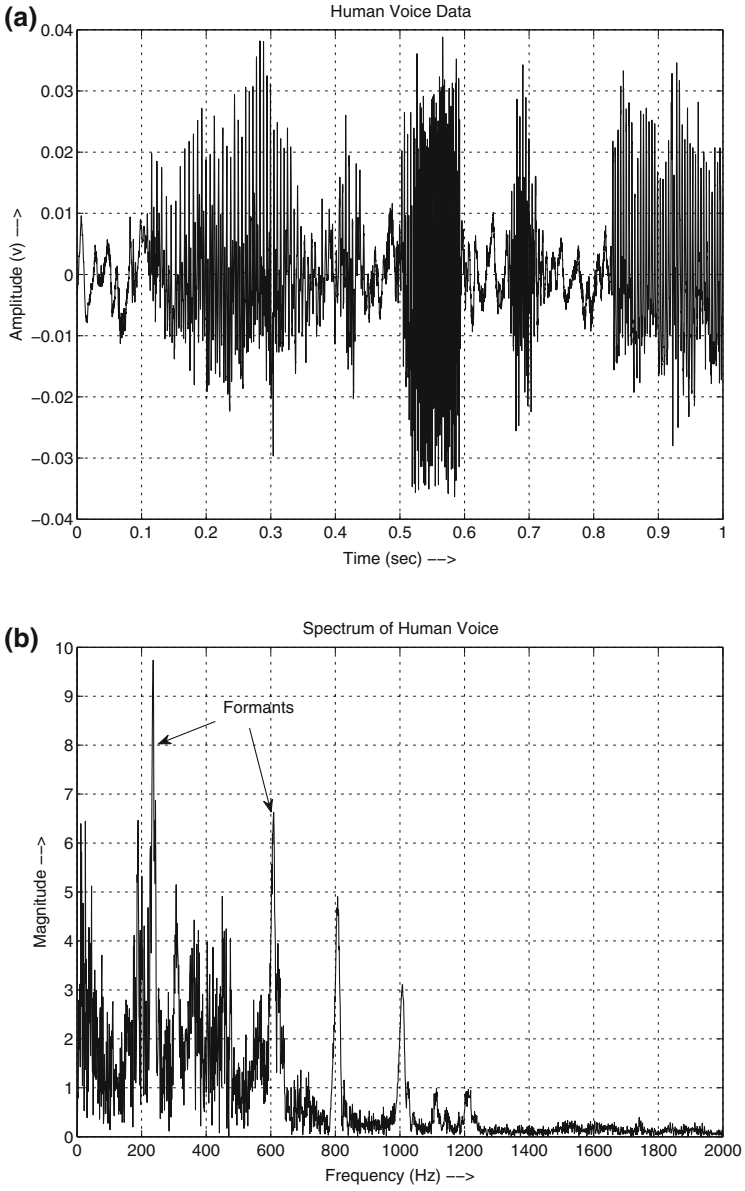


Fig. 11.16 **a** Typical human voice signature, **b** spectrum of human voice

The sound is generated by vocal cord vibrations, the opening and closing of lips, and vocal tract shaping. Figure 11.18 shows some of the components of the human sound generation process. The sound is generated through the interaction of the sound source and the time-varying-filter action of the vocal tract. More details can be found

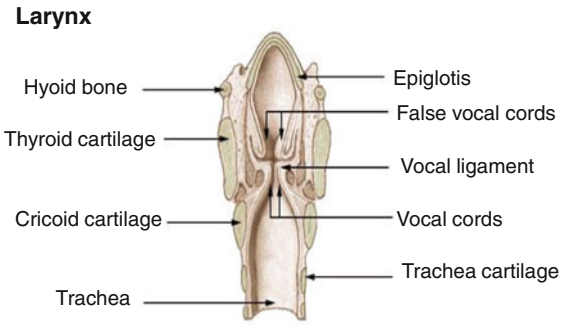


Fig. 11.17 Larynx and associated parts

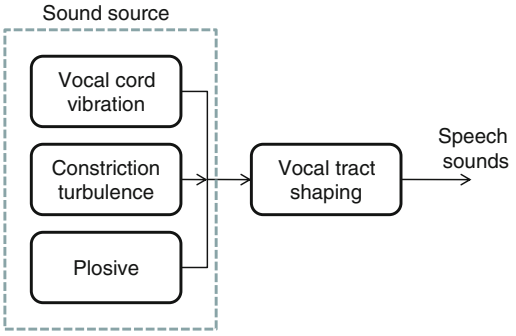


Fig. 11.18 Human sound production process

in [39, 89]. Each syllable generated by a human voice has an unique frequency associated with is called the "formant," as shown in Fig. 11.16b.

Acoustic data analysis for personnel detection is focused on (a) detection of formants characteristic to the human voice [18], (b) the energy distribution of the human voice in various spectral bands [18, 20] and (c) footstep detection for estimation of cadence.

Formants are specific frequencies associated with a person's voice. The formants lie between 125–800 Hz [77]. Whenever a person speaks, the vocal cords vibrate producing the formants, which are further modulated by the opening and closing of the vocal tract. In general, the voice emanating from a person can be modeled as an amplitude modulated signal [3] given by

$$s(t) = (A_c + A_m \sin(2\pi f_m)) \cos(2\pi f_c) \tag{11.1}$$

where 'f_c' and 'f_m' represent the carrier and modulating frequencies and A_c and A_m denote their magnitudes, respectively. There are three distinct frequencies in the signal, namely, f_c, $f_c - f_m$ and $f_c + f_m$. Formant detection involves detecting the carrier frequency with a frequency spread of $2f_m$. In practice, the spread is estimated by collecting several acoustic signals generated by several people talking and finding the carrier frequencies with the associated spread. The threshold on

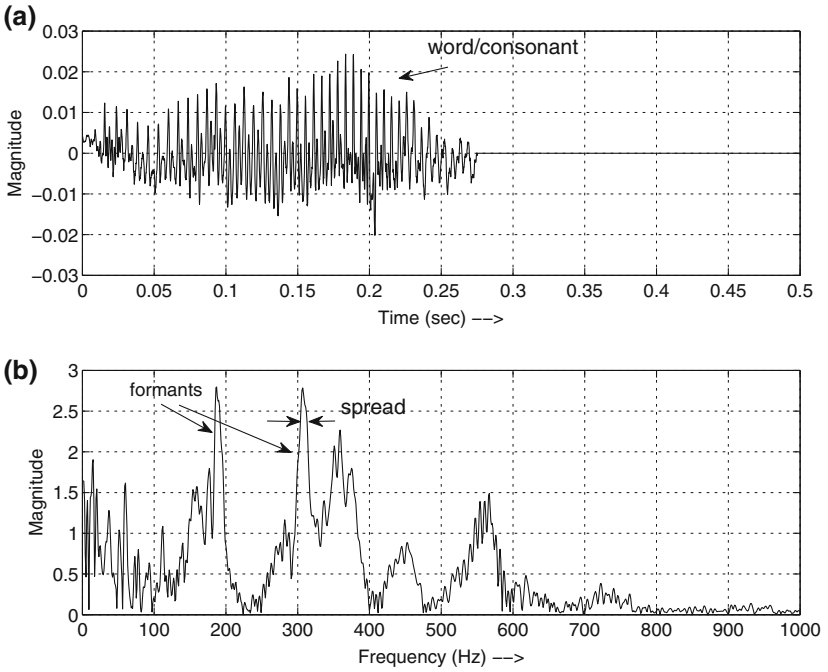


Fig. 11.19 Acoustic signal and its spectrum

the spread is determined statistically with the observed data. Figure 11.19 shows an acoustic signal and its spectrum clearly showing the formants and the spread.

Another feature of the human voice is that its energy is distributed in only 4–5 spectral bands [18]. Figure 11.19 clearly shows that the energy is concentrated in the first 1000 Hz and spread across a few bands.

11.2.2 Personnel Detection Using the Energy in Several Bands of the Voice Spectra

It is known [18, 20] that the human voice spans the 50 Hz–20 kHz frequency range. However, most of the energy is concentrated in 4–5 bands. These bands are 50–250 Hz, 251–500 Hz, 501–750 Hz and 751–1000 Hz. The energy levels in these bands are used as features to determine whether the sounds belong to a person or not. Let us denote the feature vector by $X = \{x_1, x_2, \ldots, x_m\}$, where x_i is the total energy in band 'i' and 'm' is the number of features. Feature vectors are used to classify whether the sounds belong to a human voice or not using a MVG classifier. We assume the energy levels in each band are statistically independent and have the Gaussian distribution given by

$$p(x_i) = \frac{1}{\sqrt{2\pi\sigma_i}} \exp\left\{-\frac{1}{2}\left(x_i - m_i\right)^T \sigma_i^{-1}\left(x_i - m_i\right)\right\}, \tag{11.2}$$

where m_i and σ_i are the mean and variances and T denotes the transpose. Then the m-variate normal probability density function $p(X|H_i)$ is given by

$$p(X|H_i) = \frac{1}{(2\pi)^{m/2}\,|\Sigma|^{-1/2}} \exp\left\{-\frac{1}{2}\left(X - M_i\right)^T \Sigma_i^{-1}\left(X - M_i\right)\right\}, \quad i \in \{0, 1\} \tag{11.3}$$

where the hypothesis H_0 and H_1 correspond to a person is not present and a person is present, respectively.

$$M_i = E\left[\mathcal{X}_i\right]_{\mathcal{X}_i \subset X} = \left[m_1^i, m_2^i, \ldots, m_m^i\right]$$

and the covariance

$$\Sigma_i = \begin{bmatrix} \sigma_{11}^i & \sigma_{12}^i & \cdots & \sigma_{1m}^i \\ \sigma_{21}^i & \sigma_{22}^i & \cdots & \sigma_{2m}^i \\ \vdots & \vdots & \vdots & \vdots \\ \sigma_{m1}^i & \sigma_{m2}^i & \cdots & \sigma_{mm}^i \end{bmatrix}.$$

In the event that the energy in each spectral band is independent of the others, the likelihood that a person is present is given by

$$p(X|H_1) = \prod_{i=1}^{m} p(x_i|H_1)p(H_1). \tag{11.4}$$

Then the posterior probability of human presence is given by

$$p(H_1|X) = \frac{\prod_{i=1}^{m} p(x_i|H_1)p(H_1)}{\prod_{i=1}^{m} p(x_i|H_0)p(H_0) + \prod_{i=1}^{m} p(x_i|H_1)p(H_1)}. \tag{11.5}$$

From the above equation, it is clear that the priors $p(H_0)$ and $p(H_1)$ are needed for computing the posterior. In general, we do not know the priors and hence it is a common practice to assume that $p(H_0) = p(H_1) = 0.5$. The posterior probability $p(H_1|X)$ for each new measurement X is computed and if it exceeds a particular threshold we declare it to be the sound of a human. The threshold is selected based on the training data and the expected false alarm rate. If the threshold is too high, it is possible to miss detecting a human presence, on the other hand, if the threshold is too small, we encounter a large number of false alarms.

11.2.3 Personnel Detection Using Cadence

Whenever a person or animal walks, the footfalls make audible sounds that are captured by the microphone along with the voice signal. Figure 11.20 shows acoustic signals consisting of voice as well as footstep sounds. Footsteps sounds are characterized by spikes generated by a person's heel striking the ground. Normally, a person walks with a rhythm called cadence. Cadence is defined as the average number of footfalls per second. One can analyze the signatures of human and animal footfalls and classify them into respective classes. It is estimated that the cadence of humans walking lies between 1 and 2 Hz while the cadence of animals walking is around 2.5–3 Hz. Several researchers [26, 49] have used the cadence as a way of detecting and distinguishing people from animals, which is the primary cause of false alarms. As mentioned earlier, these footfalls are impulsive in nature and result in several harmonics. Even if many people are walking in single file (on a path), they tend to synchronize their stride with others and walk more or less at the same cadence.

Figure 11.21 gives the flowchart for processing acoustic data. The acoustic data are first analyzed to determine the presence of a person using the energy in the spectral bands using MVG classifier. If the classifier gives the likelihood of a person greater than some threshold, the data are then further analyzed for the presence of formants. We also look for the presence of a person using cadence analysis. All three results

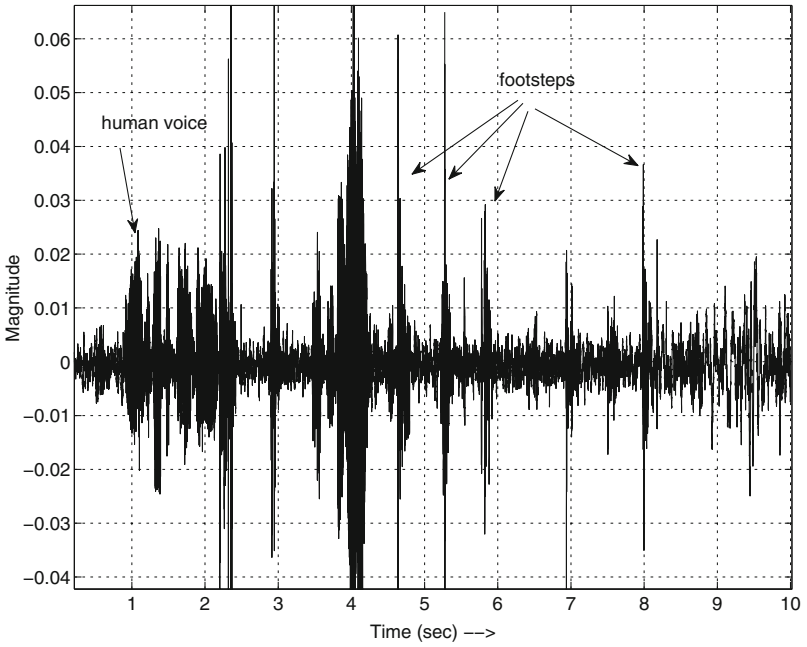


Fig. 11.20 Acoustic signal showing voice and footstep signals

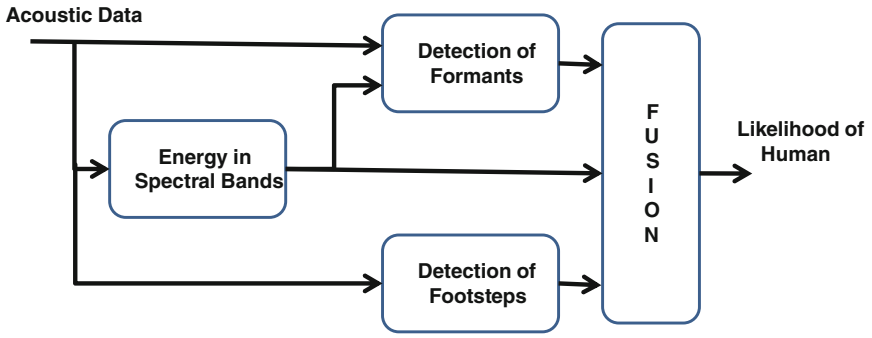


Fig. 11.21 Acoustic data processing

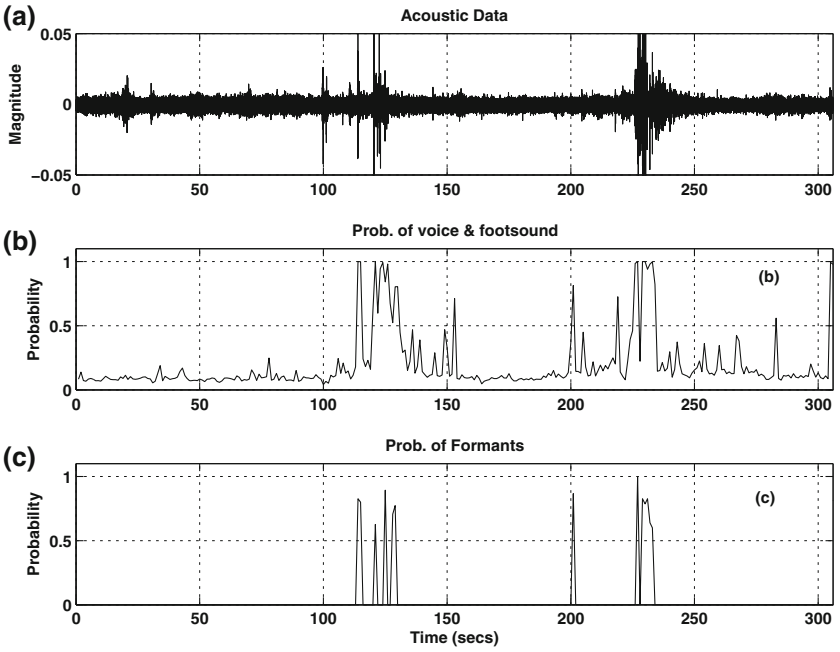


Fig. 11.22 **a** Acoustic data of a person walking, **b** probability of voice/foot sound using MVG classifier and **c** probability of formant detection

are fused using a Dempster-Shafer[2] fusion paradigm [20, 66], and the typical results are shown in Fig. 11.22. The top plot in Fig. 11.22 is the original acoustic data, the middle plot is the probability of detection of voice or footfall sound, and the bottom plot is the probability of detection of human voice by detecting formants. From the

[2] The Dempster-Shafer fusion algorithm is presented in the Chap. 12.

acoustic data plot we can see the impulses corresponding to the footfall sounds. The formant detection augments the fact that the sounds correspond to a person.

In the event, the cadence is difficult to estimate, one can use the harmonics generated by the footfalls as features and classify the targets [24].

11.2.4 Personnel Detection Using Harmonics of Footfalls

In order to estimate the harmonics of footfalls, 6 s of data are considered and the FFT of its envelop is computed [49]. The first 15 bins are used as the features for an MVG classifier. The sliding window of 6 s is moved by 1 s and the next 6 s of data are processed. The process is repeated until all the data are processed. Note that instead of MVG classifier one can use SVMs or any other classifiers discussed in the Chap. 10. Footstep detection using the various harmonics of cadence is shown in Fig. 11.23.

In order to increase the confidence in detection and reduce the number of false alarms, one uses more than one sensor modality. Quite often, a seismic sensor is used for personnel detection as it is the most robust and can be buried in the soil to avoid detection by others.

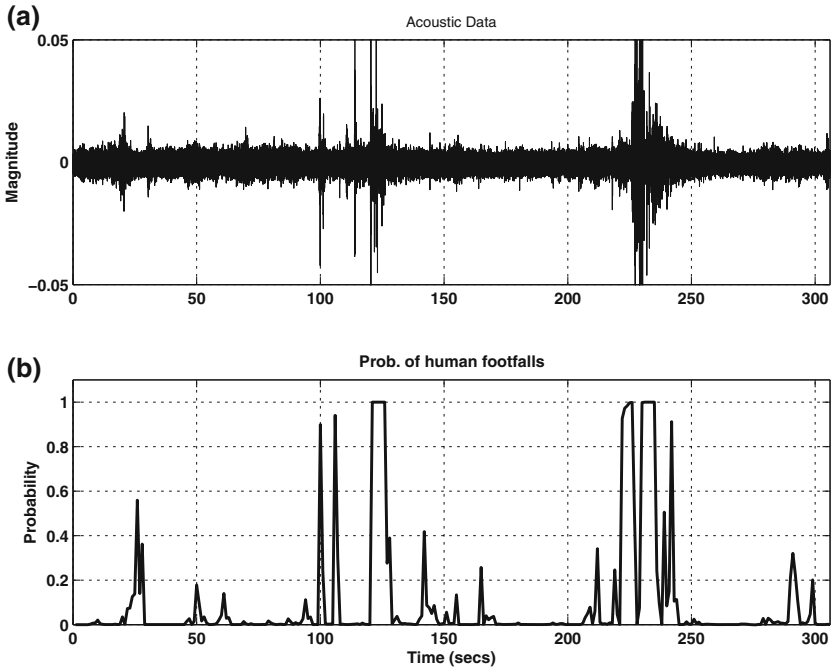


Fig. 11.23 **a** Acoustic data of a person walking and **b** probability of acoustic footstep detection

As mentioned earlier, using multiple sensors of multiple modalities would improve the detection and reduce the number of false alarms. For example, looking at Fig. 11.21, we notice that two of the three blocks will be rendered useless if there is no human voice present. The third block estimates the cadence and its harmonics. However, a detection based on cadence alone is prone to errors and results in high number of false alarms. In the next section, we present analysis of another acoustic sensor modality, namely, ultrasonic sensor, data for personnel detection.

11.3 Ultrasonic Sensor Data Analysis

In order to reduce the number of false alarms, one must use high-fidelity sensor data that captures the phenomenology associated with people and animals walking. In [21, 56, 101], ultrasonic sensors were used to get the micro-Doppler returns from an animal or a person, and classify them. In [56], classification was performed using GMMs (GMM,) where the signal distribution is assumed to be a mixture of Gaussian distributions. In [101], the cepstral coefficients are used as features and they used different classifiers such as SVM, GMM, etc., to classify the signatures. But the analysis in [50, 56, 101] is carried out purely from a signal processing point of view rather than a physics-based model analysis.

In this section, we use the model-based analysis of the Doppler returns from people and animals to extract features and classify them. The ultrasonic transducer used for this experiment emits a frequency of 40 kHz and receives the reflected signals from the target, just as in the case of a radar [42]. The micro-Doppler returns due to the swinging of the arms, legs and the torso of a person or an animal are analyzed for salient features. We use the features from these Doppler returns from the limbs to classify the targets. We show that when the targets are close to the sensor (Doppler returns with a high SNR), the spectrogram of the Doppler returns have a distinct structure identifying the returns from the limbs and torso. These distinct features extracted from this spectrogram can be used to distinguish people from animals. When the targets are far from the sensor, resulting in low SNR Doppler returns, the distinct features may not be apparent in the spectrogram. For the low SNR cases, more traditional signal processing techniques are used for classification purposes. Unlike the radar system, the ultrasonic transmitter and receiver are inexpensive, immune to electromagnetic interference and hard to detect.

11.3.1 Micro-Doppler Associated with Human and Animal Motion

Bradley et al. [10, 11] have explained the observed human-gait features in Doppler sonar grams by using the Boulic-Thalmann (BT) [9] model, shown in Fig. 11.24, to predict joint angle time histories and the temporal displacements of the body's

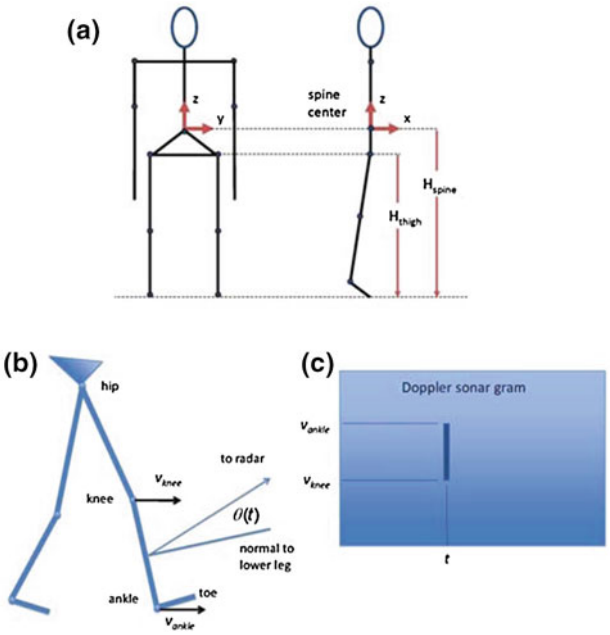


Fig. 11.24 **a** Stick model of a human body, **b** human body model as a compound pendulum and **c** Doppler sonar gram for the leg

center of mass. In the BT model, body segments are represented as ellipsoids. Temporally dependent velocities at the proximal and distal end of key body segments are determined from the BT model. Doppler sonar grams are computed by mapping velocity-time dependent spectral acoustic cross sections for the body segments onto the time-velocity space, mimicking the short-time Fourier transform (STFT) used in Doppler sonar processing. Figure 11.25a shows the estimated velocities using the model for various parts of the body for a 6-ft-tall person. The Doppler is related to the radial velocity v_r of the target and is given by $f_d = \frac{2v_r}{c} f_c$, where c is the propagation velocity of sound, and f_c is the radiated carrier frequency. The detailed computation of velocities of various limbs can be found in [10]. Similarly, models have been developed [11] for a quadruped, such as a horse. Figure 11.25b shows the estimated velocities for a horse.

Figure 11.26a shows the measured micro-Doppler returns for a person using a 40 kHz ultrasonic transducer after down-converting the carrier to the base-band. The solid line at 0 Hz corresponds to the transmitted signal (carrier) that leaked in to the receiver and the returns from stationary objects. The Doppler returns from the arms, legs and torso are clearly visible. The carrier at 0 Hz is digitally removed and the resultant Doppler signal is shown in Fig. 11.26b. These measurements also validate the model used. Further validation of the model is done in [65] by collecting data of a person walking on a treadmill.

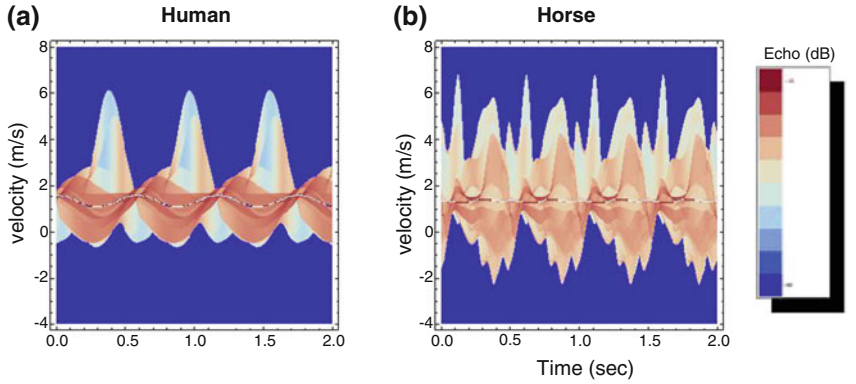


Fig. 11.25 Estimated velocity (Doppler) for **a** a person walking and **b** a horse walking

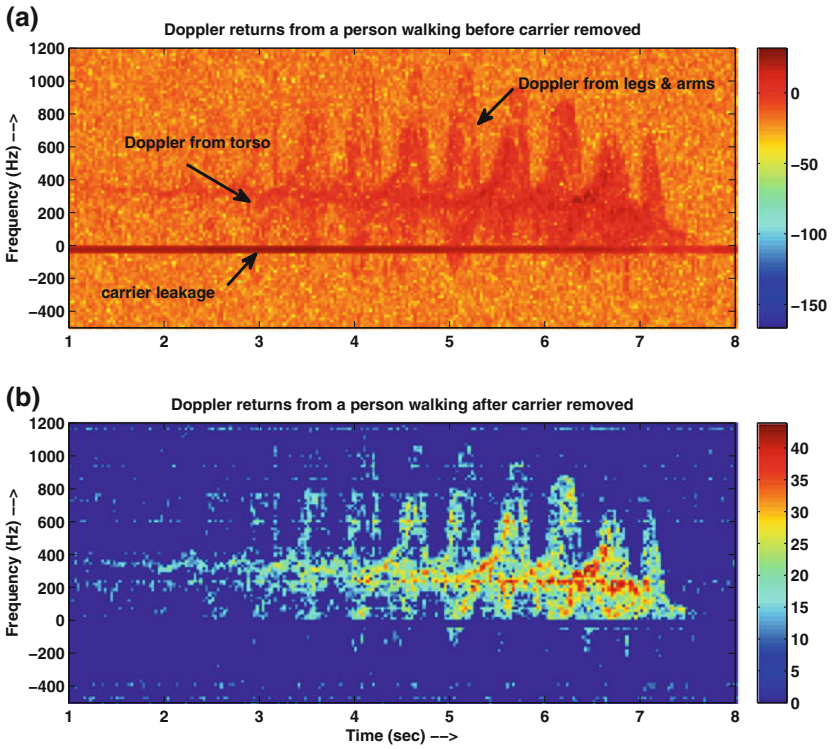


Fig. 11.26 Measured Doppler for a person walking **a** prior to leakage carrier removal and **b** after leakage carrier removed

In the next section, we analyze the Doppler returns from people and horses, and identify the salient features for classification when the returns are strong (high SNR). When we get low SNR, it may not be possible to clearly identify the Doppler associated with limb motion, and, hence, more traditional classifiers, namely, support vector machine and Bayesian classifiers are used to classify Doppler returns from people and animals walking.

11.3.2 Detection and Classification of Ultrasonic Data

In this section, we present several aspects of target detection and classification. The techniques used for human motion classification depend on the quality of the Doppler data or SNR in the demodulated ultrasonic data. The quality of the data depends on how close the target is to the sensor and the prevailing wind turbulence in the vicinity. At close ranges, the amplitude of the received ultrasonic signal is well above the system noise (see Fig. 11.26). Unlike radar, ultrasonic wave speed depends upon the temperature and pressure of the air. Wind manifests itself as pressure fluctuations, and coupled with the temperature fluctuations in the air result in fluctuations of speed of sound. These fluctuations in the speed of sound result in phase changes in addition to those from the motion of the human body limbs. This additional phase competes with the human limb Doppler signal and reduces the Doppler SNR. Reference [38] shows a comparison of indoor and outdoor effects on sonar and radar.

***Case 1*: *High Signal-to-noise Ratio (SNR)*:**

In this case, the targets walk in front of the sensor at a close proximity, and the wind effects on the received signal are minimal. This is the case where some model-based features can be clearly identified and classification can be made based on the model. An example of high SNR is shown in Fig. 11.27.

In order to characterize the target either as a person or animal, we look for (a) cadence, (b) maximum and minimum variation in the Doppler frequency due to limbs and (c) the sequential nature of limb movements. Figure 11.27 shows the enlarged version of a small section of Fig. 11.26. It shows the average Doppler of the torso (average velocity of a person walking)—the middle line—and the Doppler due to arms and legs—the sinusoidal lines on the top and at the bottom showing the contours of the Doppler variation. When the arms and legs swing forward/backward, we get a Doppler above/below the average (the sinusoidal line above/below the average line). The cadence is estimated as $1/t$, where t is the time between the two peaks of a sinusoidal curve. The cadence is estimated to be 1.8 Hz for the person. The maximum and minimum Doppler frequency of limbs with respect to the average is found to be ± 800 Hz, and this will be contrasted with the values for an animal. The vertical lines on Fig. 11.27 are drawn as markers to show the lag (sequential nature of limb movement) between the top and lower sinusoidal curves. The lag is ~ 0.1 s.

Figure 11.28 shows the ultrasonic returns from a horse walking. One clear distinction between the Doppler signatures for a person and a horse walking is that the

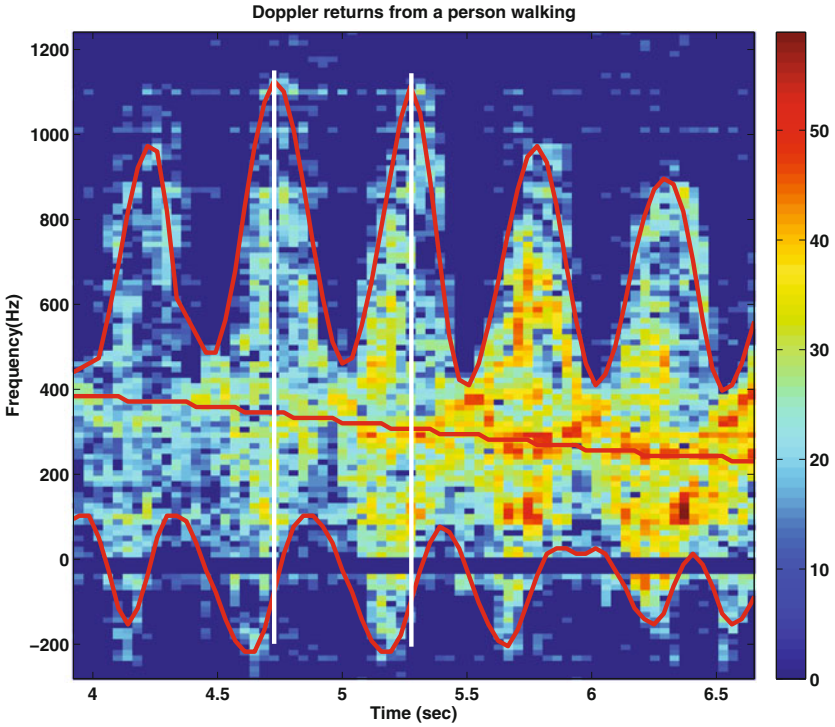


Fig. 11.27 Measured Doppler output for a person walking

signature for a person is sinusoidal in nature. Horse motion is much more complex, as there are lot more moving parts. Another distinct feature for the horse is that the Doppler below the average torso Doppler is significantly less. The maximum variation of Doppler for the horse (∼1500 Hz) is higher compared to that for a person (∼1100 Hz). Figure 11.29 shows the Doppler energy plot for a horse walking. The peaks in Fig. 11.29 show the periodic nature of a horse walking; the cadence can be estimated from it. The cadence of the horse is estimated to be around ∼1.7 Hz. This is also verified using the seismic data. The cadence of the horse is found to be close to the cadence of a person walking in this experiment where the horse is made to walk at a slower pace. Hence, cadence alone cannot be used to distinguish a person from a horse or any other quadruped. From Fig. 11.29, we notice that each peak has an adjacent smaller peak marked by ellipses in the figure. This double peakedness is characteristic of a quadruped walking and is also seen in the model shown in Fig. 11.25b. The time difference between two adjacent peaks is ∼0.12 s. Now we present the algorithm to distinguish people and animals using the features described above. The algorithm is given below:

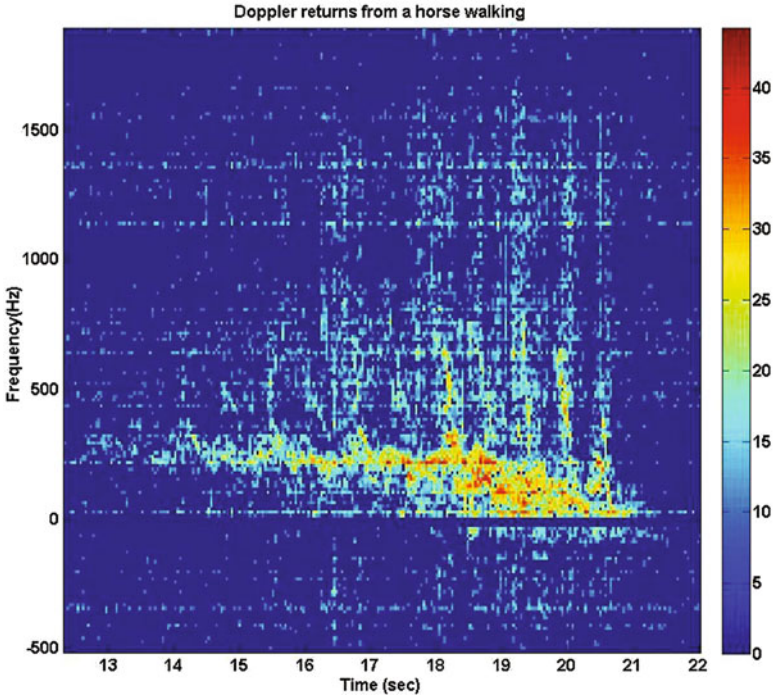


Fig. 11.28 Measured Doppler output for a horse walking

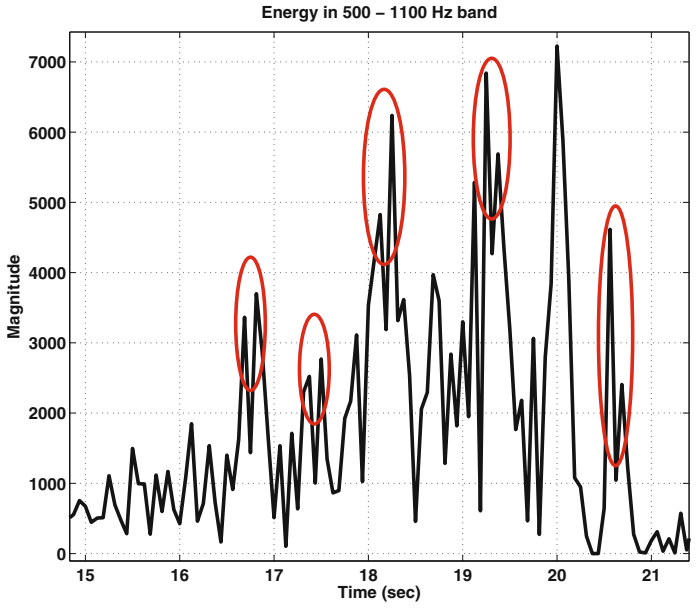


Fig. 11.29 Doppler energy in 500–1100 Hz band for a horse walking

Algorithm 1: Algorithm for classification of targets

Step 1: Determine the average Doppler corresponding to the torso motion.
Step 2: If the Doppler variation above and below the average is approximately same, estimate the periodicity of the Doppler variations on both sides. If they are roughly the same and the separation between the adjacent positive (Doppler above the torso) and negative (Doppler below the torso, see Fig. 11.27) peaks is ∼0.1 s, declare the presence of a person.
Step 3: If the Doppler variation above the torso is much higher than the Doppler below the average, estimate the energy in the signal above the average. If there are double peaks, then declare the presence of a quadruped. Figure 11.30 shows the flowchart used for the algorithm.

If the SNR is high, Algorithm 1 is the preferred method for detection and classification of people and animals. If the SNR is low, standard signal processing techniques are used since model-based features are hard to extract.

***Case 2*: *Low Signal-to-noise Ratio*:**

In the event the signal returns are weak for any reason—such as prevailing winds, the target being farther than optimal distance, the target being illuminated by the ultrasonic transducer at an angle, etc.—the features observed in the case of high SNR may not be present. For low SNR data, it is appropriate to use classical signal processing techniques to classify the targets. In [101], context length differential cepstral features generated from the STFT of the received micro-Doppler signal were

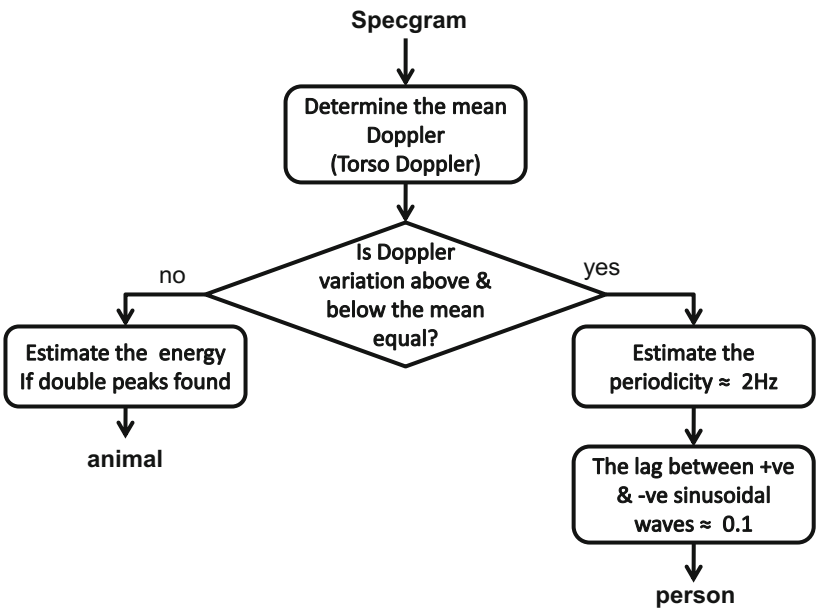


Fig. 11.30 Flowchart for the algorithm

used for classification. We present two algorithms for the classification of signals with low SNR, namely, (a) Bayesian classifier and (b) SVM. We also use two different feature sets: (a) traditional cepstral coefficients [56, 101] and (b) aggregated energy in the frequency bands.

We now describe the feature selection processes used for the classifiers.

***Aggregated energy in frequency bands*:** Clearly, from Figs. 11.26 and 11.28 the Doppler returns from animals are quite different compared to those from humans and, hence, so are the energy levels in different frequency bands. Let $S(t)$ denote the STFT of the Doppler returns from a target. In order to classify the target, the Doppler range spanning from -500 to 1500 Hz is divided into $N \approx 30$ bands and the total energy in each band is considered as a feature. Quite often, 2–5 time slots are considered together to compute the features in each band. Then each feature F_i is selected as

$$F_i = \sum_{k=0}^{2} S(t+k, f_i), \quad \forall i \in \{1, 2, \ldots, N\},$$

where $S(t, f_i)$ is the energy in f_ith band at time t and N is the number of features. Note that each F_i is the sum of energies in three consecutive time slots to capture the time dependence of the Doppler and $X = \{F_1, \ldots, F_N\}$ is the normalized feature vector.

***Cepstral features*:** Cepstral coefficients are used in the literature for classification of ultrasonic Doppler returns from targets [50, 56, 101]. Most notably, mel frequency cepstral coefficients are used for speaker identification and audio signal classification. Here we use cepstral coefficients for classification the ultrasonic micro-Doppler signature from targets. The cepstrum of a signal is defined as the inverse Fourier transform of the log magnitude spectrum. For target classification here we used real cepstral coefficients (in Matlab [64]—"rceps") defined as

$$C = real\,(\text{ifft}\ \ (\log\ \ (\text{abs}\,(\text{fft}\,(s(t))))))\,,$$

where $s(t)$ is the base-band Doppler signal. The feature set $X = \{c_1, c_2, \ldots, c_N\}$, where $c_i \in C, \forall i$, is used for classification of targets.

***Training and Testing*:** In order to verify the algorithm against the real data, several sets of data are collected consisting of (a) one man walking, (b) one woman walking, and (c) one horse walking. A SVM with a Gaussian kernel and Bayesian MVG classifiers were used to classify the targets as (a) human, (b) animal and (c) noise.

In Figs. 11.31 and 11.32, we present results from SVM and MVG classifiers for the case when a man and a horse were walking. We used the aggregated energy in frequency bands as the feature set in Fig. 11.31. Figure 11.32 shows the results using cepstral features. In both the figures, the spectrogram of the Doppler returns from both the targets are presented on the top. The SVM results are given next, followed by the MVG classifier results. From these figures, it appears that the cepstral features

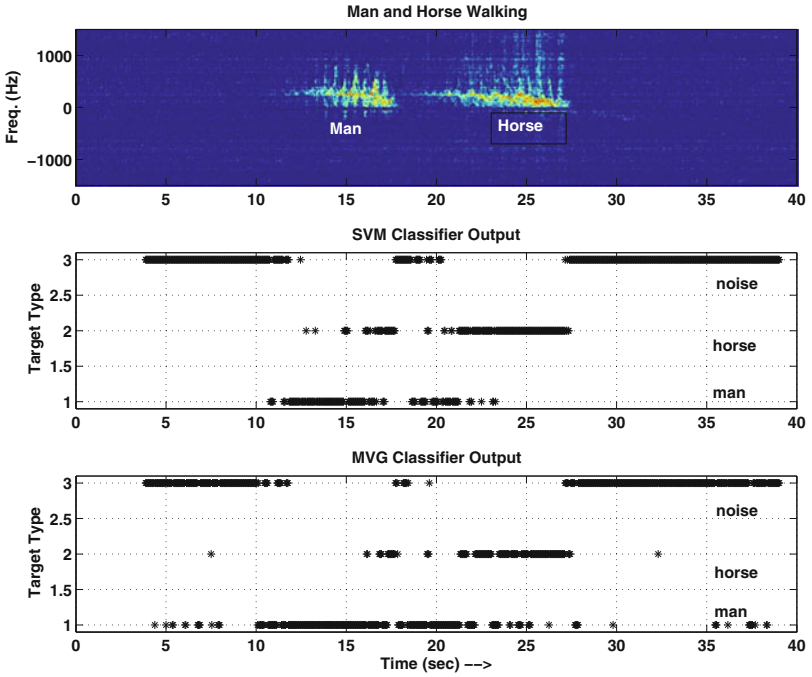


Fig. 11.31 Classification results using aggregated spectral coefficients: **a** Doppler returns of multiple targets walking, **b** classification by SVM, and **c** classification by MVG classifier

resulted in better classification. From Fig. 11.32, both the classifiers, namely, SVM and MVG classifiers, performed well when cepstral features are used.

11.3.2.1 Counting number of targets

For situational awareness, sometimes it is necessary to count the number of targets. It is possible to count the number of targets using ultrasonic data. Towards this goal, we processed the ultrasonic data to count the number of targets in the vicinity using the energy content in various bands of the Doppler. Figure 11.33 shows the flowchart for the algorithm used in counting the number of targets. For processing the ultrasonic data, a 1 s interval of the data is considered at a time and the algorithm shown in Fig. 11.33 is used to find the energy in each band. Then a sliding window is used, which slides approximately 0.1 s and next segment of data is obtained and processed. The algorithm results for several runs are shown in Fig. 11.34. The scenarios used corresponds to (a) one man walking, (b) one man leading an animal, (c) two men and one woman walking and, (d) four men and three women walking.

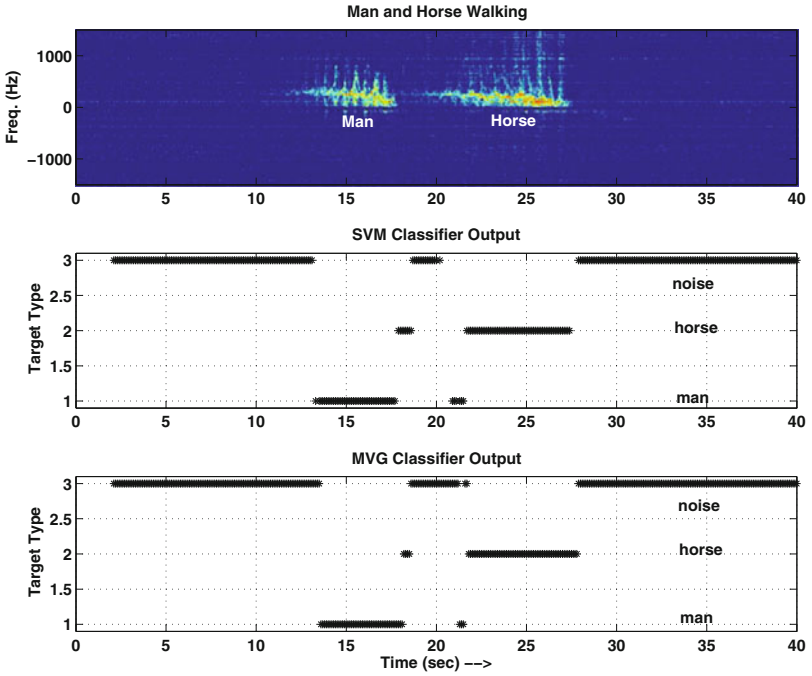


Fig. 11.32 Classification results using cepstral coefficients: **a** Doppler returns of multiple targets walking, **b** classification by SVM, and **c** classification by MVG classifier

Fig. 11.33 Flowchart showing the ultrasonic signal processing for counting number of targets

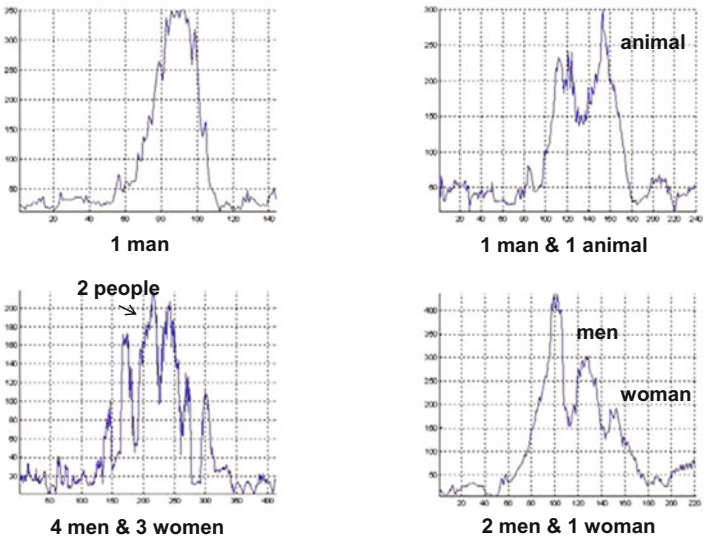


Fig. 11.34 Target count using ultrasonic data analysis

In the last case, a count of only six targets is realized using the algorithm. The reason is due to a large number of people, one is very close to the other, masking the Doppler returns from the others.

In this chapter, we presented the analysis used for classification of vehicles and presented the techniques used in detection of people. These are some of the requirements needed to produce good situational awareness.

Chapter 12
Sensor Data Fusion

In the chapter on personnel detection, we talked about using multimodal and multi-sensor systems in order to attain a high probability of detection with fewer false alarms and high confidence in the decision process. When multiple sensors of same category are used for detection of a target, we need to somehow combine the results of all the sensors to obtain the overall probability of detection. For example, if majority of sensors declare the presence of a target, it is most likely that the target is present. The confidence in a decision made by a majority of sensors is high compared to the decision made by a single sensor. The process of combining different pieces of information generated by various sensors in order to obtain a aggregate decision is called "*fusion*". The word "information" may pertain to the data or a decision that is an outcome of some process or human input, etc. Fusion can be done at different levels as shown in Fig. 12.1. Fusion at various levels is done in order to reduce the data that need to be transmitted at each level to the next level. For example, in the scenario shown in Fig. 12.1, there are several sensors distributed in a field to observe a phenomenon. Each sensor records the data. The data are clustered or analyzed to detect an acoustic event (assuming acoustic sensor). Then only the data corresponding to the acoustic event need to be sent to the next level instead of all the data. Note that decisions can be made at all levels:

$$u = f(X), \quad X = (X_1, \ldots, X_i, \ldots, X_m) \tag{12.1}$$

where f is the decision function and X_i is the data/feature from the ith sensor. Now, let us look at the length $|X|$ of the vector X. If each vector X_i corresponds to the data collected for a period of 1 sec at a sampling rate of f_s, then $|X_i| = f_s$. Then $|X| = mf_s$. Typically f_s could be anywhere between 1000 and 40,000. Hence, in a majority of the cases, data level fusion is not a viable solution, because, (a) a large amount of data needs to be transmitted to the fusion center, which, in turn, requires high bandwidth communication channels; (b) most of the data simply represents noise, but not the actual signal pertaining to the phenomenon; and (c) computational complexity of the function $f(X)$ would be high unnecessarily. Now, if we extract, say, 40 features from each sensor, then $|X_i| = 40$ and the $|X| = 40$ m. For the case,

T. Damarla, *Battlefield Acoustics*, DOI 10.1007/978-3-319-16036-8_12

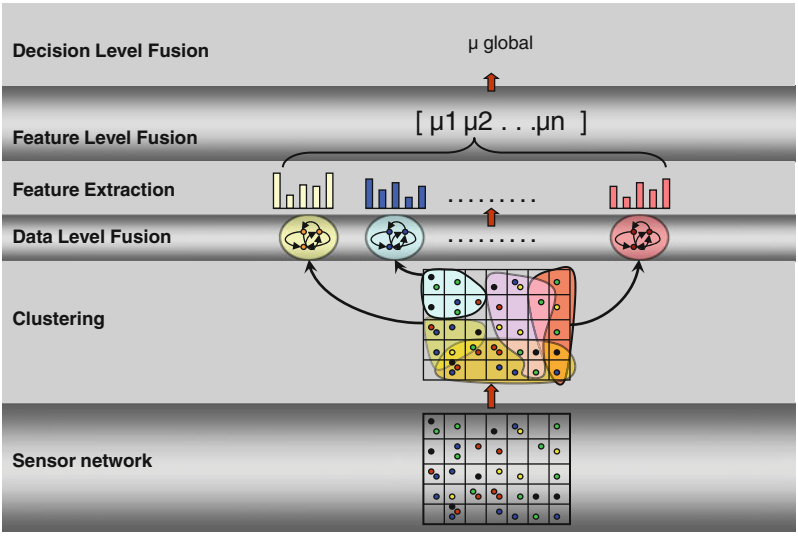


Fig. 12.1 Multimodal information fusion at different levels of hierarchy

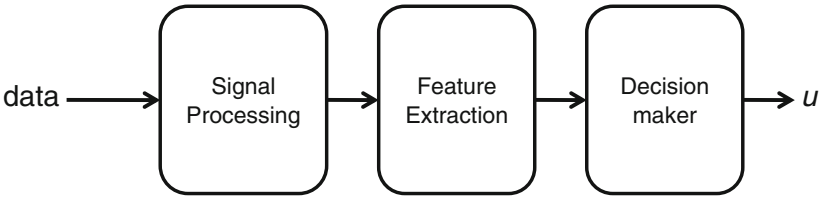


Fig. 12.2 Sensor level processing of data

$f_s = 1000$ and $m = 6$, $|X|$ changes from 6000 to 240, a considerable reduction in the bandwidth requirements.

Quite often, each sensor processes the data locally and makes a decision as shown in Fig. 12.2. The signal processing may involve filtering in order reduce the noise and then taking FFT or some other techniques appropriate to bring out the salient features of the phenomenon. The features pertaining to the phenomenon are then extracted and sent to the decision maker. The decision maker outputs a decision u. For a classification problem, the output u can take several forms:

1. $u = i$, where $i \in \{1, 2, \ldots, R\}$, R is the number of classes,
2. u_i could be the probability that the vector X_i belongs to the class i,
3. u_i could be the belief that the vector X_i belongs to the class i.

Each sensor 'j' transmits u_j to the fusion center where all of them will be fused to find the final probability of detection or belief that the observed phenomenon is due to class i. We now present several fusion algorithms.

12.1 Dempster-Shafer Rule of Fusion

The theory of evidence has its roots in the work of Dempster [27–29] and later Shafer [84] built the theory presented by Dempster. The theory of evidence, now called the Dempster-Shafer theory, has found its niche in artificial intelligence [99] and in many other applications where the absoluteness of probabilities do not make real sense of the observed phenomena. In those cases, a range of the likelihood of the phenomena is more appropriate. In Dempster-Shafer theory, there are some concepts we should understand.

Belief

Consider an example of the age range of members of various clubs listed below. In the case of Club #1, the range implies that the distribution of age is between 22 and 27. Suppose now we make a query Q with a range $Q = [27\ \ 30]$ then the following are true

(a) Age(Club#i) $\in Q$ is *possible* if $D_i \bigcap Q \neq \emptyset$ where D_i is the range of Club#i and $\emptyset$ is the empty set.
(b) Q is *certain* if (or *necessary*) $D_i \subseteq Q$.
(c) Q is *not possible* if $D_i \bigcap Q = \emptyset$.

In the third column of Table 12.1, we presented the output of the query Q. Now let us ask, what fraction of the clubs have the age group between age range $Q = [27\ \ 30]$. The answer contains two parts, one relating to the certainty of Q and the other to its possibility.

We now formalize the problem of evidence theory by considering an universe of mutually exhaustive hypothesis H. The "frame of discernment" (or "Power set 2^H") of H is the set of all possible subsets of H. For example, if $H = \{Q_1, Q_2, Q_3\}$, then the frame of discernment of H is

$$(\emptyset, Q_1, Q_2, Q_3, \{Q_1, Q_2\}, \{Q_1, Q_3\}, \{Q_2, Q_3\}, \{Q_1, Q_2, Q_3\})$$

We define a basic probability assignment as a function 'm' that maps subset of H to [0, 1] such that $m(Q)$ is between 0 and 1 for all subsets Q of H, and

Table 12.1 Age groups of various clubs and the response to a query

Club	Age range	Test
1	[22 27]	Poss
2	[19 22]	¬ poss
3	[29 34]	Poss
4	[19 22]	¬ poss
5	[27 30]	Cert

$$\sum_{Q \subseteq H} m(Q) = 1. \tag{12.2}$$

$$m(Q) = 0, \quad \text{if} \quad Q = \emptyset. \tag{12.3}$$

The properties of the function 'm' are

- $m(H) = 1$,
- $m(A) \leq m(B)$ if $A \subset B$, or
- that there is a relationship between $m(A)$ and $m(\neg A)$, where $\neg A$ is the complement of A.

We define another function called the *belief function* from subsets of H onto [0, 1] such that

$$bel\,(A) = \sum_{Q \subseteq A;\ Q \neq \emptyset} m(Q). \tag{12.4}$$

The counter part of *bel* is the plausibility measure *pl* with

$$pl\,(A) = \sum_{Q \cap A \neq \emptyset} m(Q). \tag{12.5}$$

It is important to note that

$$\begin{aligned} bel\,(A) + bel\,(\neg A) &\leq 1 \\ pl\,(A) + pl\,(\neg A) &\geq 1 \end{aligned} \tag{12.6}$$

The above equation means that it is possible to believe that something is could be both true and false to some degree. The above statement is the fundamental difference between the theory of probability and the Dempster-Shafer evidential theory.

Example 1: Consider the case where three people $\{X, Y, Z\}$ go into a room one by one and leave. It is found that there is something missing after their visit. A inspector after visiting the room assigned the following mass function values as shown in Table 12.2: The belief and plausibility in an element A of power set (2^H) are computed using the (12.4) and (12.5) and are given by the entries in Table 12.3, which are computed as follows:

$$bel(A = \{Y, Z\}) = m(\{Y\}) + m(\{Z\}) + m\,(\{Y, Z\}) = 0.1 + 0.3 + 0.15 = 0.55$$

and

$$\begin{aligned} pl(A = \{Y, Z\}) &= m(\{Y\}) + m(\{Z\}) + m\,(\{Y, Z\}) + m\,(\{X, Y\}) \\ &\quad + m\,(\{X, Z\}) + m\,(\{X, Y, Z\}) \\ &= 0.1 + 0.3 + 0.15 + 0.1 + 0.15 + 0.1 = 0.9 \end{aligned}$$

Table 12.2 Assignment of mass function values

Event	Mass m
No one is guilty	0
X is guilty	0.10
Y is guilty	0.10
Z is guilty	0.30
Either X or Y is guilty	0.10
Either X or Z is guilty	0.15
Either Y or Z is guilty	0.15
Either X or Y or Z is guilty	0.10

Table 12.3 Belief and plausibility values for A

A	$\{X\}$	$\{Y\}$	$\{Z\}$	$\{X, Y\}$	$\{X, Z\}$	$\{Y, Z\}$	$\{X, Y, Z\}$
$m(A)$	0.10	0.10	0.3	0.1	0.15	0.15	0.1
$bel(A)$	0.10	0.10	0.3	0.3	0.55	0.55	1.0
$pl(A)$	0.45	0.45	0.7	0.7	0.90	0.90	1.0

Table 12.4 Mass functions of A assigned by two different investigators

A	$\{X\}$	$\{Y\}$	$\{Z\}$	$\{X, Y\}$	$\{X, Z\}$	$\{Y, Z\}$	$\{X, Y, Z\}$
$m_1(A)$	0.10	0.10	0.3	0.1	0.15	0.15	0.1
$m_2(A)$	0.10	0.20	0.2	0.1	0.10	0.20	0.1
$[m_1 \oplus m_2](A)$	0.1201	0.2168	0.4495	0.0448	0.0591	0.0955	0.0142

Now, suppose there are two investigators who visited the room and each assigned the mass function values m_1 and m_2 for each hypothesis as shown in Table 12.4. Now, we would like to fuse the information provided by the two investigators in order to obtain one single joint value for each hypothesis. For this, we use Dempster-Shafer rule of combination

$$[m_1 \oplus m_2](A) = \frac{\sum_{B \cap C = A,\ A \neq \emptyset} m_1(B) \star m_2(C)}{1 - \sum_{B \cap C = \emptyset} m_1(B) \star m_2(C)} \tag{12.7}$$

For example, in order to combine $m_1(\{X, Y\})$ and $m_2(\{X, Y\})$, from (12.7), we get

$$[m_1 \oplus m_2](\{X, Y\}) = \frac{m_1(\{X,Y\}) * m_2(\{X,Y,Z\}) + m_2(\{X,Y\}) * m_1(\{X,Y,Z\})}{1 - (m_1(\{X,Y\}) * m_2(\{Z\}) + m_2(\{X,Y\}) * m_1(\{Z\}))}$$

After computing the value of $[m_1 \oplus m_2]$ for each hypothesis $Q = \{X, Y, \ldots\}$, one would make sure that (12.2) is satisfied.

The output of the fusion operation (the rule of combination) is shown in Table 12.4 for those values of m_1 and m_2 given in the table. Now, let us estimate the belief in $Q = (\{X, Y\})$ using m_1, m_2 and $[m_1 \oplus m_2]$ individually to see the impact of fusing m_1 and m_2.

Using m_1:

$$bel\,(\{X, Y\}) = m_1\,(\{X, Y\}) + m_1\,(\{X\}) + m_1\,(\{Y\}) = 0.1 + 0.1 + 0.1 = 0.3$$

Using m_2:

$$bel\,(\{X, Y\}) = m_2\,(\{X, Y\}) + m_2\,(\{X\}) + m_2\,(\{Y\}) = 0.1 + 0.2 + 0.1 = 0.4$$

Using $m_3 = [m_1 \oplus m_2]$:

$$\begin{aligned} bel\,(\{X, Y\}) &= m_3\,(\{X, Y\}) + m_3\,(\{X\}) + m_3\,(\{Y\}) \\ &= 0.0448 + 0.1201 + 0.2168 = 0.3817 \end{aligned}$$

We notice that the belief in $\{X, Y\}$ has improved overall when we use the combined (fused $[m_1 \oplus m_2]$) mass function. We also learn how to generate the $[m_1 \oplus m_2]$ fused mass functions from individual mass functions. In the fusion applications, m_1 and m_2 could be the estimates generated using acoustic and seismic sensors for an application such as the "personnel detection" problem we considered in the Chap. 11. Figure 12.3

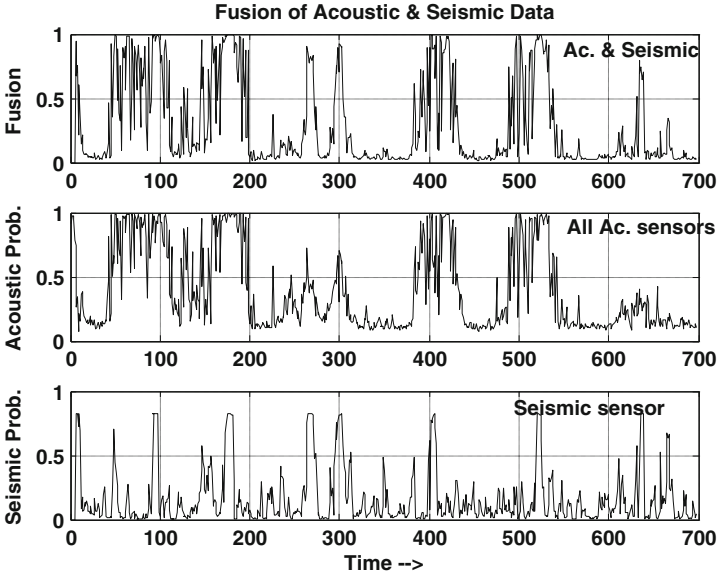


Fig. 12.3 Fusion of seismic and acoustic detection probabilities

shows the individual probabilities of detection of people using seismic and acoustic sensor and detection after fusing the two probabilities using the Dempster-Shafer fusion rule.

12.2 Fusion of Heterogeneous Sensor Data Using Bayesian Approach

In this section, we present the fusion algorithm used to fuse the data from PIR, seismic and ultrasonic detectors. The same algorithm can be used to fuse the output of multiple UGSs also. The classification of the multimodal sensor data is done at each UGS system and the fusion is done at each UGS. It is assumed that each sensor receives an input

$$y_i = s_i + n_i,$$

where s_i is the signal and n_i is the noise. Each sensor makes a decision and generates a binary output

$$u_i = d(y_i) = \begin{cases} 1, & \text{if } \mathrm{H}_1 \text{ is true} \\ -1, & \text{if } \mathrm{H}_0 \text{ is true} \end{cases} \tag{12.8}$$

where H_1 and H_0 are the hypotheses for a target being present or absent, respectively, and "$d()$" is a decision function. The fusion architecture used to fuse the sensor data at each UGS is shown in Fig. 12.4. Each sensor makes a decision and passes its output to the fusion box, which makes the final decision as to the presence of a target or not. Chair and Varshney [13] presented an optimal decision rule for the likelihood ratio test (LRT)

$$\frac{P(u_1, \ldots, u_n|H_1)}{P(u_1, \ldots, u_n|H_0)} \underset{H_0}{\overset{H_1}{\gtrless}} \frac{P_0\ (C_{10} - C_{00})}{P_1\ (C_{01} - C_{11})} \tag{12.9}$$

where P_0 and P_1 are the priors for the two hypotheses and C_{ij} is the cost for selecting H_i when H_j is true for $i, j \in \{0, 1\}$. If we assume the minimum probability of error criterion, then $C_{00} = C_{11} = 0$ and $C_{01} = C_{10} = 1$. Then (12.9) becomes

$$\frac{P\ (\mathbf{u}|H_1)}{P\ (\mathbf{u}|H_0)} \underset{H_0}{\overset{H_1}{\gtrless}} \frac{P_0}{P_1}. \tag{12.10}$$

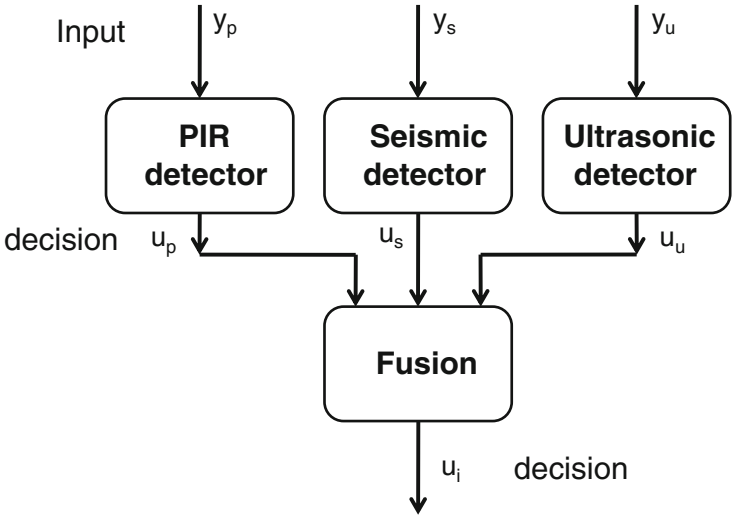


Fig. 12.4 Fusion structure

Using Bayes rule and after simplifying, we get

$$\frac{P\left(H_1|\mathbf{u}\right)}{P\left(H_0|\mathbf{u}\right)} \underset{H_0}{\overset{H_1}{\gtrless}} 1. \tag{12.11}$$

Then the log-likelihood ratio test (LLRT) is

$$\log \frac{P\left(H_1|\mathbf{u}\right)}{P\left(H_0|\mathbf{u}\right)} \underset{H_0}{\overset{H_1}{\gtrless}} 0. \tag{12.12}$$

Let us denote the probability of false alarm P_{F_i} and the probability of mis-detection P_{M_i} for the ith sensor. Then the left-hand side of the (12.12) can be expressed in terms of P_{F_i} and P_{M_i} and it is stated in the following theorem:

Theorem: Given n detectors and their decisions $\mathbf{u} = \{u_1, u_2, \ldots, u_n\}$, the log-likelihood ratio of two hypothesis can be expressed as

$$\log \frac{P\left(H_1|\mathbf{u}\right)}{P\left(H_0|\mathbf{u}\right)} = \log \frac{P_1}{P_0} + \sum_{S_+} \log \frac{1 - P_{M_i}}{P_{F_i}} + \sum_{S_-} \log \frac{P_{M_i}}{1 - P_{F_i}} \tag{12.13}$$

where S_+ is the set of all $u_i = +1$ and S_- is the set of all $u_i = -1$.

Proof: We have

$$P(H_1|\mathbf{u}) = \frac{P(H_1,\mathbf{u})}{P(\mathbf{u})} = \frac{P_1}{P(\mathbf{u})} \prod_{S_+} P(u_i = +1|H_1) \cdot \prod_{S_-} P(u_i = -1|H_1)$$
$$= \frac{P_1}{P(\mathbf{u})} \prod_{S_+} (1 - P_{M_i}) \cdot \prod_{S_-} P_{M_i} \tag{12.14}$$

Similarly

$$P(H_0|\mathbf{u}) = \frac{P_0}{P(\mathbf{u})} \prod_{S_-} (1 - P_{F_i}) \cdot \prod_{S_+} P_{F_i} \tag{12.15}$$

Thus

$$\log \frac{P(H_1|\mathbf{u})}{P(H_0|\mathbf{u})} = \log \frac{P_1}{P_0} + \sum_{S_+} \log \frac{1 - P_{M_i}}{P_{F_i}} + \sum_{S_-} \log \frac{P_{M_i}}{1 - P_{F_i}} \tag{12.16}$$

■

Hence the data fusion rule can be expressed as

$$d(u_1, \ldots, u_n) = \begin{cases} 1, & \text{if } a_0 + \sum_{i=1}^{n} a_i u_i > 0 \\ -1, & \text{otherwise} \end{cases} \tag{12.17}$$

where the optimum weights are given by

$$a_0 = \log \frac{P_1}{P_0}; \quad a_i = \log \frac{1 - P_{M_i}}{P_{F_i}}, \text{ if } u_i = 1;$$

$$a_i = \log \frac{1 - P_{F_i}}{P_{M_i}}, \text{ if } u_i = -1$$

where P_{F_i} and P_{M_i} are the probability of false alarm and the probability of miss for the ith sensor. The optimal data fusion rule can be implemented as shown in Fig. 12.5.

12.2.1 Decision Fusion to Achieve Specified False Alarm Rate

The approach presented in the previous section requires the knowledge of the a-priori probabilities P_0 and P_1, which are hard to determine. An optimal fusion decision scheme using a Neyman-Pearson (N-P) test [91] mitigates the requirement on priors. The LRT can be rewritten as

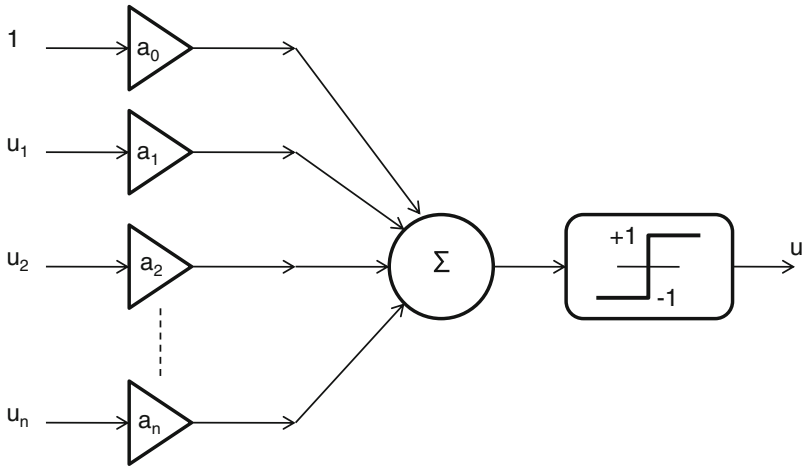


Fig. 12.5 Optimum fusion rule implementation

$$\Lambda(u) = \frac{P(u_1, \ldots, u_N|H_1)}{P(u_1, \ldots, u_N|H_0)} \underset{H_0}{\overset{H_1}{\gtrless}} t, \tag{12.18}$$

where "t" is the threshold determined by the desirable probability of false alarm P_F^f at the fusion center, that is,

$$\sum_{\Lambda(u)|>t^*} P\left(\Lambda(u)|H_0\right) = P_F^f. \tag{12.19}$$

With the assumption that the decisions from the sensors are independent, we get

$$\Lambda(u) = \prod_{i=1}^{N} \frac{P(u_i|H_1)}{P(u_i|H_0)} \underset{H_0}{\overset{H_1}{\gtrless}} t \tag{12.20}$$

In order to compute the N-P test we need to compute $P\left(\Lambda(u)|H_0\right)$. Due to independent assumption, it is easy to compute $P\left(\log \Lambda(u)|H_0\right)$, which can be expressed as the convolution of individual $P\left(\log \Lambda(u_i)|H_0\right)$. From (12.20), we get

$$P\left(\log \Lambda(u)|H_0\right) = P\left(\log \Lambda(u_1)|H_0\right) * \cdots * P\left(\log \Lambda(u_N)|H_0\right) \tag{12.21}$$

From the definition of false alarm, the likelihood ration $\Lambda(u)$ assumes the value $\frac{\left(1-P_{D_i}\right)}{\left(1-P_{F_i}\right)}$ when $u_i = 0$ with probability $\left(1 - P_{F_i}\right)$ under hypothesis H_0 and with probability $\left(1 - P_{D_i}\right)$ under hypothesis H_1. Similarly, it assumes the value $\frac{P_{D_i}}{P_{F_i}}$ when

$u_i = 1$ with probability P_{F_i} under hypothesis H_0 and with probability P_{D_i} under hypothesis H_1. Hence, (12.21) can be written as

$$P\left(\log \Lambda(u)|H_0\right) = \left(1 - P_{F_i}\right)\delta\left(\log \Lambda(u) - \log \frac{1-P_{D_i}}{1-P_{F_i}}\right) + P_{F_i}\,\delta\left(\log \Lambda(u) - \log \frac{P_{D_i}}{P_{F_i}}\right) \tag{12.22}$$

and

$$P\left(\log \Lambda(u)|H_1\right) = \left(1 - P_{D_i}\right)\delta\left(\log \Lambda(u) - \log \frac{1-P_{D_i}}{1-P_{F_i}}\right) + P_{D_i}\,\delta\left(\log \Lambda(u) - \log \frac{P_{D_i}}{P_{F_i}}\right) \tag{12.23}$$

where

$$\delta(q) = \begin{cases} 1, & \text{if } q = 0, \\ 0, & \text{if } q \neq 0. \end{cases}$$

The probability of false alarm P_F^f and probability of detection P_D^f at the fusion center are given by

$$\sum_{\Lambda(u) > t^*} P(\Lambda(u)|H_0) = P_F^f, \tag{12.24}$$

$$\sum_{\Lambda(u) > t^*} P(\Lambda(u)|H_1) = P_D^f, \tag{12.25}$$

where t^* is the threshold chosen to satisfy Eq. (12.24) for a desired P_F^f. The goal is to achieve with the N-P test, a pair (P_F^f, P_D^f) such that

$$P_F^f \leq \min_{i \in N}\left\{P_{F_i}\right\} \quad \text{and} \quad P_D^f > \min_{i \in N}\left\{P_{D_i}\right\} \tag{12.26}$$

where $\left(P_{D_i}, P_{F_i}\right)$ is the N-P test level for sensor i, $i = 1, \ldots, N$. It is shown in [91] that, if $P_{D_i} = P_{D_j} = P_D$ and $P_{F_i} = P_{F_j} = P_F$ for all $i \neq j$, $i, j \in \{1, \ldots, N\}$, then the LR test would lead to

$$\sum_{i=1}^{N} a_i u_i \underset{H_0}{\overset{H_1}{\gtrless}} t \tag{12.27}$$

where

$$a_i = \begin{cases} \log\left(\frac{P_D}{P_F}\right), & u_i = +1,\ i = 1, \ldots, N \\ \log\left(\frac{1-P_F}{1-P_D}\right), & u_i = -1,\ i = 1, \ldots, N. \end{cases} \tag{12.28}$$

If k out of N sensors result in $u_i = +1$, that is, hypothesis H_1, the (12.27) can be written as

$$k \log\left(\frac{P_D\,(1-P_F)}{P_F\,(1-P_D)}\right) \underset{H_0}{\overset{H_1}{\gtrless}} t + N \log\left(\frac{1-P_F}{1-P_D}\right). \tag{12.29}$$

Using the fact that $P_F < P_D$, we have

$$\log\left(\frac{P_D\,(1-P_F)}{P_F\,(1-P_D)}\right) > 0$$

and hence the N-P test becomes

$$k \underset{H_0}{\overset{H_1}{\gtrless}} t^* \tag{12.30}$$

where t^* is some threshold determined based on the allowed false alarm rate P_F at the fusion center. The variable k has the binomial distribution with parameters N and P_F under H_0 and N and P_D under H_1 and is given by:

$$P_F^f = \sum_{k=\left\lceil t_f^* \right\rceil}^{N} \binom{N}{k} P_F^k\,(1-P_F)^{N-k} \tag{12.31}$$

$$P_D^f = \sum_{k=\left\lceil t_f^* \right\rceil}^{N} \binom{N}{k} P_D^k\,(1-P_D)^{N-k} \tag{12.32}$$

where $\left\lceil t_f^* \right\rceil$ indicates the smallest integer (number of sensors) required to exceed the threshold t^* to achieve the desired P_F^f. If the sensors have different probabilities of detection and false alarms, that is, $P_{D_i} \neq P_{D_j}$ and $P_{F_i} \neq P_{F_j}$, one can use the techniques described in [91] to estimate the final probabilities and the algorithm is given below:

Algorithm 1: **Estimation of final probabilities for sensors with different receiver operating curves (ROCs)**:

Let the set $S = \{1, 2, \ldots, N\}$ represent all the sensors where N is the number of sensors, and $|t_f^*|$ is determined based on the final P_F desired. Then, let us denote all the subsets of S with k elements by S_k, where $|S_k| = \binom{N}{k}$, then the final probability of false alarms and the final probability of detection are given by

$$P_F^f = \sum_{k=|t_f^*|}^{N} \left(\sum_{j=1}^{|S_k|} \left(\prod_{q=1}^{|s_j|; s_j \in S_k} P_{F_q} \prod_{r=1}^{|\bar{s}_j|} (1 - P_{F_r}) \right) \right) \tag{12.33}$$

$$P_D^f = \sum_{k=|t_f^*|}^{N} \left(\sum_{j=1}^{|S_k|} \left(\prod_{q=1}^{|s_j|; s_j \in S_k} P_{D_q} \prod_{r=1}^{|\bar{s}_j|} (1 - P_{D_r}) \right) \right) \tag{12.34}$$

where $\bar{s}$ is the complement of s.

12.2.1.1 Data Collection and Algorithm Results

The fusion algorithms are tested on some data collected at various locations and on several terrains by enacting several scenarios. Some of the scenarios are (a) one person, two people and three people walking, (b) one animal, two animals and three animals walking, (c) one animal and one person walking, and several people and several animals walking, etc. A total of 180 scenarios were enacted. One of the trails used for the data collection is shown in Figs. 12.6 and 12.7 shows the path. The data are collected using three UGSs, each system has a (a) PIR sensor, (b) seismic sensor and (c) ultrasonic sensor. The separation between two UGS is about 60–75 m.

Fig. 12.6 Typical trail

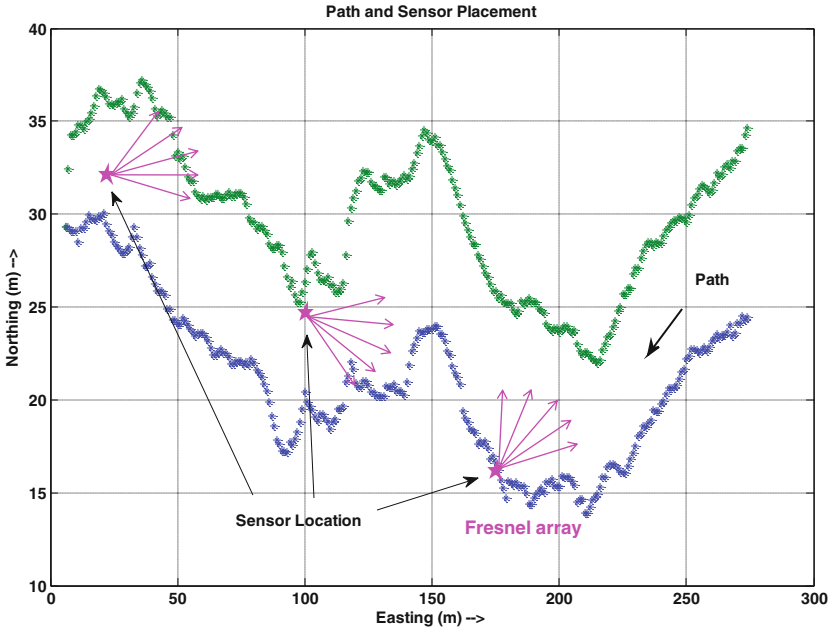


Fig. 12.7 Typical path used to collect data

In order to estimate the overall probability of detection at each UGS, we use the following algorithm:

Algorithm 2: **Algorithm for computing probability of detection at a UGS**

- Compute the probability of detection P_{D_i} for each sensor modality, namely, PIR, seismic and ultrasonic, and their probabilities of false alarm P_{F_i}.
- The P_{D_i} and P_{F_i} are computed as follows:
  - Generate a training set for a sensor
  - Train the PIR, seismic and ultrasonic detectors using their respective training sets
  - Use individual detectors to test the entire data set collected on all days
  - Estimate the probability of false alarm based on the number of non-targets identified as targets and the probability of detection based on the number of targets correctly identified as targets.
- Perform the detection of new data using the detectors to obtain u_i, for all i.
- Estimate the probability of detection $P_F^{U_j}$ and $P_D^{U_j}$ for all UGS U_j, using (12.33) and (12.34), respectively.

Table 12.5 shows the P_{D_i} and P_{F_i} for the three sensors, and the overall probability of detection and false alarms for various $\left|t_f^*\right|$ values are given in Table 12.6. From

Table 12.5 Prob. of false alarm and detection for various sensors

PIR sensor	Seismic sensor	Ultrasonic sensor
$P_F = 0.1$	$P_F = 0.1$	$P_F = 0.07$
$P_D = 0.9$	$P_D = 0.9$	$P_D = 0.8$

Table 12.6 Threshold versus final prob. of false alarm and detection at each UGS U_j

$\lvert t_f^* \rvert$	P_F^f	P_D^f
1.0000	0.2467	0.9980
2.0000	0.0226	0.9540
3.0000	0.0007	0.6480

Table 12.6, we notice that an improved overall probability of false alarm is achieved for $\left|t_f^*\right| \geq 2$, which implies that two or more sensors should have the correct detection and classification.

Algorithm 3: **Algorithm for computing the probability of a correct classification at a UGS**

- First detect the presence of a target (animal/person) using each sensor modality
- If a target is detected, determine whether it is an animal or a person. Determine the probability of person and the probability of animal at each modality
- Use (12.33) and (12.34) to determine the overall P_F and P_D at each UGS. Note that the probability of false alarm here is the probability of an animal being classified as a person.
- One may use 'OR' or 'AND' rules [91] to fuse the information from different modalities instead of the (12.33) and (12.34).

It should be noted that the above fusion techniques assume that the decisions made by the sensors are independent. However, all the sensors are observing the same phenomenon, hence the data collected and their decisions are correlated. In the next section, we present the techniques useful for fusing correlated decisions.

12.2.2 Fusion of Correlated Decisions

In order to fuse correlated data from two different sensors, one needs to know the joint probability density function (*pdf*) of the two sensors. One can generate the joint *pdf* for the sensors by collecting data for a long time. Often, this approach is impractical. However, if the individual *pdfs* of the sensors are known, then copula functions can be used. The definition of copula is as follows:

Copula: An N-dimensional copula is an N-variate CDF on $[0; 1]^N$ whose univariate marginals are uniformly distributed on [0; 1].

From the definition of copula, it is clear that the copulas couple multivariate joint distribution functions to their component marginal distribution functions [53, 60, 70]. The following theorem by Sklar [85, 52] provides the basis for the copula theory.

Sklar's Theorem: Let F be an N-dimensional cumulative distribution function (*cdf*) with continuous marginal *cdfs* $F_1, F_2, \ldots, F_N$, then there exists a unique copula C such that for $u_1, \ldots, u_N$ in $[-\infty, \infty]$

$$F(u_1, u_2, \ldots, u_N) = C\left(F_1(u_1), F_2(u_2), \ldots, F_N(u_N)\right). \tag{12.35}$$

The joint density is obtained by taking the Nth derivative of (12.35)

$$\begin{aligned} f(\mathbf{u}) &= \frac{\partial^N}{\partial u_1 \ldots \partial u_N} C\left(F_1(u_1), F_2(u_2), \ldots, F_N(u_N)\right) \\ &= f_1(u_1) \ldots f_N(u_N)\, c\left(F_1(u_1), \ldots, F_N(u_N)\right) \\ &= f_p(\mathbf{u})\; c\left(F_1(u_1), F_2(u_2), \ldots, F_N(u_N)\right). \end{aligned} \tag{12.36}$$

From (12.36), it is clear that the copula density $c(.)$ weights the product distribution f_p appropriately to incorporate dependence between the random variables $\{u_n\}_{n=1}^N$.

The hypothesis testing that is considered in (12.18) can be rewritten as a LLRT

$$\Lambda(U_L) = \sum_{l=1}^{L} \log \frac{f(u_{1l}, \ldots, u_{Nl})}{g(u_{1l}, \ldots, u_{Nl})} \underset{H_0}{\overset{H_1}{\gtrless}} t, \tag{12.37}$$

where $U_L = \{[u_1, \ldots, u_N]_l : l = 1, \ldots, L\}$ denote a sequence of L independent identically distributed (i.i.d.) observations with *pdfs* $f(U)$ and $g(U)$ under hypothesis H_1 and H_0, respectively. Substituting *pdfs* f and g given by (12.36), the copula based test static becomes

$$\begin{aligned} \Lambda(U_L) &= \sum_{l=1}^{L} \log \left(\frac{f_p(U_l)}{g_p(U_l)} \frac{c_1(F_1(u_{1l}, \ldots, F_N(u_{Nl})}{c_0(G_1(u_{1l}, \ldots, G_N(u_{Nl})} \right) \\ &= \sum_{l=1}^{L} \sum_{n=1}^{N} \log \frac{f(u_{nl})}{g(u_{nl})} \\ &\quad + \sum_{l=1}^{L} \log \frac{c_1(F_1(u_{1l}), \ldots, F_N(u_{Nl}))}{c_0(G_1(u_{1l}), \ldots, G_N(u_{Nl}))} \end{aligned} \tag{12.38}$$

where the copula density c_i denotes the statistical dependence structure between variables under the hypothesis $H_i,\ i \in \{0, 1\}$.

There are several copulas, namely, Gaussian, Gumbel, Frank, Clayton, Student-t, to name few. Sklar's theorem [85] guarantees the existence of a copula. However, there is no method that one can use to decide that one particular copula is the best for a given set of data. So one has to try several copulas and determine the copula that maximizes the $\Lambda(U_L)$ in (12.38).

The algorithm for finding the suitable copula is presented below:

Algorithm 4:

- Process the PIR, seismic and ultrasonic data and generate $\{u_{pl}, u_{sl}, u_{al}\}$, $\forall l$ for the hypotheses H_1 and H_0.
- Transform each $\{u_i\}|H_j$, $\forall\, i \in \{p, s, a\}$, $j \in \{0, 1\}$ data to the copula scale (unit square).
- Find the copula parameters (one may use the "copulafit" in MATLAB [64]) for various copulas, namely, Gaussian and Student-t copulas.
- Use the copula parameters to generate the copula based joint *pdf*.
- Estimate the value of the joint *pdf* function at each point $\{u_{pl}, u_{sl}, u_{al}\}$.
- Estimate the LLRT for product assumption and the copula version in (12.38) and find the difference to determine the improvement.

For this chapter, Gaussian and Student-t copulas are used and are given by

$$\Phi_{\Sigma}\left(\Phi^{-1}(u_1), \Phi^{-1}(u_2), \ldots, \Phi^{-1}(u_N)\right) \tag{12.39}$$

where Φ and Φ_{Σ} are standard univariate and multivariate *cdfs* with a correlation matrix Σ, respectively,

$$t_{\nu,\Sigma}\left(t_{\nu}^{-1}(u_1), \ldots, t_{\nu}^{-1}(u_N)\right) \tag{12.40}$$

where t_{ν} and $t_{\nu,\Sigma}$ are the standard univariate and multivariate Student-t *cdfs* with ν degrees of freedom and a correlation matrix Σ.

As mentioned earlier, the decisions made by the sensors at an UGS are correlated. In order to see if there is an improvement using copulas to compute the joint *pdf*, Algorithm 4 is applied to the data collected when people and animals (horses) walked along trails. The improvements obtained using copulas over the independent assumption are given in Table 12.7. Clearly, the Gaussian copula has some improvement.

Table 12.7 Improvements in prob. of detection over independent assumption

Copula	Gaussian	Student-t
Improvement in prob. det	0.1003	−0.0126

References

1. Physics info: shock wave. http://physics.info/shock/
2. A. Aljaafreh, A. Al-Fuqaha, Multi-target classification using acoustic signatures in wireless sensor networks: a survey. Sig. Process. Int. J. **4**(4), 174–200 (2010)
3. H. Anton, *Elementary Linear Algebra* (Wiley, New York, 1977)
4. M. Sanjeev Arulampalam, S. Maskell, N. Gordon, T. Clapp, A tutorial on particle filters for online nonlinear/non-gaussian baysian tracking. IEEE Trans. Signal Process. **50**(2), 174–188 (2002)
5. K. Attenborough, K. Ming, K. Horoshenkov, *Predicting Outdoor Sound* (Taylor & Francis, New York, 2007)
6. J. Bedard, S.P. Ferret, A small arms' fire detection system: localization concepts, in *Proceedings of SPIE*, vol. 5071, pp. 497–509 (2003)
7. N. Bergman. Recursive bayesian estimation: navigation and tracking applications. Ph.D. dissertation, Linkoping University, Sweden, 1999
8. D.T. Blackstock, *Fundamentals of Physical Acoustics* (Wiley, New York, 2000)
9. R. Boulic, N. Magnenat-Thalmann, D. Thalmann, A global human walking model with real-time kinematic personification. Vis. Comput. **6**, 344–358 (1990)
10. M. Bradley, J.M. Sabatier, Applications of fresnel-kirchhoff diffraction theory in the analysis of human-motion Doppler sonar grams. J. Acoust. Soc. Am. Express Lett. **128**, 1–10 (2010)
11. M. Bradley, J.M. Sabatier, Using acoustic micro-Doppler sonar to distinguish between human and equine motion, in *RTO-MP-SET-176—NATO workshop on autonomous sensing and multi-sensor integration for ISR applications*, Cardiff, UK, Oct 2011
12. C.J.C. Burges, A tutorial on support vector machines for pattern recognition. Data Min. Knowl. Discov. **2**(2), 121–167 (1998)
13. Z. Chair, P.K. Varshney, Optimal data fusion in multiple sensor detection systems. IEEE Trans. Aerosp. Electron. Syst. **AES–22**(1), 98–101 (1986)
14. Y.T. Chan, K.C. Ho, A simple and efficient estimator for hyperbolic location. IEEE Trans. Signal Process. **42**(8), 1905–1915 (1994)
15. Q. Chen, R. Liu, On the explanation of spatial smoothing in music algorithm for coherent sources, in *International Conference on Information Science and Technology* (2011), pp. 699–702
16. Y. Chen, M.R. Gupta, *EM Demystified: An Expectation Maximization Tutorial* (University of Washington, Seattle, 2010)
17. D. Colton, R. Kress, *Inverse Acoustic and Electromagnetic Scattering Theory*, Applied Mathematical Sciences (Springer, Berlin, 1998)
18. M.J. Crocker, *Encyclopedia of Acoustics* (Wiley, New York, 1997)
19. A. Cuyt, V. Petersen, B. Verdonk, H. Waadeland, W.B. Jones, *Handbook of Continued Fractions for Special Functions* (Springer, Berlin, 2008)

T. Damarla, *Battlefield Acoustics*, DOI 10.1007/978-3-319-16036-8

20. R. Damarla, D. Ufford, Personnel detection using ground sensors, in *Proceedings of SPIE*, vols. 6562, 656205 (2007)
21. T. Damarla, M. Bradley, A. Mehmood, J.M. Sabatier, Classification of animals and people ultrasonic signatures. IEEE Sens. J. **13**(5), 1464–1472 (2013)
22. T. Damarla, WC. Kirkpatrick Alberts, Helicopter tracking using acoustic arrays, in *2014 17th International Conference on Information Fusion (FUSION)* (2014), pp. 1–7
23. T. Damarla, L. Kaplan, G. Whipps, Sniper localization using acoustic asynchronous sensors. IEEE Sens. J. **10**(9), 1469–1478 (2010)
24. T. Damarla, A. Mehmood, J. Sabatier, Detection of people and animals using non-imaging sensors, in *Proceedings of the 14th International Conference on Information Fusion (FUSION)*, (2011), pp. 1–8
25. T. Damarla, T. Pham, D. Lake, Algorithm for classifying multiple targets using acoustic signatures, in *Proceedings of SPIE*, vol. 5429 (2004), pp. 421–427
26. T. Damarla, J. Sabatier, A. Ekimov, Personnel detection at a border crossing, in *Proceedings of Military Sensing Symposium National* (2010)
27. A.P. Dempster, New methods for reasoning towards posterior distributions based on sample data. Ann. Math. Stat. **37**, 355–374 (1966)
28. A.P. Dempster, Upper and lower probabilities induced by a multivalued mapping. Ann. Math. Stat. **38**, 325–339 (1967)
29. A.P. Dempster, A generalization of bayesian inference. J. R. Stat. Soc.: B (Methodol.) **30**, 205–247 (1968)
30. A.P. Dempster, N.M. Laird, D.B. Rubin, Maximum likelihood from incomplete data via the em algorithm. J. R. Stat. Soc.: B **39**(1), 1–38 (1977)
31. P.M. Djuric, J.H. Kotecha, J. Zhang, Y. Huang, T. Ghirmai, M.F. Bugallo, J. Miguez, Particle filtering. IEEE Signal Process. Mag. **20**(5), 19–38 (2003)
32. C.B. Do, S. Batzoglou, What is the expectation maximization algorithm. Comput. Biol. Nat. Biotechnol. **26**(8), 897–899 (2008)
33. A. Doucet, *On sequential monte carlo methods for bayesian filtering*. Department of Engineering, University of Cambridge, UK, Technical Report (1998)
34. A. Doucet, J.F.G. de Freitas, N.J. Gordon, in *An Introduction to sequential Monte Carlo Methods*, Sequential Monte Carlo Methods in Practice (Springer, New York, 2001)
35. A. Doucet, S.J. Godsill, C. Andrieu, On sequential monte carlo sampling methods for bayesian filtering. Stat. Comput. **3**, 197–208 (2000)
36. G.L. Duckworth, D.C. Gilbert, J.E. Barger, Acoustic counter-sniper system, in *Proceedings of the SPIE*, vol. 2938 (1997), pp. 262–275
37. J. DuMond, E. Cohen, W. Panofski, E. Deeds, Determination of the wave forms and laws of propagation and dissipation of ballistic shock waves. J. Acoust. Soc. Am. **18**, 97–118 (1946)
38. A. Ekimov, J.M. Sabatier, Rhythm analysis of orthogonal signals from human walking. J. Acoust. Soc. Am. **129**(3), 1306–1314 (2011)
39. F.A. Everest, *Master Handbook of Acoustics*, 4th edn. (McGraw-Hill Book Company, New York, 2001)
40. L. Fausett, *Fundamentals of Neural Networks: Architectures, Algorithms, and Applications* (Printice Hall, Englewood Cliffs, 1994)
41. K.F. Gauss, *Theory of Motion of the Heavenly Bodies* (Dover Publications, New York, 1963)
42. J.L. Geisheimer, W.S. Marshall, E. Greneker, A continous-wave (cw) radar for gait analysis, in *Proceedings of 35th Asilomar Conference on Signals, Systems and Computers* (2001), pp. 834–838
43. M.A. Goodrich, A tutorial on simple particle filters. http://students.cs.byu.edu/cs470ta/goodrich/fall2006/lectures/ParticleFilterTutorial.pdf
44. B. Guo, M.S. Nixon, T.R. Damarla, Acoustic information fusion for ground vehicle classification, in *11th International Conference on Information Fusion 2008* (2008)
45. J. Hartikainen, A. Solin, S. Sarkka, Optimal filtering with Kalman filters and smoother—a manual for the matlab toolbox EKFUKF. http://becs.aalto.fi/en/research/bayes/ekfukf. Accessed Aug 2011

46. S. Haykin, *Adaptive Filter Theory* (Printice Hall, Englewood Cliffs, 1991)
47. M.A. Hearst, S.T. Dumais, E. Osman, J. Platt, B. Scholkopf, Support vector machines. IEEE Intell. Syst. Appl. **13**(4), 18–28 (1998)
48. R. Hecht-Nielsen, *Neurocomputing* (Addison-Wesley Publishing Company Inc., Reading, 1991)
49. K.M. Houston, D.P. McGaffigan, Personnel detection at a border crossing, in *Proceedings of SPIE*, vol. 5090 (2003), pp. 162–173
50. P. Huang, T. Damarla, M. Johnson, Acoustic Doppler sonar for gait recognition, in *Proceedings of the 14th International Conference on Information Fusion Chicago*, Chicago, IL, pp. 27–32, 5–8 July 2011
51. A. Hyvarinen, J. Karhunen, E. Oja, *Independent Component Analysis* (Wiley, New York, 2001)
52. S.G. Iyengar, P.K. Varshney, T. Damarla, A parametric copula-based framework for hypothesis testing using heterogeneous data. IEEE Trans. Signal Process. **59**(5), 2308–2319 (2011)
53. H. Joe, *Multivariate Dependence and Related Concepts*. Monographs on Statistics and Applied Probability (Chapman and Hall/CRC, New York, 1997)
54. D.H. Johnson, D.E. Dudgeon, *Array Signal Processing—Concepts and Techniques* (Prentice Hall, Upper Saddle River, 1993)
55. S.J. Julier, J.K. Uhlmann, Unscented filtering and nonlinear estimation. Proc. IEEE **92**(3), 401–422 (2004)
56. K. Kalgaonkar, B. Raj, Acoustic Doppler sonar for gait recognition, in *Proceedings of IEEE Conference on Advanced Video and Signal Based Surveillance* (2007) pp. 27–32
57. R. E. Kalman, A new approach to linear filtering and prediction problems. Trans. ASME J. Basic Eng. 35–45 (1960)
58. S.M. Kay, *Fundamentals of Signal Processing—Estimation Theory* (Prentice Hall Inc, Upper Saddle River, 1993)
59. C.H. Knapp, G.C. Carter. The generalized correlation method for estimation of time delay, in *IEEE Transactions Acoustics, Speech, Signal Processing*, vol. ASSP-24, pp. 320–327, April 1976
60. D. Kurowicka, R. Cooke, *Uncertainty Analysis with High Dimensional De-pendence Modeling* (Wiley, New York, 2006)
61. J. Li, P. Stoica, Z. Wang, On robust capon beamforming and diagonal loading. IEEE Trans. Signal Process. **51**(7), 1702–1715 (2003)
62. R.P. Lippman, An introduction to computing with neural nets. IEEE ASSP Mag. **4**, 4–22 (1987)
63. S. Liu, C. Zhang, Y. Huang, Research on acoustic source localization using time difference of arrival measurements, in *Internation Conference on Measurement, Information and Control*, pp. 220–224, May 2012
64. MATLAB, MATLAB. http://www.mathworks.com. Accessed July 2012
65. A. Mehmood, J.M. Sabatierand, M. Bradley, A. Ekimov, Extraction of the velocity of walking human's body segments using ultrasonic doppler. J. Acoust. Soc. Am. **128**(5):EL316–EL322 (2010)
66. H.B. Mitchell, *An Introduction to Multi-Sensor Data Fusion* (Springer, New York, 2007)
67. T.K. Moon, The expectation maximization algorithm. IEEE Signal Process. Mag. 47–60 (1996)
68. J. Myung, Tutorial on maximum likelihood estimation. J. Math. Psychol. **47**, 90–100 (2003)
69. M.V. Namorato, A concise history of acoustics in warfare. Appl. Acoust. **59**, 101–135 (2000)
70. R.B. Nelsen, *An Introduction to Copulas* (Springer, New York, 1999)
71. N.J. Nilsson, *Learning Machines–Foundations of Trainable Pattern-Classifying Systems* (McGraw-Hill Book Company, New York, 1965)
72. Y.-H. Pao, *Adaptive Pattern Recognition and Neural Networks* (Addison-Wesley Publishing Company Inc., Reading, 1989)
73. A. Papoulis, Probability, *Random Variables and stochastic Processes*, 3rd edn. (McGraw-Hill Book Company, New York, 1991)

74. PILAR, PILAR sniper countermeasures system. http://www.canberra.com/products
75. R.L. Plackett, The principle of the arithmetic mean. Biometrika 45–130 (1958)
76. H.V. Poor, *An Introduction to Signal Detection and Estimation* (Springer, Berlin, 1994)
77. R.H. Randall, *An Introduction to Acoustics* (Dover Publications Inc., Mineola, 2005)
78. C.D. Ross, Outdoor sound propagation in the U.S. civil war. http://asa.aip.org/Echoes/Vol9No1/EhoesWinter1999.html
79. D.E. Rumelhart, J. McClelland, *Parallel Distributed Processing: Explorations in the Microstructure of Cognition. Vol. 1: Fondations* (MIT Press, Cambridge, 1986)
80. J. Sallai, P. Volgyesi, K. Pence, A. Ledeczi, Fusing distributed muzzle blast and shockwave detections, in *14th International Conference on Information Fusion*, pp. 748–755, July 2011
81. E.M. Salomons, *Computational Atmospheric Acoustics* (Kluwer Academic Publishers, Norwell, 2001)
82. S. Sarkka, On unscented kalman filtering for state estimation of continuous-time nonlinear systems. IEEE Trans. Automat. Control **52**(9), 1631–1641 (2007)
83. A.H. Sayed, A. Tarihat, N. Khajehnouri, Network-based wireless location: challenges faced in developing techniques for accurate wireless location information. Signal Process. Mag. IEEE **22**(4), 24–40 (2005)
84. G. Shafer, *A Mathematical Theory of Evidence* (Princeton University Press, Princeton, 1976)
85. A. Sklar, Fonctions de repartition a *n* dimensions et leurs marges. Publ. Inst. Stat. Univ. Paris **8**, 229–231 (1959)
86. J.O. Smith, J.S. Abel, Closed-form least-squares source location estimation from range-difference measurements. IEEE Trans. Acoust. Speech, Signal Process. **ASSP-35**, 1661–1669 (1987)
87. H.W. Sorenson, *Parameter Estimation: Principles and Problems* (Mercel Dekker Inc., New York, 1980)
88. R.B. Stoughton, SAIC SENTINEL acoustic counter-sniper system. Proc. SPIE **2938**, 276–284 (1997)
89. S.P. Parker (ed.), *Acoustics Source Book*, 3rd edn. (McGraw-Hill Book Company, New York, 1987)
90. C.W. Therrien, *Discrete Random Signals and Statistical Signal Processing* (Prentice Hall, Englewood Cliffs, 1992)
91. S.C.A. Thomopoulos, R. Viswanathan, D.C. Bougoulias, Optimal decision fusion in multiple sensor systems. Aerosp. Electron. Syst. IEEE Trans. **AES-23**(5):644–653 (1987)
92. H.L. Van Trees, *Optimum Array Processing—Part IV of Detection, Estimation, and Modulation Theory* (Wiley, New York, 2002)
93. V. Vapnik, *Estimation of Dependencies Based on Empirical Data* (Springer, New York, 1982)
94. V. Vapnik, *The Nature of Statistical Learning Theory* (Springer, New York, 1995)
95. P. Volgyesi, G. Balogh, A. Nadas, C.B. Nash, A. Ledeczi, Shooter localization and weapon classification with soldier-wearable networked sensors, in *Proceedings of MobiSys'07*, San Juan, Puerto Rico, USA (June 2007), pp. 113–126
96. M. Wax, T. Kailath, Detection of signals by information theoretic criterion. IEEE Trans. Acoust. Speech Signal Process. **ASSP-33**(2):387–392 (1985)
97. G.B. Whitham, The flow pattern of a supersonic projectile. Commun. Pure Appl. Math. **5**, 301–348 (1952)
98. B. Widrow, M.E. Hoff Jr., Adaptive switching circuits, in *IRE WESCON Convention Record, part 4*, pp. 96–104 (1960)
99. L.A. Zadeh, A simple view of the dempster-shafer theory of evidence and its implication for the rule of combination. AI Mag. **7**(2), 85–90 (1986)
100. Y. Zhang, W.H. Abdulla, *A Comparative Study of Time-Delay Estimation Techniques Using Microphonee Arrays* (2005)
101. Z. Zhang, P. Pouliquen, A. Waxman, A.G. Andreou, Acoustic micro-Doppler gait signatures of humans and animals, in *Conference on Information Sciences and Systems*, Chicago, IL (March 2007), pp. 627–630

Index

T. Damarla, *Battlefield Acoustics*, DOI 10.1007/978-3-319-16036-8