IVAN TASHEV

Sound Capture and Processing

PRACTICAL APPROACHES

Companion Website

WILEY

Sound Capture and Processing

Sound Capture and Processing
Practical Approaches

Ivan J. Tashev
Microsoft Research, USA

WILEY
A John Wiley and Sons, Ltd., Publication

This edition first published 2009

Registered office
John Wiley & Sons Ltd, The Atrium, Southern Gate, Chichester, West Sussex, PO19 8SQ, United Kingdom

Library of Congress Cataloging-in-Publication Data

Tashev, Ivan J. (Ivan Jelev)
Sound capture and processing : practical approaches / Ivan J. Tashev.
p. cm.
Includes index.
ISBN 978-0-470-31983-3 (cloth)
1. Speech processing systems. 2. Sound–Recording and reproducing–Digital techniques. 3. Signal processing–Digital techniques. I. Title.
TK7882.S65T37 2009
621.382'8–dc22

2009011987

A catalogue record for this book is available from the British Library.

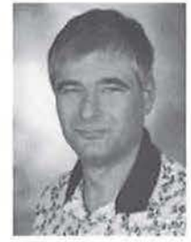
ISBN 978-0-470-31983-3 (H/B)

Typeset in 11/13pt Times by Thomson Digital, Noida, India.
Printed and bound in Great Britain by CPI Antony Rowe, Chippenham, Wiltshire.

To my family: the time to write this book was taken from them

Contents

About the Author



Dr Ivan Tashev took both his Engineering Diploma in Electronics and PhD in Computer Science degrees at the Technical University of Sofia, Bulgaria, in 1984 and 1990 respectively. After his graduation he worked as R&D engineer and researcher in the R&D Department of the same university. Dr Tashev became assistant professor in 1989. He created and taught two courses, "Data and signal processing" and "Programming of real-time systems" to the students of fourth and fifth year in the Department of Electronics.

Dr Tashev joined Microsoft in 1998 and held positions in various product teams until 2001 when he moved to Microsoft Research. Here he was involved in projects such as RingCam (now a Microsoft product – Round Table Device), microphone array (currently part of Windows Vista), and many others related to sound capturing devices and audio signal processing. Currently he is a member of the Speech Technology Group in Microsoft Research lab at the Microsoft headquarters in Redmond, Washington.

Dr Ivan Tashev is senior member of IEEE and IEEE Signal Processing Society, member of Audio Engineering Society and its Pacific Northwest Committee. He is reviewer for most of the audio and signal processing journals and conferences. Dr Tashev has published three books, more than fifty scientific papers and is listed as inventor of five granted U.S. patents and seventeen U.S. patent applications.

The research interests of Dr Tashev include sound capturing devices, signal processing for arrays of transducers, speech enhancement algorithms, and signal processing of audio, speech and biological signals.

Foreword

Just a couple of decades ago we would think of "sound capture and processing" as the problems of designing microphones for converting sounds from the real world into electrical signals, as well as amplifying, editing, recording, and transmitting such signals, mostly using analog hardware technologies. That's because our intended applications were mostly analog telephony, broadcasting, and voice and music recording. We have come a long way: small digital audio players have replaced bulky portable cassette tape players, and people make voice calls mostly via digital mobile phones and voice communication software in their computers. Thanks to the evolution of digital signal processing technologies, we now focus mostly on processing sounds not as analog electrical signals, but rather as digital files or data streams in a computer or digital device. We can do a lot more with digital sound processing, such as transcribe speech into text, identify persons speaking, recognize music from humming, remove noises much more efficiently, add special effects, and so much more. Thus, today we think of sound capture as the problem of digitally processing the signals captured by microphones so as to improve their quality for best performance in digital communications, broadcasting, recording, recognition, classification, and other applications.

This book by Ivan Tashev provides a comprehensive yet concise overview of the fundamental problems and core signal processing algorithms for digital sound capture, including ambient noise reduction, acoustic echo cancellation, and reduction of reverberation. After introducing the necessary basic aspects of digital audio signal processing, the book presents basic physical properties of sound and propagation of sound waves, as well as a review of microphone technologies, providing the reader with a strong understanding of key aspects of digitized sounds. The book discusses the fundamental problems of noise reduction, which are usually solved via techniques based on statistical models of the signals of interest (typically voice) and of interfering signals. An important discussion of properties of the human auditory system is also presented; auditory models can play a very important role in algorithms for enhancing audio signals in communication and recording/playback applications, where the final destination is the human ear.

Microphone arrays have become increasingly important in the past decade or so. Thanks to the rapid evolution and reduction in cost of analog and digital electronics in recent years, it is inexpensive to capture sound through several channels, using an array of microphones. That opens new opportunities for improving sound capture, such as detecting the direction of incoming sounds and applying spatial filtering techniques. The book includes two excellent

chapters whose coverage goes from the basics of microphone array configurations and delay-and-sum beamforming, to modern sophisticated algorithms for high-performance multichannel signal enhancement.

Acoustic echoes and reverberation are the two most important kinds of signal degradations in many sound capture scenarios. If you're a professional singer, you probably don't mind holding a microphone or wearing a headset with a microphone close to your mouth, but most of us prefer microphones to be invisible, far away from our mouths. That means microphone will capture not only our own voices, but also reverberation components because of sound reflections from nearby walls, as well as echoes of signals that are being played back from loudspeakers. Removing such undesirable artifacts presents significant technical challenges, which are well addressed in the final two chapters, which present modern algorithms for tackling them.

A key quality of this book is that it presents not only fundamental theoretical analyses, models, and algorithms, but it also considers many practical aspects that are very important for the design of real-world engineering solutions to sound capture problems. Thus, this book should be of great appeal to both students and engineers.

I have had the pleasure of working with Ivan on research and development of sound capture systems and algorithms. His enthusiasm, deep engineering and mathematical knowledge, and pragmatic approaches were all contagious. His work has had significant practical impact, for example the introduction of multichannel sound capture and processing modules in the Microsoft Windows operating system. I have learned a considerable amount about sound capturing and processing from my interactions with Ivan, and I am sure you will, as well, by reading this book. Enjoy!

Henrique Malvar
Managing Director
Microsoft Research
Redmond Laboratory

Preface

Capturing and processing sounds is critical in mobile and handheld devices, communication systems, and computers using automatic speech recognition. Devices and technologies for proper conversion of sounds to electric signals and removing unwanted parts, such as noise and reverberation, have been used since the first telephones. They evolved, becoming more and more complex. In many cases the existing algorithms exceed the abilities of typical processors in these devices and computers to provide real-time processing of the captured signal.

This book will discuss the basic principles for building an audio processing stack, sound capturing devices, single-channel speech-enhancement algorithms, and microphone arrays for sound capture and sound source localization. Further, algorithms will be described for acoustic echo cancellation and de-reverberation – building blocks of a sound capture and processing stack for telecommunication and speech recognition. Wherever possible the various algorithms are discussed in the order of their development and publication. In all cases the aim is to try to give the larger picture – where the technology came from, what worked and what had to be adapted for the needs of audio processing. This gives a better perspective for further development of new audio signal processing algorithms.

Even the best equations and signal processing algorithms are not worth anything before being implemented and verified by processing of real data. That is why, in this book, stress is placed on experimenting with recorded sounds and implementation of the algorithms. In practice, frequently a simpler model with fewer parameters to estimate works better than a more precise but more complex model with a larger number of parameters. With the latter one has either to sacrifice estimation precision or to increase the estimation time. This balance of simplicity, precision, and reaction time is critical for real-time systems, where on top of everything we have to watch out for parameters such as latency, consumed memory, and CPU time.

Most of the algorithms and approaches described in this book are based on statistical models. In mathematics, a single example cannot prove but can disprove a theorem. In statistical signal processing, a single example is ... just a sample. What matters is careful evaluation of the algorithms with a good corpus of speech or audio signals, distributed in their signal-to-noise ratios, type of noise, and other parameters – as close as possible to the real problem we are trying to solve.

The solution of practically any signal processing problem can be improved by tuning the parameters of the algorithm, provided we have a proper criterion for optimality. There are always adaptation time constants, thresholds, which cannot be estimated and their values have to be adjusted experimentally. The mathematical models and solutions we use are usually

optimal in one or another way. If they reflect properly the nature of the process they model, then we have a good solution and the results are satisfactory. In all cases it is important to remember that we do not want a "minimum mean-square error solution," or a "maximum-likelihood solution," or even a "log minimum mean-square error solution." We do not want to improve the signal-to-noise ratio. What we want is for listeners to perceive the sound quality of the processed signal as better – improved – compared to the input signal. From this perspective, the final judge of how good is an algorithm is the human ear, so use it to verify the solution. Hearing is an important sense for humans and animals. In many places in this book are provided examples of how humans and animals hear and localize sounds – this explains better some signal processing approaches and brings biology-inspired designs for sound capture and processing systems.

In many cases the signal processing chain consists of several algorithms for sound capture and speech enhancement. The practice shows us that a sequence of separately optimized algorithms usually provides suboptimal results. Tuning and optimization of the designed sound capturing system end-to-end is a must if we want to achieve best results.

For further information please visit http://www.wiley.com/go/tashev_sound

Ivan Tashev
Redmond, WA
USA

Acknowledgements

I want to thank the Book Program in MathWorks and especially Dee Savageau, Naomi Fernandes, and Meg Vulliez for the help and responsiveness. The MATLAB® scripts, part of this book, were tested with MATLAB® R2007a, provided as part of this program.

I am grateful to my colleagues from Microsoft Research Alex Acero, Amitav Das, Li Deng, Dinei Florencio, Cormac Herley, Zicheng Liu, Mike Seltzer, and Cha Zhang. They read the chapters of this book and provided valuable feedback.

And last, but not least, I want to express my great pleasure working with the nice and helpful people from John Wiley & Sons, Ltd. During the long process from proposal, through writing, copyediting, and finalizing the book with all the details, they were always professional, understanding, and ready to suggest the right solution. I was lucky enough to work with Tiina Ruonamaa, Sarah Hinton, Sarah Tilley, and Catlin Flint – thank you all for everything you did during the process of writing this book!

4

Single-channel Noise Reduction

This chapter deals with noise reduction of a single channel. We assume that we have a mixture of a useful signal, usually human speech, and an unwanted signal – which we call noise. The goal of this type of processing is to provide an estimate of the useful signal – an enhanced signal with better properties and characteristics.

The problem with a noisy speech signal is that a human listener can understand a lower percentage of the spoken words. In addition, this understanding requires more mental effort on the part of the listener. This means that the listener can quickly lose attention – an unwanted outcome during meetings over a noisy telephone line, for example. If the noisy signal is sent to a speech recognition engine, the noise reduces the recognition rate as it masks speech features important for the recognizer.

With noise-reduction algorithms, as with most other signal processing algorithms, there are multiple trade-offs. One is between better reduction of the unwanted noise signal and introduction of undesired effects – additional signals and distortions in the wanted speech signal. From this perspective, while improvement in the signal-to-noise ratio (SNR) remains the main evaluation criterion of the efficiency of these algorithms, subjective listening tests or objective sound quality evaluations are also important. The perfect noise-reduction algorithm will make the main speaker's voice more understandable so that it seems to stand out, while preserving relevant background noise (train station, party sounds, and so on). Such an algorithm should not introduce noticeable distortions in either foreground (wanted speech) or background (unwanted noise) signals.

Most single-channel algorithms are based on building statistical models of the speech and noise signals. In this chapter we will look at the commonly used approaches for suppression of noise, the algorithms to distinguish between noise and voice (called "voice activity detectors"), and some adaptive noise-canceling algorithms. Exercises with implementation of some of these algorithms will be provided for better understanding of the processes inside the noise suppressors.

Sound Capture and Processing Ivan J. Tashev

4.1 Noise Suppression as a Signal Estimation Problem

Let the speech signal be $x(t)$. This signal is captured after it has been mixed with noise $d(t)$. We can assume that these two signals are statistically independent. The capturing process is linear, so the captured signal is $y(t) = x(t) + d(t)$. The goal of the noise reduction is to estimate the speech signal $x(t)$ using the observed signal $y(t)$ and some known properties of both speech and noise signals. This process is shown in Figure 4.1. We have the unobservable part when the speech and noise signals are mixed. In the observable part we have only the observed signal and some a-priori knowledge about the character of the signals. The estimation process is optimal in one way or another; that is, it satisfies a certain criterion.

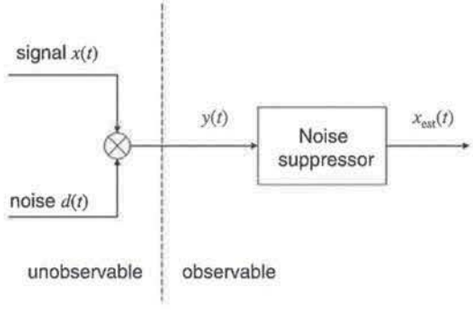


Figure 4.1 Noise reduction as a signal estimation problem

As we saw in Chapter 2, the majority of the audio processing algorithms work with audio frames, obtained by the overlap–add procedure, described in the same chapter. This is why we will look at most of the noise suppression algorithms from the perspective of processing in the frequency domain, after short-time spectral conversion. Since the transformation is linear, we have $Y_k^{(n)} = X_k^{(n)} + D_k^{(n)}$ where k is the frequency bin index and n is the frame index. Under these conditions it is a common assumption that the values of the frequency bins are statistically independent for both noise and speech signals. This allows derivation of the speech signal estimation algorithms for each frequency bin independently, which greatly simplifies the algorithms and the corresponding equations.

4.2 Suppression Rules

4.2.1 Noise Suppression as Gain-based Processing

The early work of Norbert Wiener [1] has an impact on signal processing algorithms by providing an original look from a statistical point of view. He derived an optimal

filter for estimation of a signal corrupted by noise in the time domain. Later derivations for representation in the frequency domain found that this optimal estimator applies a real-valued gain to the complex spectral vector. In general, human perception of speech is insensitive to the phase of the signal [2], and the same is true for automated speech recognizers. In later work, Ephraim and Malah [3] proved that the optimal phase estimator for a signal corrupted with noise just takes the phase of the noisy signal. In this manner, noise reduction can be viewed as an application of a time-varying, non-negative, real-valued gain $H_k^{(n)}$ to each frequency bin k of the observed signal $Y_k^{(n)}$ to obtain the estimate $\hat{X}_k^{(n)}$ of the original signal spectrum:

$$\hat{X}_k^{(n)} = H_k^{(n)} \cdot Y_k^{(n)}. \tag{4.1}$$

The time-varying real-valued gain H_k is called the *suppression rule* and it is estimated for each frame. Note that the complex value of the signal estimation for this frequency bin $\hat{X}_k^{(n)}$ keeps the phase of the observed complex signal $Y_k^{(n)}$ because $H_k^{(n)}$ is real-valued.

4.2.2 Definition of A-Priori and A-Posteriori SNRs

In most algorithms for estimation of the suppression rule, a-priori and a-posteriori SNRs are involved. They were defined for the first time in [4]. The authors model the elements of the noise spectrum as independent, identically distributed Gaussian variables with a zero mean and variances $\lambda_d(k)$:

$$D_k \sim \mathbb{N}(0, \lambda_d(k)). \tag{4.2}$$

In the same paper they model the signal as a stationary sum of sinusoidal signals with magnitude A_k which is an estimation of the signal magnitude for this frequency bin. In general we can say that $\lambda_d(k) \triangleq E\{|D_k|^2\}$ and $\lambda_x(k) \triangleq E\{|X_k|^2\}$. Then we can define the a-priori SNR ξ_k and a-posteriori SNR γ_k as

$$\begin{aligned} \xi_k &\triangleq \frac{\lambda_x(k)}{\lambda_d(k)} \\ \gamma_k &\triangleq \frac{|Y_k|^2}{\lambda_d(k)}. \end{aligned} \tag{4.3}$$

Note that while ξ_k is an average (statistical) SNR, γ_k can be viewed as a momentary SNR.

4.2.3 Wiener Suppression Rule

The Wiener estimator for a discrete signal corrupted by noise was initially derived in the time domain as an N-tap FIR (finite impulse response) filter; that is:

$$\hat{x}(n) = \sum_{i=0}^{N-1} h_i^* \cdot y(n-i). \tag{4.4}$$

Then the estimation error will be

$$e(n) = x(n) - \hat{x}(n) \tag{4.5}$$

and the goal is to find filter $\mathbf{h}_{\text{opt}}$ with coefficients h_i that minimizes the estimation error:

$$\mathrm{E}\{|e(n)|^2\} = \mathrm{E}\{e(n)e^*(n)\}. \tag{4.6}$$

After taking first derivatives, the filter that zeroes them is

$$\mathbf{h}_{\text{opt}} = \mathbf{R}_{yy}^{-1} \cdot \mathbf{r}_{yy}(0) \tag{4.7}$$

where $\mathbf{r}_{yy}(n)$ is the autocorrelation vector

$$\mathbf{r}_{yy}(n) = [r_{yy}(n),\, r_{yy}(n-1), \cdots,\, r_{yy}(n-N+1)]^{\mathrm{T}} \tag{4.8}$$

and $\mathbf{R}_{yy}$ is the autocorrelation matrix

$$\mathbf{R}_{yy} = \begin{bmatrix} r_{yy}(0) & r_{yy}(1) & \cdots & r_{yy}(N-1) \\ r_{yy}^*(1) & r_{yy}(0) & \cdots & r_{yy}(N-2) \\ \vdots & \vdots & \ddots & \vdots \\ r_{yy}^*(N-1) & r_{yy}^*(N-2) & \cdots & r_{yy}(0) \end{bmatrix}. \tag{4.9}$$

Considering the stationarity of the signals, and the fact that they are statistically independent, in the frequency domain the optimal filter derivation is much simpler. The filter that minimizes the derivatives is

$$\mathbf{S}_{ss}^* - \mathbf{H}_{\text{opt}} \cdot \mathbf{S}_{yy} = 0 \tag{4.10}$$

where $\mathbf{S}_{ss}$ is the power spectral density of the signal and $\mathbf{S}_{yy}$ is the power spectral density of the observation. Due to statistical independence, $\mathbf{S}_{yy} = \mathbf{S}_{ss} + \mathbf{S}_{dd}$ and the optimal filter is

$$H_{\text{opt}} = \frac{S_{ss}}{S_{dd} + S_{ss}} = \frac{\lambda_s}{\lambda_d + \lambda_s}. \tag{4.11}$$

Equation 4.11 can easily be converted in terms of a-priori SNR for each frequency bin as

$$H_k = \frac{\xi_k}{1 + \xi_k} \tag{4.12}$$

which is the *Wiener suppression rule*. This rule minimizes the mean square error of the estimated signal.

4.2.4 Artifacts and Distortions

The derived elegant solution for the optimal suppression rule stumbles on some difficulties when it is applied to real noise-reduction systems – estimation of the a-priori SNR. While obtaining the noise variation $\lambda_d(k)$ is relatively easy, estimation of the signal variation is not trivial. The logical step of using the a-posteriori SNR γ_k to estimate ξ_k leads to the approximate solution

$$H_k^{(n)} = \frac{\xi_k}{1 + \xi_k} \approx \frac{\gamma_k - 1}{\gamma_k} = \frac{|Y_k^{(n)}|^2 - \lambda_d(k)}{|Y_k^{(n)}|^2} \tag{4.13}$$

which is easier to implement in practice. By definition, $H_k^{(n)}$ is non-negative and real; but, for values of $|Y_k^{(n)}|^2$ smaller than $\lambda_d(k)$, Equation 4.13 can have negative values. The common approach is to limit the values of the suppression rule to be non-negative. Another practical problem is potential division by zero for frequency bins where the input signal has a zero value. This is solved by adding a small number ε, and we finally have the Wiener suppression rule widely used in practice:

$$H_k^{(n)} = \frac{\max[0, |Y_k^{(n)}|^2 - \lambda_d(k)]}{|Y_k^{(n)}|^2 + \varepsilon}. \tag{4.14}$$

This approximate rule provides good noise suppression and improves the SNR of the output signal. On the down side, in the output signal some artifacts and distortions may be audible.

The artifacts appear during the silence segments when a very specific type of noise, called "musical noise," can be heard. Investigations have shown that this is due to the fact that some frequency bins are zeroed during these segments owing to the way

suppression gain is estimated via Equation 4.14. During silence segments, where we only have a noise signal with variation $\lambda_d(k)$, a substantial number of cases $|Y_k^{(n)}|^2$ will be smaller than $\lambda_d(k)$ and the corresponding frequency bin will be zeroed. This "patchy" spectrogram, converted to the time domain, has the specific musical noise sound.

The Wiener filter is a minimum mean-square estimator, which provides an approximate value of the output signal. Introducing some distortions when compared with the original signal is inevitable. The problem with distortions, audible as metallic and unnatural sound of the estimated speech signal, increases in signals with low SNR owing to the approximation in Equation 4.13.

4.2.5 *Spectral Subtraction Rule*

The musical noises and distortions in the output signal stimulated the search for better estimators. Considering that less suppression means less musical noise and less distortion, the spectral subtraction rule [5] was introduced. It is defined as

$$H_k^{(n)} = \sqrt{\frac{\gamma_k - 1}{\gamma_k}} \tag{4.15}$$

and frequently used with the approximation

$$H_k^{(n)} \approx \sqrt{\frac{\max[0, |Y_k^{(n)}|^2 - \lambda_d(k)]}{|Y_k^{(n)}|^2 + \varepsilon}}. \tag{4.16}$$

The rule is optimal in the sense of estimation of the speech magnitude spectrum. The overall noise suppression is less, but the spectral subtraction suppression rule has a lower distortion of the estimated speech signal; that is, the output sounds better to the human ear. The problem with the musical noise remains, as many frequency bins are still zeroed during silence periods.

4.2.6 *Maximum-likelihood Suppression Rule*

McAulay and Malpass [4] proposed a new suppression rule, optimal in the maximum-likelihood sense. They modeled the speech signal as a deterministic waveform of unknown amplitude and phase, and the noise as a random Gaussian signal. Under these

conditions the maximum-likelihood suppression rule is

$$H_k^{(n)} = \frac{1}{2} + \frac{1}{2}\sqrt{\frac{|Y_k^{(n)}|^2 - \lambda_d(k)}{|Y_k^{(n)}|^2}}. \tag{4.17}$$

The first thing to notice is that this suppression rule never becomes zero. This completely eliminates the musical noise. A second fact to notice is that its minimal gain value is actually quite high and the rule never goes below 0.5. This substantially reduces the noise-suppression capabilities of the maximum-likelihood suppression rule; it has the lowest noise suppression among the suppression rules we discuss in this book. The same practical measures to limit the value under the square root to be non-negative and to prevent division by zero as in Equation 4.16 should be taken here as well.

Figure 4.2 shows the three suppression rules, discussed so far, as a function of the a-posteriori SNR. It is obvious that Wiener filtering suppresses the most, and maximum-likelihood suppresses the least, of the signal energy. For performance comparison of the different suppression rules, see later in this chapter. We will return to the work of McAulay and Malpass in the section about suppression with the uncertain presence of a speech signal.

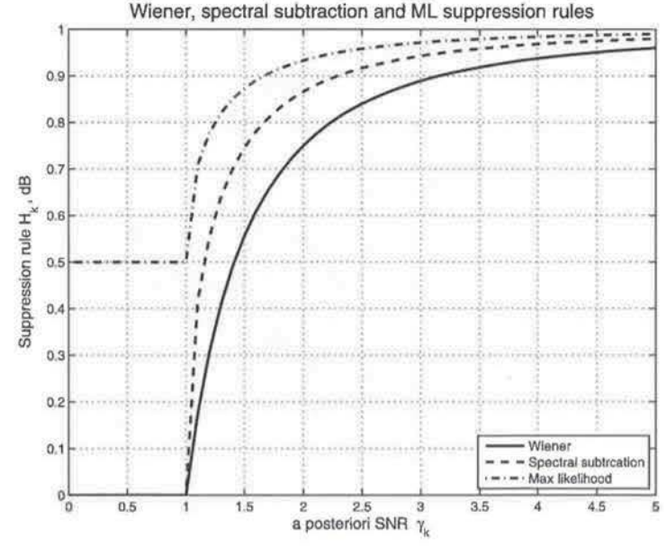


Figure 4.2 Wiener, spectral subtraction, and maximum-likelihood suppression rules as functions of the a-posteriori SNR

4.2.7 Ephraim and Malah Short-term MMSE Suppression Rule

Ephraim and Malah [3] model both speech and noise signals as zero-mean random Gaussian signals. Under the conditions of short term-spectral estimation, they derive a suppression rule, known in the form

$$H_k = \frac{\sqrt{\pi\nu_k}}{2\gamma_k}\left[(1+\nu_k)I_0\left(\frac{\nu_k}{2}\right)+\nu_k I_1\left(\frac{\nu_k}{2}\right)\right]\exp\left(\frac{\nu_k}{2}\right). \tag{4.18}$$

Here $I_0(\cdot)$ and $I_1(\cdot)$ denote the modified Bessel functions of zero- and first-order, respectively, and

$$\nu_k \triangleq \frac{\xi_k}{1+\xi_k}\gamma_k. \tag{4.19}$$

The spectral magnitude estimator, given by Equation 4.18, is optimal in the MMSE sense. It provides good noise suppression comparable to that of the Wiener filter, while maintaining lower distortions and artifacts. For the first time, the suppression rule is defined as a function of both a-priori SNR ξ_k and a-posteriori SNR γ_k: $H_k(\xi_k,\gamma_k)$. Figure 4.3 shows the shape of the Ephraim and Malah suppression rule as a function of these two parameters.

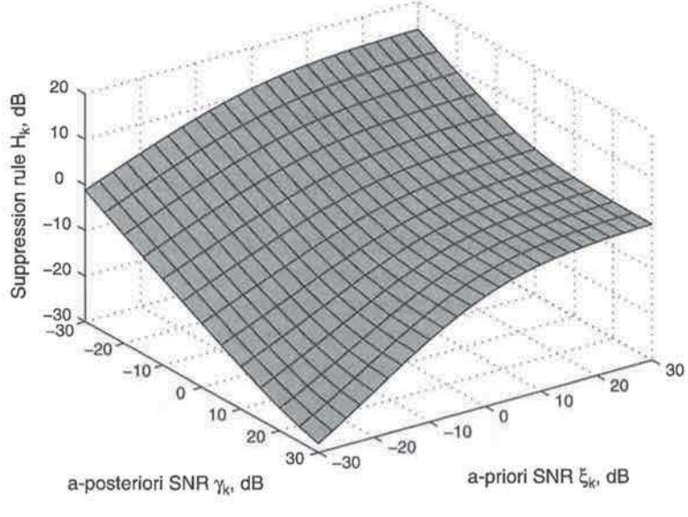


Figure 4.3 Short-term minimum mean-square estimator (MMSE) suppression rule as a function of a-priori and a-posteriori SNRs

4.2.8 Ephraim and Malah Short-term Log-MMSE Suppression Rule

Considering the fact that humans hear in a logarithmic scale of sound pressure level (i.e., magnitudes), Ephraim and Malah [6] derived another suppression rule that is optimal in the MMSE log-spectral amplitude sense. The suppression rule is simpler than the MMSE spectral amplitude estimator:

$$H_k = \frac{\xi_k}{1+\xi_k}\left\{\frac{1}{2}\int_{\nu_k}^{\infty}\frac{\exp(-t)}{t}dt\right\} \tag{4.20}$$

but unfortunately it contains an integral that has to be computed in real time. As this is one of the best performing suppression rules, numerous interpolations and approximations for fast computation of this integral have been designed. The integral is a function of one variable and can be easily tabulated and interpolated in real time.

Regardless of the quite different optimization criterion and analytic form, the short-term log-MMSE suppression rule is surprisingly close to the short-term MMSE suppression rule. Figure 4.4 shows the shape of a log-MMSE suppression rule and the difference between this rule and the MMSE suppression rule. The mean of the difference is 1.12 dB, and the maximum difference is only 1.46 dB for $\xi_k \in [-30, +30]$ dB and $\gamma_k \in [-30, +30]$ dB.

4.2.9 More Efficient Solutions

The Wiener filter approach relies on second-order statistics only. Therefore it makes fewer assumptions about the shapes of the probability densities involved. The suppression rules from Ephraim and Malah take explicitly into account the probability density functions of the speech and the noise signals. The MMSE and log-MMSE optimal solutions lead to integrals, exponents, and Bessel functions that are difficult to compute.

Wolfe and Godsill [7] looked for computationally more efficient alternatives. They derived three additional suppression rules, using different criteria for optimality: the 'joint maximum a-posteriori spectral amplitude and phase' (JMAP SAP) estimator, the 'maximum a-posteriori spectral amplitude' (MAP SA) estimator, and the 'minimum mean-square-error spectral power' (MMSE SP) estimator. They assume both speech and noise signals to be Gaussian random processes. The three criteria they use and the corresponding suppression rules are shown in Table 4.1. The table shows the mean and maximum difference between these suppression rules, and Ephraim and Malah's short-term MMSE estimator.

All three rules are much faster to compute in real time as they do not contain Bessel functions and exponents. In the same paper the authors compare these three rules with the short-time MMSE suppression rule, derived by Ephraim and Malah. The average

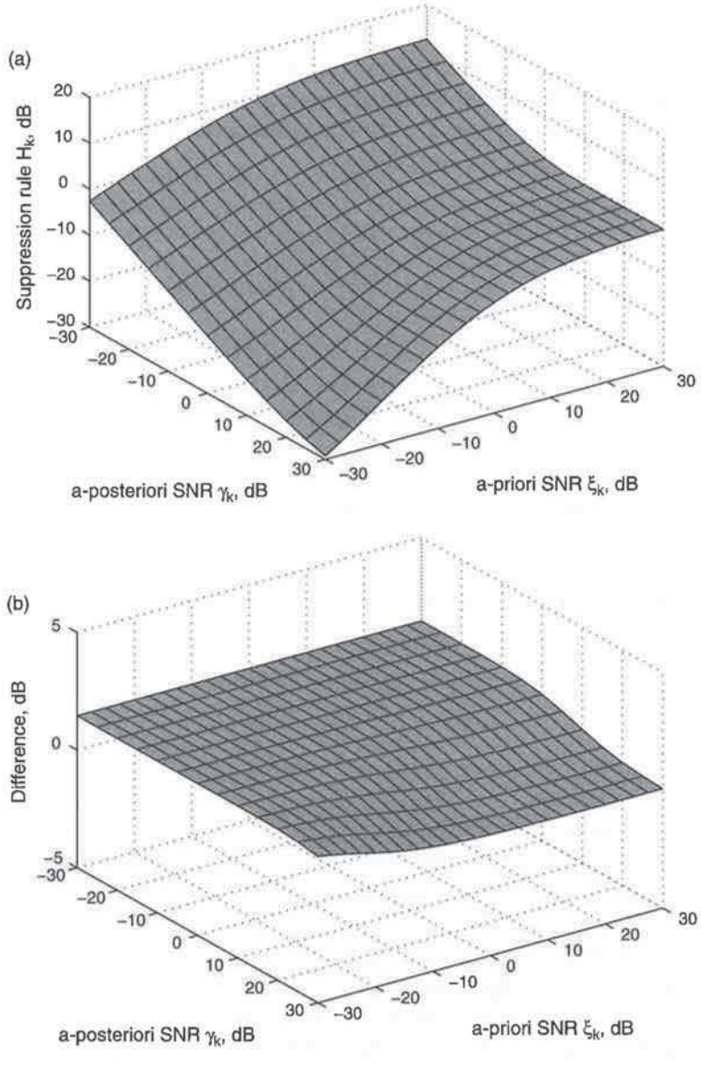


Figure 4.4 Short-term log-MMSE suppression rule: (a) as a function of a-priori and a-posteriori SNRs; (b) difference between MMSE and log-MMSE suppression rules

difference for $(\gamma_k - 1, \xi_k) \in [-30, 30]$ dB is of the order of 1 dB, as Table 4.1 shows. This means that all four rules should have approximately the same noise suppression. Figure 4.5 shows the difference between Ephraim and Malah's suppression rule and the MAP SA estimator – the rule with highest difference.

Table 4.1 More efficient suppression rules

Optimality	Suppression rule	Mean diff (dB)	Max diff (dB)
Joint maximum a-posteriori spectral amplitude and phase (JMAP SAE) estimator	$H_k = \dfrac{\xi_k + \sqrt{\xi_k^2 + 2(1+\xi_k)\frac{\xi_k}{\gamma_k}}}{2(1+\xi_k)}$	0.522	+1.77
Maximum a-posteriori spectral amplitude (MAP SA) estimator	$H_k = \dfrac{\xi_k + \sqrt{\xi_k^2 + (1+\xi_k)\frac{\xi_k}{\gamma_k}}}{2(1+\xi_k)}$	1.261	+4.70
MMSE spectral power estimator	$H_k = \sqrt{\dfrac{\xi_k}{1+\xi_k}\left(\dfrac{1+\nu_k}{\lambda_k}\right)}$	0.685	−1.05

4.2.10 *Exploring Other Probability Distributions of the Speech Signal*

As was noted in Chapter 2, a speech signal does not have a Gaussian probability distribution. With suppression rules taking into account the actual PDF of the speech signal, the next logical step is to derive suppression rules with better probabilistic models of the speech signal. Martin [8] derives three new suppression rules, under the assumption of Gaussian noise and Gaussian, gamma, and Laplace PDFs of the speech signal – see Equations 2.1, 2.3, and 2.5, respectively. All three rules are optimal in amplitude MMSE sense. For the first time, here the real and imaginary parts in each frequency bin are estimated separately, which may eventually lead to better estimation of the phase; that is

$$\mathrm{E}\{S|Y\} = \mathrm{E}\{S_\mathrm{R}|Y_\mathrm{R}\} + j\mathrm{E}\{S_\mathrm{I}|Y_\mathrm{I}\} \tag{4.21}$$

where Y_R and Y_I are the real and imaginary parts of the input signal, S_R and S_I are the real and imaginary parts of the estimated speech signal, all in the corresponding frequency bins.

Assumption of Gaussian noise and Gaussian speech PDFs leads directly to the Wiener estimation rule in Equation 4.12.

With Gaussian noise and gamma distribution of the speech signal, the suppression rule is

$$\mathrm{E}\{S_\mathrm{R}|Y_\mathrm{R}\} = \frac{\sqrt[4]{1.5}}{2\pi\sigma_n p(Y_\mathrm{R})} \int_{-\infty}^{+\infty} S_\mathrm{R}|S_\mathrm{R}|^{-0.5} \cdot \exp\left(-\frac{Y_\mathrm{R}^2}{\sigma_n^2} + \frac{2Y_\mathrm{R}S_\mathrm{R}}{\sigma_n^2} - \frac{S_\mathrm{R}^2}{\sigma_n^2} - \frac{\sqrt{3}|S_\mathrm{R}|}{\sqrt{2}\sigma_s}\right) \mathrm{d}S_\mathrm{R}. \tag{4.22}$$

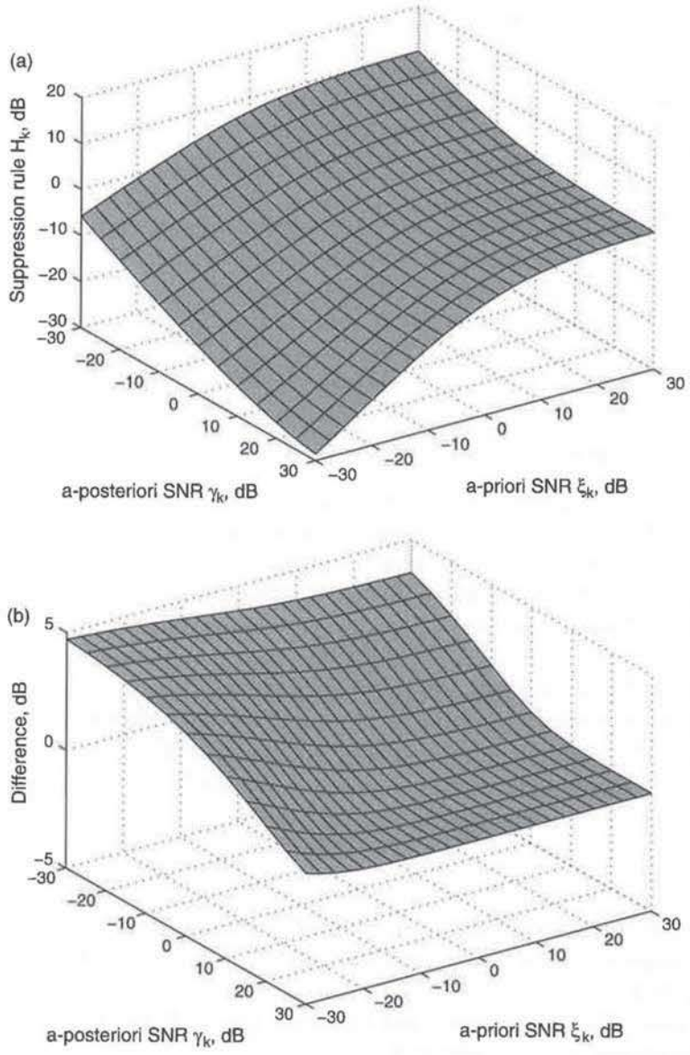


Figure 4.5 Maximum a-posteriori spectral amplitude estimator (MAP SA) suppression rule: (a) as a function of a-priori and a-posteriori SNRs; (b) difference between MMSE and MAP SA suppression rules

Here σ_n and σ_s are the variations of the noise and speech signal. After solving the integral, the suppression rule takes the form

$$\mathrm{E}\{S_\mathrm{R}|Y_\mathrm{R}\} = \frac{\sigma_n}{2\sqrt{2}Z_{GR}}\left\{\exp\left(\frac{G_{\mathrm{R}-}^2}{2}\right)D_{-1.5}(\sqrt{2}G_{\mathrm{R}-}) - \exp\left(\frac{G_{\mathrm{R}+}^2}{2}\right)D_{-1.5}(\sqrt{2}G_{\mathrm{R}+})\right\} \tag{4.23}$$

where $G_{\mathrm{R}+}$, $G_{\mathrm{R}-}$ and Z_{GR} are given by

$$G_{\mathrm{R}+} = \frac{\sqrt{3}\sigma_n}{2\sqrt{2}\sigma_s} + \frac{Y_\mathrm{R}}{\sigma_n} = \frac{\sqrt{3}}{2\sqrt{2}\sqrt{\xi}} + \frac{Y_\mathrm{R}}{\sigma_n} \tag{4.24}$$

$$G_{\mathrm{R}-} = \frac{\sqrt{3}\sigma_n}{2\sqrt{2}\sigma_s} - \frac{Y_\mathrm{R}}{\sigma_n} = \frac{\sqrt{3}}{2\sqrt{2}\sqrt{\xi}} - \frac{Y_\mathrm{R}}{\sigma_n} \tag{4.25}$$

$$Z_{GR} = \exp\left(\frac{G_{\mathrm{R}-}^2}{2}\right)D_{-0.5}(\sqrt{2}G_{\mathrm{R}-}) + \exp\left(\frac{G_{\mathrm{R}+}^2}{2}\right)D_{-0.5}(\sqrt{2}G_{\mathrm{R}+}). \tag{4.26}$$

In the equations above, $D_p(z)$ denotes a parabolic cylinder function. The computational complexity is obvious, the suppression rule includes numerous exponents and parabolic cylinder functions. Note that in real time the estimation of this rule has to be done twice for each frequency bin – once for the real part and once for the imaginary part. Figure 4.6 shows the shape of this suppression rule. While we already had suppression gain values above 0 dB in previous suppression rules, here in high a-priori

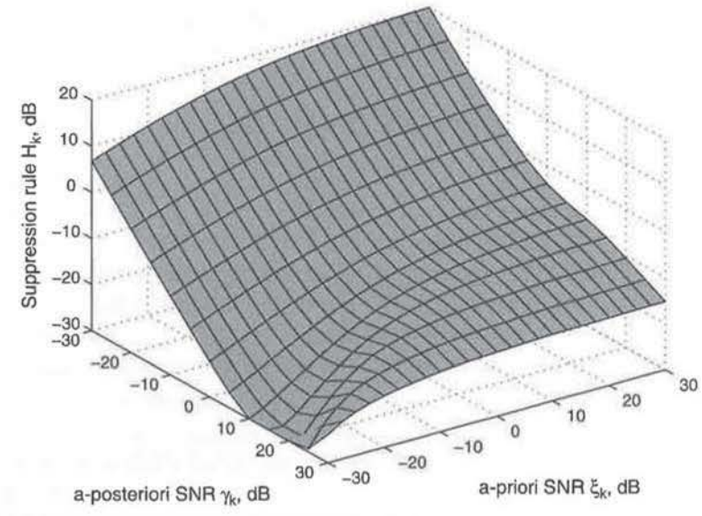


Figure 4.6 Suppression rule under the assumption of Gaussian noise and gamma speech distributions as function of a-priori and a-posteriori SNRs

and low a-posteriori SNRs they reach +20 dB. As we will see later, this can cause some instability of the entire noise suppressor, which is not desirable.

Under the assumptions of Gaussian noise and Laplace PDF, one more suppression rule is derived, which is even more complex than Equation 4.23. The experimental results with a speech signal corrupted with Gaussian and car noise show a slight advantage with the Gaussian/gamma suppression rule – in the range of 0.1–0.4 dB better suppression than the Wiener suppression rule.

4.2.11 Probability-based Suppression Rules

Assume that we have a zero-mean Gaussian noise with magnitude variance λ_d and a speech signal with Gaussian distribution and magnitude variance λ_x. If we have a complex signal with independent and identically distributed real X_R and imaginary X_I parts, both modeled as Gaussian noise $\mathbb{N}(0, \sigma^2)$, then the magnitude of this noise, $|X| = (X_R^2 + X_I^2)^{1/2}$, will have the Rayleigh distribution

$$p(|X||\sigma) = \frac{|X|}{\sigma^2}\exp\left(-\frac{|X|^2}{2\sigma^2}\right) \tag{4.27}$$

where σ is the only parameter. The maximum-likelihood estimator for the parameter σ is $\sigma^2 = \frac{1}{2N}\sum_{i=0}^{N-1}|X_i|^2$ which leads to $\sigma_d = \lambda_d/2$ and $\sigma_x = \lambda_x/2$. Then the noise and speech signals will have the following distributions:

$$\begin{aligned} p_d(|Y|) &= \frac{2|Y|}{\lambda_d}\exp\left(-\frac{|Y|^2}{\lambda_d}\right) \\ p_x(|Y|) &= \frac{2|Y|}{\lambda_x}\exp\left(-\frac{|Y|^2}{\lambda_x}\right). \end{aligned} \tag{4.28}$$

Now assume that we have two hypotheses:

- H_d: the noise signal dominates in this frame and frequency bin;
- H_x: the speech signal dominates in this frame and frequency bin.

Then the probability of the second hypothesis is given by

$$P(H_x||Y|) = \frac{p_x(|Y|)P(H_x)}{p_x(|Y|)P(H_x) + p_d(|Y|)P(H_d)} \tag{4.29}$$

where $P(H_d)$ and $P(H_x)$ are the a-priori probabilities for the corresponding hypotheses, and $p_d(|Y|)$ and $p_x(|Y|)$ are the distributions, defined in Equation 4.28. We will return to this equation later, but for now we just factorize $P_x(|Y|)$. In this case the probability of the speech signal to be dominant for a given magnitude $|Y|$ will be

$$P(H_x||Y|) = P_x(|Y|)\frac{p_x(|Y|)}{p_x(|Y|) + \frac{P_d(|Y|)}{P_x(|Y|)}p_d(|Y|)} \tag{4.30}$$

which after some substitutions can be expressed in terms of a-priori and a-posteriori SNRs:

$$P(H_x||Y|) = P_x(|Y|)\frac{\exp\left(-\frac{\gamma}{\xi}\right)}{\exp\left(-\frac{\gamma}{\xi}\right) + \exp(-\gamma)} = P_x(|Y|)\frac{1}{1 + \exp(-\gamma)/\exp(-\gamma/\xi)}. \tag{4.31}$$

The probability of the speech signal dominating the frequency bin can be used as a suppression rule:

$$H_k = \frac{1}{1 + \dfrac{\exp(-\gamma_k)}{\exp(-\gamma_k/\xi_k)}}. \tag{4.32}$$

We will return later to discuss why the prior probability of signal presence $P_x(|Y|)$ was removed from the suppression rule. The shape of the derived 'most probable amplitude' (MPA) estimator is shown in Figure 4.7a. It is quite different from the suppression rules plotted above. One of the differences is that this suppression rule never goes above 1.0, as it is a probability. This guarantees the stability of the entire noise-reduction system, as we will see later in this chapter. Figure 4.7b shows the suppression rule as a function of the a-posteriori SNR γ (i.e., the magnitude $|Y|$ for given λ_d) for 5 dB and 15 dB a-priori SNRs ξ. The Wiener and maximum-likelihood suppression rules are plotted for comparison. The second obvious difference is that the rule never goes to zero – again because it is a probability. This eliminates the problem with musical noise. The MPA estimator has interesting behavior in the area of very low a-priori SNR, where it actually suppresses the high amplitudes and lets the low amplitudes go unattenuated. In the area of 0 dB a-priori SNR, the suppression rule is constant and equals 0.5 – we cannot separate a mixture of two Gaussians with the same variation, the best we can do from the probabilistic standpoint is to attenuate the magnitude to one-half.

This probabilistic approach can be easily adopted for other than Gaussian PDFs of the speech signal and the noise. There are many studies regarding the speech signal probability distribution, but it is commonly accepted that for short-time audio frames (10–50 ms) a Laplace distribution models the speech signal best [9]. For periods of speech in the range of one to two seconds, the gamma distribution provides better

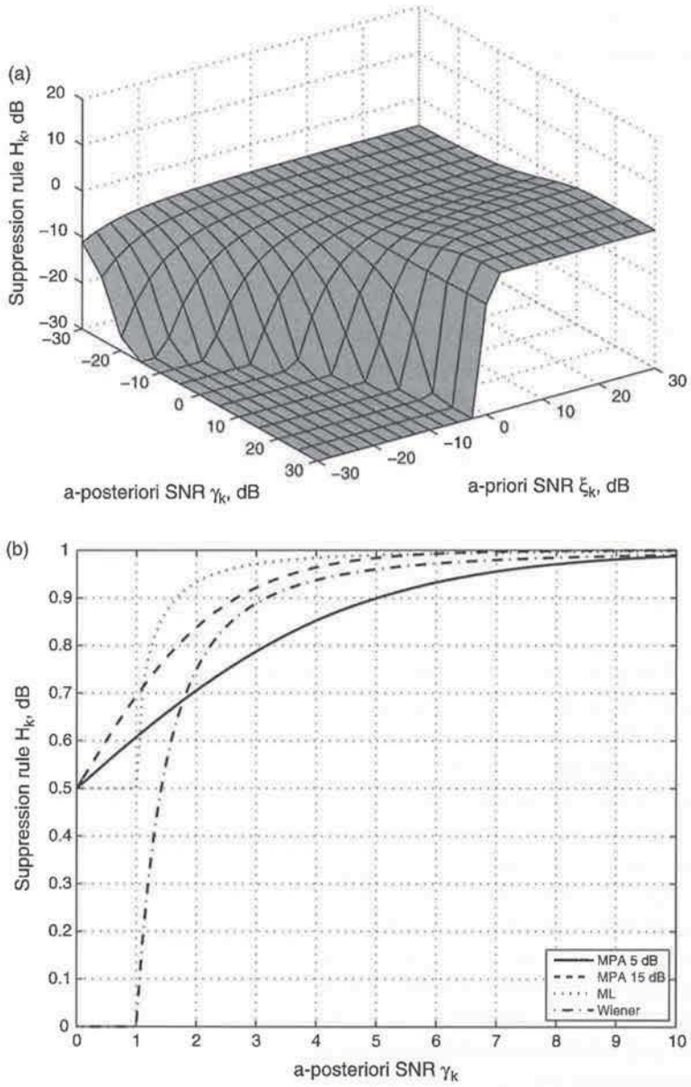


Figure 4.7 MPA estimator suppression rule under the assumption of Gaussian noise and speech distributions: (a) as a function of a-priori and a-posteriori SNRs; (b) comparison between Wiener, ML, and MPA suppression rules as a function of the a-posteriori SNR (MPA is given for 5 and 15 dB a-priori SNRs, respectively)

modeling. Considering the fact that the gamma distribution is more generic (the Laplace distribution is a particular case of the gamma distribution), we will use the gamma distribution. Given Gaussian complex noise with magnitude variance λ_d and Rayleigh distribution of the amplitudes (the same as above), and a speech signal with magnitude variance λ_x and gamma distribution, we have

$$p(x|k,\theta) = \frac{|x|^{k-1}}{\theta^k \Gamma(k)} \exp\left(-\frac{|x|}{\theta}\right) \tag{4.33}$$

where k is the shape and θ is the scale parameter. In our case $k = 1$, and the magnitude of the speech signal is exponentially distributed:

$$p(|x||\theta) = \frac{1}{2\beta} \exp\left(-\frac{|x|}{2\beta}\right). \tag{4.34}$$

The exponential distribution parameter is the magnitude variance $\beta^2 \approx \lambda_x$. Under the same assumption of the two hypotheses above, the suppression rule for gamma speech and Gaussian noise is

$$H_k = \frac{1}{1 + 4\sqrt{\frac{\gamma_k}{\xi_k}} \frac{\exp(-\gamma_k)}{\exp\left(-\frac{1}{2}\sqrt{\frac{\gamma_k}{\xi_k}}\right)}}. \tag{4.35}$$

This suppression rule is plotted in Figure 4.8. Note the similarity in the shape of the rule for low a-priori/high a-posteriori SNRs with Figure 4.6 – the MMSE solution for gamma speech and Gaussian noise – and the similarity in the shape in low a-posteriori/high a-priori SNRs with the shape of the previous probabilistic rule.

4.2.12 Comparison of the Suppression Rules

To compare the effectiveness of the suppression rules alone, an experiment was conducted in a controlled environment. The speech signal, recorded with a close-talk microphone and high SNR, was mixed with noise, recorded in normal office conditions, to generate signals with 0, 10 and 20 dB SNRs. For each experiment, all three signals – clean speech, the noise signal, and the mixture – were available. Audio frames with 512 samples and 50% overlapping frames, weighted with a Han window, were used for conversion to the frequency domain and synthesis back to the time domain. The entire overlap–add process was discussed in detail in Chapter 2, where a MATLAB® script called *ProcessWAV.m* was provided. All three signals were converted to the frequency domain and precise estimations for a-priori SNR ξ_k and a-posteriori SNR γ_k were available for each frame. They were estimated as

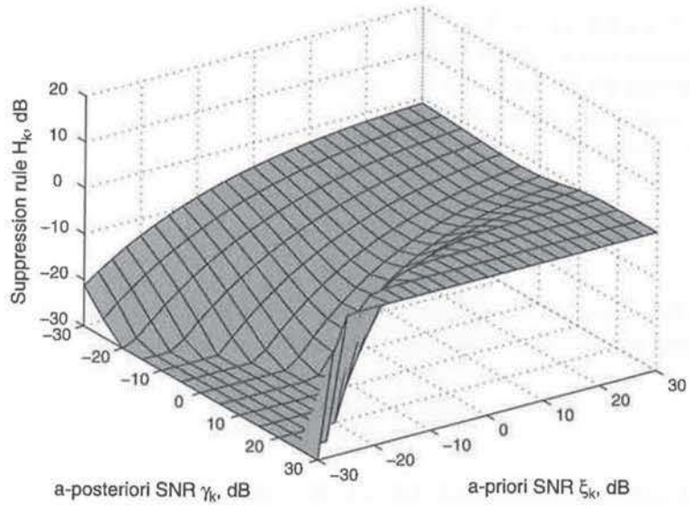


Figure 4.8 MPA estimator suppression rule under the assumption of Gaussian noise and Laplace speech distributions as a function of a-priori and a-posteriori SNRs

$$\begin{aligned}\lambda_d^{(n)}(k) &= (1-\alpha)\lambda_d^{(n)}(k) + \alpha|D_k^{(n)}|^2 \\ \xi_k^{(n)} &\triangleq \frac{|X_k^{(n)}|^2}{\lambda_d(k)} \\ \gamma_k^{(n)} &\triangleq \frac{|Y_k^{(n)}|^2}{\lambda_d^{(n)}(k)}\end{aligned} \tag{4.36}$$

for $\alpha = 0.02$. Here $D_k^{(n)}$, $X_k^{(n)}$, and $Y_k^{(n)}$ are the noise, clean speech, and mixed signals for the k-th frequency bin in the n-th frame. Note that while the noise variance λ_d is averaged, the speech signal variance is taken as $|X|^2$; that is, as momentary variance. Under any circumstances in a real scenario, when we have access only to the mixed signal, we cannot have such a precise estimation of the signal and noise variances and SNRs.

The criteria for comparison were the mean square error (MSE), log-spectral distance (LSD, see Equation 2.47), improvement in the SNR (as difference between the SNR after and before the processing, measured in decibels), and mean opinion score (MOS), measured with the implementation of objective quality measurement algorithm PESQ-W.

The results are shown in Table 4.2. There are sections for each SNR separately and averaged values for each algorithm.

From the MSE perspective, the best performers are Wiener (which is optimal in exactly this sense), closely followed by the entire group of efficient alternatives, and

Table 4.2 Comparison of various suppression rules

		0 dB			10 dB			20 dB			Average		
Algorithm	Equation	MSE	LSD	SNRI	MOS	MSE	LSD	SNRI	MOS	MSE	LSD	SNRI	MOS	MSE	LSD	SNRI	MOS
Do nothing	(4.54)	0.0429	0.885		2.377	0.0136	0.536		3.129	0.0043	0.300		3.906	0.0203	0.574		3.137
Wiener	(4.14)	0.0066	0.458	22.23	3.311	0.0028	0.405	20.27	3.875	0.0012	0.291	13.53	4.217	0.0035	0.384	18.68	3.801
Wiener DDA	(4.12)	0.0052	0.767	39.94	3.462	0.0024	0.475	29.34	4.029	0.0012	0.267	16.97	4.298	0.0029	0.503	28.75	3.930
Maximum-likelihood	(4.17)	0.0229	0.694	5.76	2.728	0.0075	0.421	5.56	3.443	0.0026	0.258	4.67	4.082	0.0110	0.457	5.33	3.418
MMSE	(4.18)	0.0057	0.317	29.39	3.631	0.0026	0.279	23.27	4.032	0.0013	0.211	13.77	4.276	0.0032	0.269	22.14	3.980
Log-MMSE	(4.20)	0.0055	0.332	30.71	3.677	0.0025	0.295	24.44	4.064	0.0013	0.221	14.71	4.288	0.0031	0.283	23.29	4.010
Spectral subtraction	(4.16)	0.0059	0.322	28.50	3.604	0.0027	0.286	22.55	4.001	0.0013	0.223	13.10	4.259	0.0033	0.277	21.38	3.955
JMAP SAE	Table 4.1, 1	0.0056	0.325	30.79	3.658	0.0026	0.280	24.01	4.038	0.0013	0.210	13.92	4.280	0.0031	0.272	22.91	3.992
MAP SAE	Table 4.1, 2	0.0054	0.370	33.17	3.717	0.0025	0.308	25.76	4.075	0.0012	0.220	15.08	4.295	0.0030	0.300	24.67	4.029
MMSE SAE	Table 4.1, 3	0.0058	0.314	28.44	3.592	0.0027	0.274	22.39	4.003	0.0013	0.208	13.05	4.265	0.0033	0.265	21.29	3.953
Prob. Gauss-Gauss	(4.32)	0.0055	1.187	38.80	3.008	0.0025	0.746	29.79	3.692	0.0012	0.333	17.00	4.214	0.0031	0.755	28.53	3.638
Prob. Laplace-Gauss	(4.35)	0.0055	1.233	40.04	2.958	0.0025	0.822	30.10	3.623	0.0012	0.379	17.61	4.180	0.0031	0.811	29.25	3.587

probabilistic rules. The maximum-likelihood suppression rule is definitely worst in this sense.

From the LSD perspective, the front runners are MMSE and log-MMSE (which is optimal in the log-MMSE sense). Good results are shown by the entire group of efficient alternatives. Note that Wiener and probabilistic rules are worse from this perspective, which means that they do not deal well with low levels of noise and speech.

The best average SNR improvement definitely has Wiener and probabilistic rules, followed by the efficient alternatives and spectral subtraction. The maximum-likelihood rule, as expected, has the lowest improvement in SNR. It is outperformed by the approximate Wiener suppression rule.

The highest MOS score and the best sound is achieved by log-MMSE and MAP SAE, followed closely by the group of efficient alternatives. The maximum-likelihood suppression rule sounds worse owing to a substantial amount of noise.

Figure 4.9 shows the relationship between the average improvement of SNR and the average MOS score – the last two columns in Table 4.2. It is clear that, to a certain degree, the noise suppression helps, and the signals with more suppressed noise achieve better perceptual sound quality. Enforcing the noise suppression further actually decreases the sound quality, regardless of the better SNR. This is good evidence that, when evaluating noise-suppressing algorithms, improvement in the SNR should not be used as the only criterion, and even not as a main evaluation criterion. Ultimately the goal of this type of speech enhancement is to make the output signal sound better for the human listener. From this perspective, the MOS is a much better criterion. When targeting speech recognition, the best criterion is, of course, the highest recognition rate.

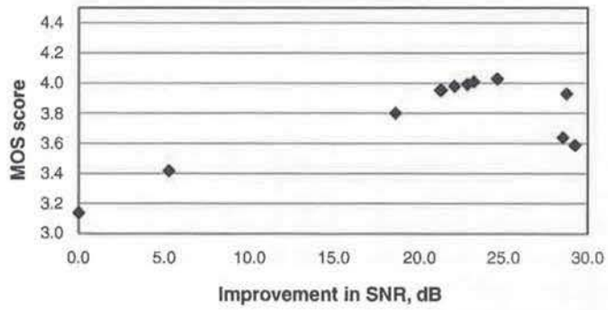


Figure 4.9 MOS results as a function of the SNR improvement for various suppression rules

On comparing the overall performance of the suppression rules, spectral subtraction and the entire group of efficient alternatives of Ephraim and Malah's MMSE (and that very same rule, of course) definitely stand out. To reiterate, this evaluation has been done under the best possible conditions and precise SNR estimates. In real conditions, the parameters for computation of the suppression rules will be estimated and have

certain errors. Thus it will be important how robust each one of these suppression rules is to those errors.

EXERCISE

Look at the MATLAB script *SuppressionRule.m* which returns the suppression rule values for the given vectors of a-priori and a-posteriori SNRs:

```
Gain = SuppressionRule(gamma, xi, SuppressionType)
```

The argument `SuppressionType` is a number from 0 to 9 and determines which suppression rule is to be used. The script contains implementation of most of the suppression rules discussed so far. Finish the implementation of the rest of the suppression rules.

Write a MATLAB script that computes the suppression rules as a function of the a-priori and a-posteriori SNRs in the range of ± 30 dB. Limit the gain values in the range from -40 dB to $+20$ dB and plot the rules in three dimensions using the `mesh` function.

4.3 Uncertain Presence of the Speech Signal

All the suppression rules discussed above were derived under the assumption of the presence of both noise and speech signals. The speech signal, however, is not always presented in the short-term spectral representations. Even continuous speech has pauses with durations of 100–200 ms – which, compared with the typical frame sizes of 10–40 ms, means that there will be a substantial number of audio frames without a speech signal at all. Trying to estimate the speech signal in these frames leads to distortions and musical noise.

Classification of audio frames into "noise only" and "contains some speech" is in general a detection and estimation problem [10]. Stable and reliable work of the *voice activity detector* (VAD) is critical for achieving good noise-suppression results. Frame classification is used further to build statistical models of the noise and speech signals, so it leads to modification of the suppression rule as well.

4.3.1 Voice Activity Detectors

Voice activity detectors are algorithms for detecting the presence of speech in a mixed signal consisting of speech plus noise. They can vary from a simple binary decision (yes/no) for the entire audio frame to precise estimators of the speech presence probability for each frequency bin. Most modern noise-suppression systems contain at least one VAD, in many cases two or more. The commonly used algorithms base their decision on the assumption of a quasi-stationary noise;

that is, the noise variance changes more slowly than the variance of the speech signal. This allows one to build a model of the noise and track the changes. Then the decision is based on the assumption that the presence of a speech signal means an increase in energy – for the entire frame and per frequency bin. Such VADs work reliably for SNRs down to 0 dB. There is a separate group of approaches for detecting the presence of the speech signal in very low SNR conditions, or when the noise is highly non-stationary and changes as fast as, or faster than, the energy envelope of the speech signal.

One of the most commonly used and cited VADs is described in [11]. The purpose of this VAD is to detect the silent periods and improve the work of the G.729 codec. It is frequently used as a baseline to compare the performance of improved algorithms for VADs.

4.3.1.1 ROC Curves

Simple VADs produce a binary decision for the presence or absence of a speech signal in the current audio frame. In signal detection theory, such binary classifiers are characterized by the so-called *receiver operating characteristic* (ROC), or simply ROC curve. This is a graphical plot of probabilities of true positives versus false positives. In general, the binary classifier makes a decision that can be interpreted in four ways:

- true positive TP – a correct decision for presence was made = hit;
- true negative TN – a correct decision for absence was made = correct rejection;
- false positive FP – a decision for presence was made when a speech signal is not present = false alarm;
- false negative FN – a decision for absence was made when a speech signal was present = miss.

If we have a total number of positives N_P and a total number of negatives N_N, the true positive rate P_{TP}, or sensitivity, or recall, is defined as the proportion of true positives and all positives:

$$P_{TP} = \frac{N_{TP}}{N_P} = \frac{N_{TP}}{N_{TP} + N_{FN}} \tag{4.37}$$

and the false positive rate P_{FP}, or false-alarm rate, is defined as the proportion of false positives and all negatives:

$$P_{FP} = \frac{N_{FP}}{N_N} = \frac{N_{FP}}{N_{FP} + N_{TN}}. \tag{4.38}$$

Another important parameter of this type of detectors is the accuracy P_A, defined as the proportion of the correct decisions (i.e., true positives and true negatives) to all decisions:

$$P_A = \frac{N_{TP} + N_{TN}}{N_P + N_N}. \tag{4.39}$$

Energy-based binary classification (or VAD) for the absence/presence of speech signals can operate using a certain threshold. That is, if the energy of the current frame is greater than the threshold, then we have a speech signal, otherwise the speech signal is not present. The optimal value of the threshold can be estimated using the ROC curves. A typical ROC curve in this case will look like the one in Figure 4.10. For each threshold there is a corresponding point on the chart. For higher thresholds we have fewer false positives, but more false negatives. With a lower threshold the detection rate of the speech signal will be higher, but with the price of more noise frames detected as speech. The diagonal line shows the random-decision approach. The ROC of our classifier should be above this line (otherwise we can just negate the decisions made by the classifier). The threshold, corresponding to the point closest to the upper left corner, minimizes the sum of the squares of false positives and false negatives, $P_{FN}^2 + P_{FP}^2$. It is

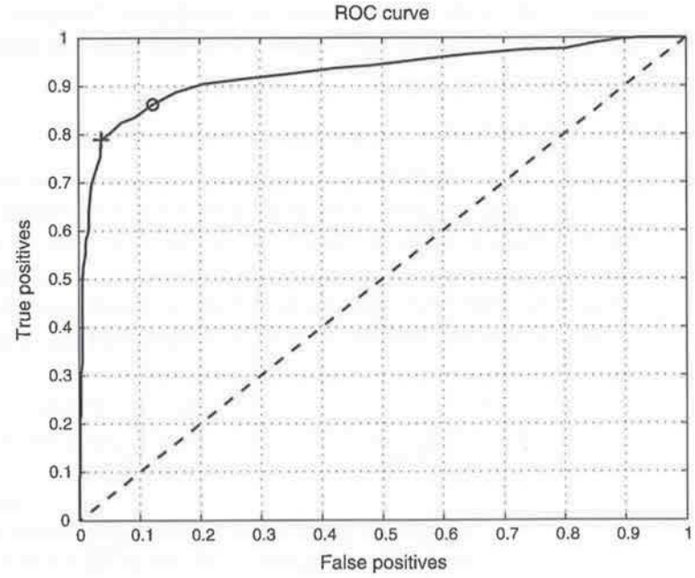


Figure 4.10 Receiver operating characteristics (ROC) curves for various thresholds of energy-based voice activity detectors. A "+" marks the highest accuracy point, an "o" marks the closest to the upper left corner point, where the sum of the squares of false positives and false negatives is minimal

marked with a small circle. The point where the accuracy from Equation 4.39 is highest is marked with a plus sign. Under some special circumstances the optimal point can be different. For estimation of the noise statistical parameters, for example, we may want to specify some true positive rate at the price of reducing the false positive rate, when energy from the speech signal will be averaged as noise statistical parameters.

ROC analysis was first used in World War II for the analysis of radar signals before being employed in signal detection theory. Currently it is widely used in biology and medicine. ROC curves find application in some machine-learning techniques as well.

EXERCISE

Write a MATLAB script to mix a clean speech signal (record or use *Speech.WAV*) with noise (record or use *NoiseHoth.WAV*) to achieve ~5 dB SNR. Write the output into *NoisySpeech.WAV*.

Create a text file containing the beginnings and ends of each speech segment – this is going to be used as a ground truth.

Write a MATLAB script to build the ROC characteristics of a fixed-threshold VAD. The script reads the *NoisySpeech.WAV* and the text file with the ground truth. Then use a set of thresholds for classifying the frames to speech (RMS of the frame above the threshold) or noise (RMS of the frame below the threshold). Compute the false positives and false negatives. Plot the ROC curve – it should look like Figure 4.10.

4.3.1.2 Simple VAD with Dual-time-constant Integrator

Using a fixed threshold in binary VAD limits the system to a certain level of noise in the signal. In real VAD systems, the noise floor is usually adaptively tracked and the decision for absence/presence is made based on the average noise floor level. Assuming that the noise floor changes more slowly than the speech envelope, we can track the noise floor level with two different time constants: one low when the current level is higher than the estimate, and one high when the level is lower than the estimate:

$$L_{\min}^{(n)} = \begin{cases} \left(1-\dfrac{T}{\tau_{\text{up}}}\right)L_{\min}^{(n-1)} + \dfrac{T}{\tau_{\text{up}}}L^{(n)} & L^{(n)} > L_{\min}^{(n-1)} \\ \left(1-\dfrac{T}{\tau_{\text{down}}}\right)L_{\min}^{(n-1)} + \dfrac{T}{\tau_{\text{down}}}L^{(n)} & L^{(n)} \le L_{\min}^{(n-1)}. \end{cases} \tag{4.40}$$

Here $L_{\min}^{(n)}$ is the estimate of the noise floor for the n-th frame, $L^{(n)}$ is the estimated signal level for the same frame, T is the frame duration, and τ_{down} and τ_{up} are the time

constants for tracking the noise floor level when the level goes lower or higher than the current estimate. As the speech-plus-noise signal has a higher level than just the noise signal, $\tau_{\text{down}} \ll \tau_{\text{up}}$. For estimation of the signal level, weighted RMS is usually used:

$$L^{(n)} = \sqrt{\frac{1}{K}\sum_{k=0}^{K-1}(W_k.|Y_k^{(n)}|)^2}. \tag{4.41}$$

Here **W** is a weighting function. The idea is to give more weight to the frequency bins with a higher difference between the signal and the noise (i.e., with higher SNR), which will allow easier differentiation. As the noise energy is usually concentrated in the lower part of the frequency band, this can just be a high-pass filter. Considering the fact that the speech energy decreases in the higher parts of the frequency band, it is a good idea to suppress the higher frequencies as well, in case of unexpected noises there. A standard weighting is often used – C-message or ITU-T Recommendation O.41 for modeling the telephone channel bandwidth. These weightings increase the signal SNR, which makes the detection easier.

The decision for presence/absence of the speech signal can be made based on the estimated noise floor $L_{\min}^{(n)}$, the signal level $L^{(n)}$, and the previous value of the voice activity flag V ($V=0$ for noise, $V=1$ for speech):

$$V^{(n)} = \begin{cases} 0 & \text{if} \quad \dfrac{L^{(n)}}{L_{\min}^{(n)}} < T_{\text{down}} \\ 1 & \text{if} \quad \dfrac{L^{(n)}}{L_{\min}^{(n)}} > T_{\text{up}} \\ V^{(n-1)} & \text{otherwise.} \end{cases} \tag{4.42}$$

Note that the flag V switches unconditionally to state "noise" if the proportion of the current and minimal level is below the threshold T_{down}. To switch to "speech" state, this proportion should go above the threshold T_{up} ($T_{\text{down}} < T_{\text{up}}$). To switch back, the proportion should fall below T_{down}, and so on. This hysteresis stabilizes the work of the VAD, decreases the false positives, and the VAD switches back to its "noise" state after the end of the word. The downside is that the VAD switches to "speech" state with a small delay and can cut off the beginning of the word if it starts with a low-level consonant.

This simple, energy-based VAD with binary decision is a software implementation of the well-known hardware VAD used in a countless number of amateur radio stations – see Figure 4.11. The first two RC groups, R_1C_1 and R_2C_2, form the band-pass filter. The low cut-off frequency is $f_{\text{low}} = 1/(2\pi C_1 R_2)$, the high is $f_{\text{high}} = 1/(2\pi R_1 C_2)$. The diode D_1, the resistor R_3, and the integration capacitor C_3 estimate the envelope

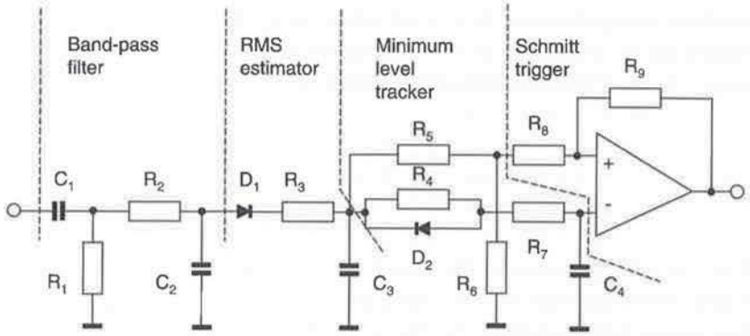


Figure 4.11 Hardware implementation of a binary energy-based voice activity detector

RMS – note that the non-linear VA characteristic of the diode is used as an interpolation of the squaring in Equation 4.41, but the square-root is missing in this analog RMS estimator. The group R_4, D_2, R_7, and C_4 is the minimum level tracker from Equation 4.40. Here $\tau_{up} = (R_4 + R_7)C_4$ and $\tau_{down} = R_7C_4$. Resistors R_5 and R_6 form the absolute threshold, and the operational amplifier, together with R_8 and R_9, forms a Schmitt trigger with a given hysteresis. The Schmitt trigger output is the VAD state.

This VAD has four parameters to adjust, $S = [\tau_{down}, \tau_{up}, T_{down}, T_{up}]$. Typical values used in analog circuitry are $\tau_{down} = 40$ ms, $\tau_{up} = 10$ s, $T_{down} = 1.2$, and $T_{up} = 3$. The performance of this VAD can be improved by maximizing the accuracy (Equation 4.39):

$$S_{opt} = \arg\max_{S}(P_A(S)) \tag{4.43}$$

which minimizes the sum of false positives and false negatives, regardless of their proportions. The point with maximal accuracy is marked with "+" on the ROC curve in Figure 4.10. If we assume that true positives and true negatives are equally important, we can minimize the minimal distance between the ROC curve and the upper left corner, where the ideal classifier is – zero false positives and all true positives. This optimization criterion is given by

$$S_{opt} = \arg\min_{S}\left(\min\left(\sqrt{[1-P_{TP}(S)]^2 + P_{FP}(S)^2}\right)\right). \tag{4.44}$$

The optimal from this perspective point is marked with "o" on the ROC curve in Figure 4.10. The accuracy and the corresponding probabilities can be estimated with a large set of manually labeled WAV files with SNR varying in the work range of this

VAD. An easier approach is to use the VAD and clean speech signal to label the frames with the presence and absence of a speech signal, and then to contaminate the clean speech signal with variable amounts of noise. Almost any optimization method can find the solutions above; gradient-based methods will most probably be the most efficient.

Regardless of its simplicity, the energy-based binary decision VAD works surprisingly well. It easily achieves accuracy above 95% for SNRs varying from 5 to 30 dB. A major contributor to this is the weighting in Equation 4.41, which removes most of the noise energy under normal conditions. This weighting is equivalent to band-pass filtering for telephone bandwidth. One of the advantages of this classifier is that it does not need prior knowledge of the noise and speech statistical parameters – the estimated $L_{min}^{(n)}$ is nothing more than tracking the noise floor. This VAD does not provide soft probability of the speech signal presence in the current frame and per frequency bin.

EXERCISE

Examine the MATLAB script *SimpleVAD.m*. This is an implementation of the voice activity detector above. The function needs the current level, the current time, and a data structure as input parameters, carrying the pre-initialized time constants, thresholds, and so on as described in the initial comment. This data structure has to be initialized at the beginning with the recommended values.

Write a MATLAB script to compute the SNR of the given WAV file, which takes as parameters the file name and returns the signal-to-noise-ratio:

```
SNR = SNRMeasurement(inpFileName)
```

Read the WAV file, initialize the necessary variables (VAD data structure, signal and noise levels, and signal and noise frame counters), organize a loop for processing the file on frames with duration of 20 ms, and compute the RMS level of the current frame. Then use the simple VAD above to do the classification of speech or noise, and add the square of the computed level to the corresponding variable. Increment the corresponding frame counter. When you reach the end of the file, compute the average levels of the signal and noise frames, convert to decibels, and subtract. Display the result.

Modify the MATLAB script above to work in the frequency domain. Use *Process WAV.m* as a template. Pre-compute the weightings for A-, B-, and C-weightings using the approximation formulas from Chapter 3. Organize three sets of noise and speech levels – one for each weighting. At the end, compute and display three SNRs – one for each weighting. For the level provided to the VAD, use either C-message weighting or a rectangular band-pass filter (300–3400 Hz).

Test the final version of the SNR measurement tool and make sure it works well, as this will be a frequently used tool later in the book.

4.3.1.3 Statistical-model-based VAD with Likelihood Ratio Test

Sohn *et al.* [12] use the likelihood ratio test and an effective hangover scheme for classification of the audio frames. Given a speech signal degraded by uncorrelated additive noise, two hypotheses can be considered:

- H_0: speech absent, $\mathbf{Y} = \mathbf{D}$;
- H_1: speech present, $\mathbf{Y} = \mathbf{D} + \mathbf{X}$;

where $\mathbf{D}$, $\mathbf{X}$, and $\mathbf{Y}$ are the K-dimensional complex vectors of the noise, speech and noisy speech. Assuming Gaussian distribution and known variances λ_d and λ_x for the real and imaginary parts of the noise and speech signals, the probability density functions of these two hypotheses are

$$\begin{aligned} p(\mathbf{Y}|H_0) &= \prod_{k=0}^{K-1} \frac{1}{\pi\lambda_d(k)} \exp\left\{-\frac{|Y_k|^2}{\lambda_d(k)}\right\} \\ p(\mathbf{Y}|H_1) &= \prod_{k=0}^{K-1} \frac{1}{\pi[\lambda_d(k)+\lambda_x(k)]} \exp\left\{-\frac{|Y_k|^2}{\lambda_d(k)+\lambda_x(k)}\right\} \end{aligned} \tag{4.45}$$

This yields the likelihood ratio

$$\Lambda_k \triangleq \frac{p(\mathbf{Y}|H_1)}{p(\mathbf{Y}|H_0)} = \frac{1}{1+\xi_k} \exp\left\{\frac{\gamma_k \xi_k}{1+\xi_k}\right\} \tag{4.46}$$

where ξ_k and γ_k are the a-priori and a-posteriori SNRs. Then a decision rule is established by comparing the geometric mean of the likelihood ratios for all frequency bins:

$$\log \Lambda = \frac{1}{L} \sum_{k=0}^{K-1} \log \Lambda_k \underset{H_0}{\overset{H_1}{\gtrless}} \eta, \tag{4.47}$$

where η is a fixed threshold for deciding which one of the two hypotheses is true.

To account for the timing of the speech signal, Sohn and co-authors use an HMM-based hang-over scheme. It is based on the idea that there is a strong correlation in the consecutive occurrences of speech frames. The sequence is modeled with a first-order Markov process – that is, assuming that the current state depends only on the previous states. The correlative characteristics of speech occurrence can be represented by $P(q_n = H_1 | q_{n-1} = H_1)$ with the following constraint:

$$P(q_n = H_1 | q_{n-1} = H_1) > P(q_n = H_1). \tag{4.48}$$

Here q_n denotes the state of the n-th frame and is either H_0 or H_1. With the assumption that the Markov process is time-invariant, then $a_{ij} \triangleq P(q_n = H_j | q_n = H_i)$; and under the assumption of process staionarity, $P(q_n = H_i) = P(H_i)$, where $P(H_0)$ and $P(H_1)$ are the steady-state probabilities, obtained from $a_{01}P(H_0) = a_{10}P(H_1)$ and $P(H_0) + P(H_1) = 1$. Then the overall process is described with only two parameters, a_{01} and a_{10}. The decision rule is modified as

$$\widehat{\Lambda}_n \triangleq \frac{p(\aleph_n | q_n = H_1)}{p(\aleph_n | q_n = H_0)} = \frac{P(H_0)}{P(H_1)} \cdot \frac{P(q_n = H_1 | \aleph_n)}{P(q_n = H_0 | \aleph_n)} \underset{H_0}{\overset{H_1}{\gtrless}} \eta \tag{4.49}$$

where $\aleph_n = \{X_n, X_{n-1}, \ldots, X_1\}$ is the set of observations up to the current frame n. The left fraction in Equation 4.49 is the a-priori probability ratio; the right is the a-posteriori probability ratio. Denoting the second one with Γ_n and using the forward procedure described by Sohn and co-authors, we can obtain the following recursive formula:

$$\Gamma_n = \frac{a_{01} + a_{11}\Gamma_{n-1}}{a_{00} + a_{10}\Gamma_{n-1}} \Lambda_n \tag{4.50}$$

where Λ_n is computed using Equation 4.47. Then the modified decision rule with hangover scheme is compared to the threshold η:

$$\widehat{\Lambda}_n = \frac{P(H_0)}{P(H_1)} . \Gamma_n \underset{H_0}{\overset{H_1}{\gtrless}} \eta. \tag{4.51}$$

In their practical implementation, the authors used hand-labeled voice data to estimate the probabilities. In their case, $a_{01} = 0.2$ and $a_{10} = 0.1$. With hand-labeled data it is relatively easy to draw the ROC curves for various η and to pick the value with highest accuracy. Taking a more detailed look at Equation 4.50, we find that this is just a more sophisticated time smoothing, replacing the traditional first-order integrator.

4.3.1.4 VAD with Floating Threshold and Hangover Scheme with State Machine

One potential limiting factor of the previous VAD is the fixed threshold. This reduces the range of various SNRs the VAD can operate without significant decrease of the classification accuracy. Theoretically, for each SNR there is an optimal threshold η that can work satisfactorily in a certain range of SNRs around the one it is optimal for. Davis and Nordholm [13] derive a way to estimate the optimal threshold for a given SNR. They compute the log-likelihood ratio using a similar approach as in the previous

section. Under the assumption of Gaussian distributions of both noise and speech, the optimal threshold for the acceptable probability for a false alarm P_{FA} is

$$\eta_{\mathrm{opt}}(k) = \sqrt{2}\lambda_d(k).\mathrm{erfc}^{-1}(2P_{\mathrm{FA}}) \tag{4.52}$$

where $\mathrm{erfc}(u)$ is the complimentary error function [14]. Then the modified decision rule with floating threshold for classification of the current frame is

$$\frac{1}{L}\sum_{k=0}^{K-1}\log\Lambda_k \underset{\mathrm{H_0}}{\overset{\mathrm{H_1}}{\gtrless}} \mathrm{erfc}^{-1}(2P_{\mathrm{FA}}).\sqrt{2}.\frac{1}{L}\sum_{k=0}^{K-1}\lambda_d(k). \tag{4.53}$$

In the same paper the authors discuss a slightly different hangover scheme. It is a state machine, which requires the decision rule of Equation 4.53 to be in the "speech" state for at least four consequent frames before switching to "speech" state, and the decision rule to be in the "noise" state for ten consequent frames before switching to "noise" state. This state machine effectively delays the decision to gain confidence. The number of frames to delay the decision can be tuned for specific conditions, but in general the first delay reduces the false alarms, and the second delay allows covering the end of the word, which usually has low energy.

4.3.2 Modified Suppression Rule

To make the noise suppressors more efficient, a modified suppression rule was derived by McAulay and Malpass [4]. While comparing the ML suppression rule, derived by them in the same paper, with the Wiener and spectral subtraction rules existing at that time, they concluded that it was apparent that none of the suppression rules adequately suppresses the background noise when the speech is absent, as all of them are derived under the assumption of speech presence. The presence or absence of the speech signal is considered a two-state model:

- $\mathrm{H_0}$: speech absent, $|Y_k^{(n)}| = |D_k^{(n)}|$;
- $\mathrm{H_1}$: speech present, $|Y_k^{(n)}| = |D_k^{(n)} + X_k^{(n)}|$.

Under their assumption of Gaussian noise and the speech signal model as a sum of sinusoidal signals, the modified suppression rule is derived as

$$\tilde{H}_k^{(n)} = P(\mathrm{H_1}||Y_k^{(n)}|).H_k^{(n)} \tag{4.54}$$

where $\tilde{H}_k^{(n)}$ is the modified suppression rule, $H_k^{(n)}$ is the estimated suppression rule (in their case, the ML amplitude estimator), and $P(\mathrm{H_1}||Y_k^{(n)}|)$ is the probability to have a

speech signal present in this frequency bin and frame. The derivation of Equation 4.54 is generic and does not depend on the specific assumptions for the distributions of the speech and noise signals.

Ephraim and Malah [3] used results from [15] to modify the suppression rule under the uncertain presence of a speech signal as

$$\tilde{H}_k^{(n)} = \frac{\Lambda(Y_k, q_k)}{1+\Lambda(Y_k, q_k)}.H_k^{(n)} \tag{4.55}$$

where $\Lambda(Y_k, q_k)$ is the generalized likelihood ratio defined by

$$\Lambda(Y_k, q_k) = \mu_k \frac{P(Y_k|\mathrm{H_1})}{P(Y_k|\mathrm{H_0})} \tag{4.56}$$

$\mu_k \triangleq (1-q_k)/q_k$ and q_k is the probability of signal absence in the k-th frequency bin. The two hypotheses $\mathrm{H_0}$ and $\mathrm{H_1}$ for the current frequency bin are the same as above – the absence and presence of a speech signal. For Gaussian speech and Gaussian noise, $\Lambda(Y_k, q_k)$ can be expressed in terms of a-priori and a-posteriori SNRs:

$$\Lambda(Y_k, q_k) = \mu_k \frac{\exp(\nu_k)}{1+\xi_k}, \tag{4.57}$$

where now ξ_k is defined as the a-priori SNR when speech is present:

$$\xi_k \triangleq \frac{\mathrm{E}\{|X_k|^2|\mathrm{H}_k\}}{\lambda_d(k)}. \tag{4.58}$$

For convenience and easier estimation, we define

$$\eta_k \triangleq \frac{\mathrm{E}\{|X_k|^2\}}{\lambda_d(k)} = (1-q_k)\xi_k, \tag{4.59}$$

which is easier to estimate, and this finalizes the modification of the suppression rule under the uncertain presence of a speech signal. Looking closely at the modifier of the suppression rule in Equation 4.55, we can say that it is actually a generalized probability for the presence of a speech signal, which concurs with the conclusion in Equation 4.54. Note the similarity of Equations 4.46 and 4.57 – they both are actually speech-presence probability estimators.

The modified suppression rule is derived under certain assumptions for the short-term speech and noise probability distributions. As already mentioned, the short-term speech distribution is modeled best with a Laplace distribution, while a Gaussian distribution best models the noise in the majority of cases. On the other

hand, the gamma distribution models longer intervals of speech signals (1–2 seconds); see [9].

4.3.3 Presence Probability Estimators

An alternative approach to introduce the uncertain presence of the speech signal is presented in Equation 4.29 with the a-priori probabilities for a dominant presence of speech $P(\mathrm{H}_d)$ or noise $P(\mathrm{H}_x)$, respectively. The noise and speech probability for the entire frame can be used as a prior for each frequency bin. Using the likelihood, estimated from Equation 4.47 or 4.51, we can estimate the probability for speech presence or absence in the current audio frame:

$$\begin{aligned} P^{(n)}(\mathrm{H}_1) &= \frac{\Lambda^{(n)}}{1+\Lambda^{(n)}} \\ P^{(n)}(\mathrm{H}_0) &= 1-P^{(n)}(\mathrm{H}_1). \end{aligned} \tag{4.60}$$

The level $L^{(n)}$, or RMS estimated according to Equation 4.41, has a Rayleigh distribution during the noise frames. The distribution of the levels during speech frames can be assumed to be either Rayleigh (assuming Gaussian distribution of the speech signal) or exponential (assuming Laplace distribution of the speech signal). These assumptions and the hypotheses for a predominant noise or speech signal lead to the probability estimators in Equation 4.31 or 4.35, but this time for the entire audio frame.

Estimation of the speech-presence probability $P_k^{(n)}(\mathrm{H}_1||Y_k|)$ for each frequency bin can be done using either of the two methods described above for estimation of the speech-presence probability per audio frame.

4.4 Estimation of the Signal and Noise Parameters

4.4.1 Noise Models: Updating and Statistical Parameters

Estimation of the noise variation λ_d, given the VAD state $V^{(n)}$ for the n-th frame can be done using a simple recursive formula:

$$\lambda_d^{(n)}(k) = \begin{cases} (1-\alpha)\lambda_d^{(n-1)}(k) + \alpha|Y_k^{(n)}|^2 & \text{for} \quad V^{(n)}=0 \\ \lambda_d^{(n-1)}(k) & \text{for} \quad V^{(n)}=1 \end{cases} \tag{4.61}$$

where α is the adaptation parameter with typical values between 0.5 and 0.95. This adaptation formula works well in real systems. One thing we may want to fix is that at the end of the word the noise estimation is outdated if the noise variance changed meantime. In addition, speech is quite a sparse signal in the

frequency domain. Even with speech present, a lot of frequency bins contain just noise and we can update the noise variance for them even when the speech signal is present in the frame. This leads to the probability-based noise variance estimator

$$\lambda_d^{(n)}(k) = P_k^{(n)}(\mathrm{H}_1||Y_k|)\lambda_d^{(n-1)}(k) + P_k^{(n)}(\mathrm{H}_0||Y_k|)|Y_k|^2; \tag{4.62}$$

that is, we use the speech absence and presence probabilities for the frequency bin to modify the adaptation parameter. A small problem here is that, when $P_k^{(n)}(\mathrm{H}_1||Y_k|)$ approaches one, which is completely possible, then we will have an estimate that is actually the momentary value of the noise variation. In most cases this is not desirable, as λ_k is a statistical parameter. This leads to combining the two estimators:

$$\lambda_d^{(n)}(k) = (1-\alpha)P_k^{(n)}(\mathrm{H}_1||Y_k|)\lambda_d^{(n-1)}(k) + \alpha P_k^{(n)}(\mathrm{H}_0||Y_k|)|Y_k|^2 \tag{4.63}$$

which is nothing more than limiting the adaptation speed to the value of the adaptation parameter α in cases of high probability of noise-only presence.

Another interesting approach for estimation of the noise variance is based on minimum statistics [16]. It is based on the estimation of floating and optimal values of the smoothing parameter α. During the speech-absence frames we want our noise variance estimate $\hat{\lambda}_d$ to be as close as possible to the actual σ_d^2. Therefore the goal is to minimize $\mathrm{E}\{(\lambda_d^{(n)}-\sigma_d^2)^2|\lambda_d^{(n-1)}\}$. After setting the first derivative to zero, we can find the optimal value for the smoothing parameter:

$$\alpha_{\mathrm{opt}}^{(n)}(k) = \frac{1}{1+\left(\dfrac{\lambda^{(n-1)}}{\lambda_d}-1\right)^2}. \tag{4.64}$$

Since the parameter $\alpha_{\mathrm{opt}}^{(n)}(k)$ is always between 0 and 1, a stable non-negative variance estimation is guaranteed. Looking closely at the denominator, we can say that the term $\tilde{\gamma}_k^{(n)} = \lambda^{(n-1)}/\lambda_d$ can be recognized and estimated as a smoothed version of the a-posteriori SNR $\gamma_k^{(n)}$. The values of the parameter $\alpha_{\mathrm{opt}}^{(n)}$ for $\tilde{\gamma}_k^{(n)}$ varying between 0 and 10 are shown in Figure 4.12. Note that the minimum statistics approach does not require VAD, as the smoothed a-posteriori SNR $\tilde{\gamma}_k^{(n)}$ is estimated for each frame and then the optimal parameter $\alpha_{\mathrm{opt}}^{(n)}(k)$ adapts accordingly.

4.4.2 A-Priori SNR Estimation

Most of the noise suppression rules described so far depend on two parameters: a-priori and a-posteriori SNRs, ξ_k and γ_k, respectively. They have to be estimated for each

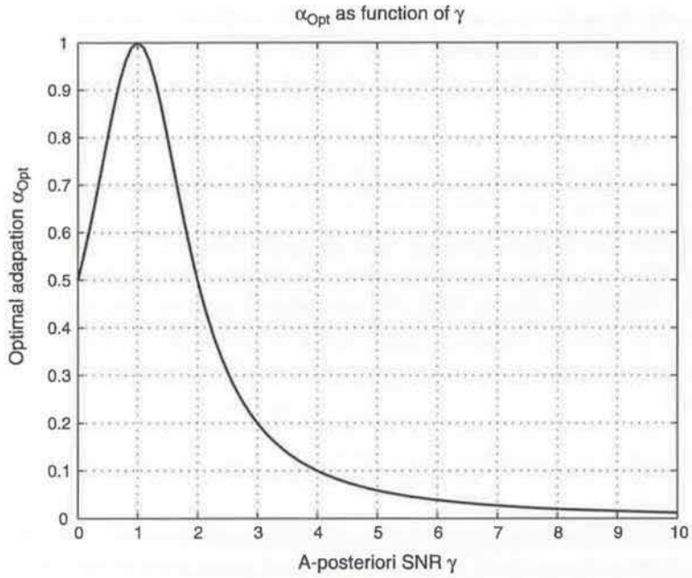


Figure 4.12 The adaptation parameter optimal in a minimum statistics sense as a function of the smoothed a-priori SNR

frame and frequency bin, according to Equation 4.36. With the noise variance estimate $\lambda_k^{(n)}$ and current magnitude $|Y_k^{(n)}|$ in the frequency bin, estimation of the a-posteriori SNR $\gamma_k^{(n)}$ is not a problem. However, estimation of the a-priori SNR requires using the clean speech magnitude $|X_k^{(n)}|$ (which we actually want to estimate). There are several approaches to overcome this problem.

The first is to use the fact that noise and speech are not correlated. Then the maximum-likelihood estimate of the a-priori SNR is

$$\xi_k^{(n)} = \max[0, (\gamma_k^{(n)} - 1)]. \tag{4.65}$$

We already discussed the disadvantages of this approach in the section on suppression rules, and experiments confirmed that the Wiener suppression rule based on this estimate (Equation 4.14) performed worse than the one based on the actual a-priori SNR estimation (Equation 4.12).

A definitely better approach is based on the fact that the speech signal changes more slowly than the normal duration of an audio frame, 10–40 ms. This makes the speech signal in consequent audio frames highly correlated, and we can use the previous clean-speech signal estimate $|\hat{X}_k^{(n-1)}|$ as an approximate value of the clean-speech signal magnitude in the current frame. This is called the *decision-directed approach* (DDA)

and was derived by Ephraim and Malah [3]. Analyses showed that this approach contributes as much to the success of their noise suppressor as the suppression rule derived by them. The a-priori SNR is estimated as

$$\xi_k^{(n)} = \beta \frac{|\hat{X}_k^{(n-1)}|^2}{\lambda_d^{(n)}} + (1-\beta).\max[0, (\gamma_k^{(n)} - 1)] \tag{4.66}$$

where β varies between 0.9 and 0.98. The second term is mostly a stabilizer and plays a role at the beginning. Regardless of the fact that this is an approximate solution, it provides much better results with any suppression rule than the estimator from Equation 4.65. On the negative side, the decision-directed approach introduces dependency of the current output on the values of previous outputs; that is, it converts the noise suppressor to a system with feedback. This immediately raises concerns about the stability of the entire system. These concerns increase especially when we have modules with gain higher than 1. This is a case with both suppression rules from Ephraim and Malah and their computationally efficient alternatives. Under certain circumstances and not very well set time constants, these noise suppressors can become unstable and provide output with audible echoes. This is not the case with the Wiener or probabilistic suppression rules.

Another potential problem with a DDA-based estimation of a-priori SNR is that it provides a one-frame-delayed estimation, which in the transition moments (from speech to noise and from noise to speech) is not correct. This problem can be addressed by using forward–backward DDA [17]. Assuming backward off-line processing, we can define the backward DDA-based a-priori SNR estimation as

$$\xi_{Bk}^{(n)} = \beta \frac{|\hat{X}_k^{(n+1)}|^2}{\lambda_d^{(n)}} + (1-\beta).\max[0, (\gamma_k^{(n)} - 1)]. \tag{4.67}$$

Note that, for estimation of the n-th frame, frame $(n+1)$ is used. The final estimation of the a-priori SNR is a linear combination between the forward and backward estimations:

$$\xi_k^{(n)} = \beta_k^{(n)} \xi_{Fk}^{(n)} + \left(1 - \beta_k^{(n)}\right) \xi_{Bk}^{(n)} \tag{4.68}$$

where $\xi_{Fk}^{(n)}$ is the forward estimation according to Equation 4.66, $\xi_{Bk}^{(n)}$ is the backward estimation according to Equation 4.67, and $\beta_k^{(n)}$ is a time- and frequency-dependent adaptation constant. At the beginning of a speech segment, $\xi_{Bk}^{(n)}$ should be preferred, while at the end of the speech segment $x_{Fk}^{(n)}$ is a better estimate. The adaptation constant can be computed using labeled audio files and learning techniques, or interpolated with a smooth function based on the distance after the beginning of the word or before its

end. The proposed approach provides a 0.7–1.1 dB better SNR improvement; and, according to tests conducted with human listeners, the output of a forward–backward DDA-based noise suppressor is preferred. Good estimation of $\xi_{Bk}^{(n)}$ requires at least two frames delay, which is not acceptable in most of the cases when the noise suppressor is part of a real-time communication system.

4.5 Architecture of a Noise Suppressor

Having discussed all the components of a noise suppressor it is time to put all of them together. The overall block diagram is shown in Figure 4.13.

One of the most important things in statistical-model-based noise suppression is to have good estimations of the noise statistical parameters. This can be done in two stages:

- Use a simple VAD like the one described in Section 4.3.1.2, and do frame classification and create a rough noise model $\tilde{\lambda}_d^{(n)}(k)$ with adaptation per Equation 4.61.
- With the rough noise model, estimate rough a-priori and a-posteriori SNRs, $\tilde{\xi}_k^{(n)}$ and $\tilde{\gamma}_k^{(n)}$, according to Equations 4.66 and 4.3, and use them in a more sophisticated VAD, like the one described in Section 4.3.1.3.
- With the obtained likelihood ratios for each frequency bin, estimate the speech presence probability $P_k^{(n)}(\mathrm{H}_1||Y_k|)$ according to Equation 4.60. Estimate the speech presence probability $P^{(n)}(\mathrm{H}_1)$ for the entire frame as well.
- Use the speech-presence probability to estimate a more precise noise model $\lambda_d^{(n)}(k)$ according to Equation 4.62 or 4.63.

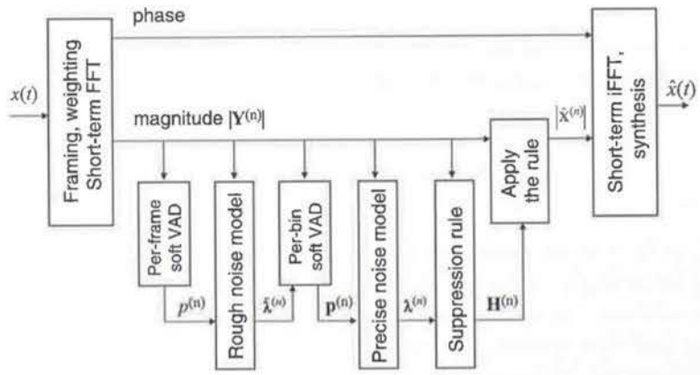


Figure 4.13 Block diagram of noise suppressor

An alternative approach is to use the minimal statistics method, described in Equation 4.64. It can be used for estimation of the final noise model of the first stage only.

The precise noise model $\lambda_d^{(n)}(k)$ can be used for estimation of the final a-priori and a-posteriori SNRs, $\xi_k^{(n)}$ and $\gamma_k^{(n)}$, and the selected suppression rule $H_k^{(n)}$. Spectral subtraction and the log-MMSE estimator are among the most frequently used. The suppression rule can be modified with the speech-presence probability for the entire frame.

Estimate the output signal by applying the suppression rule to the output signal.

The noise suppressor described above is a good practical example of a working system. Properly implemented it sounds good and provides improvement in the overall SNR. The number of potential combinations between various VADs, noise model updaters, and suppression rules is very large and it is not intended here to provide detailed comparison between all variants.

An example of how the noise suppressor works with a real signal is shown in Figure 4.14. Part (a) shows the input signal with SNR = 10 dB in the time domain. Note the noise during the pauses. The same figure shows the speech presence probability, computed later by the voice detector using Equations 4.51 and 4.60. Part (b) shows the spectrogram of the input signal. The noise is easily visible during the pauses and especially in the lower part of the frequency band. There are some constant tones,

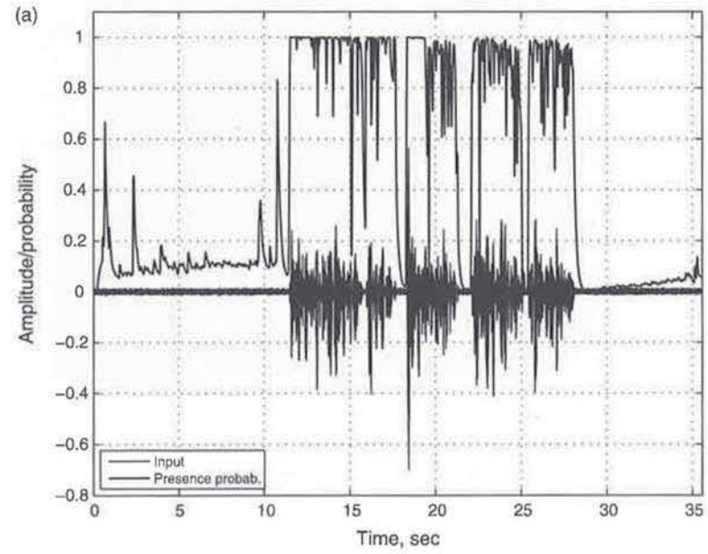
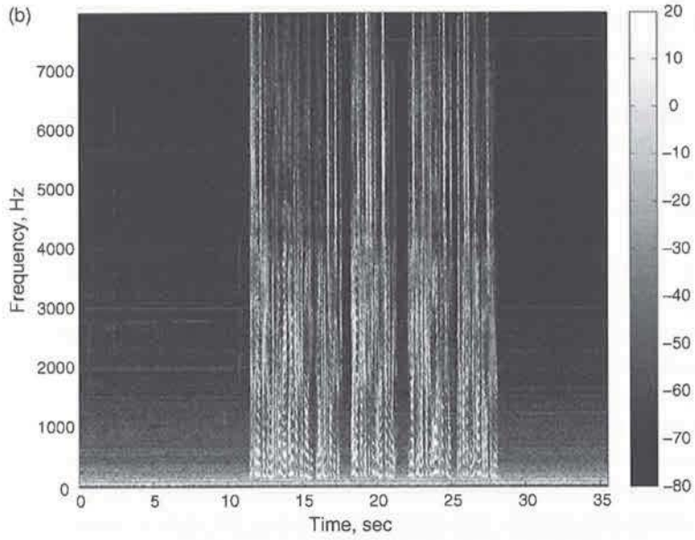


Figure 4.14 Noise suppressor: (a) input signal and speech presence probability per frame; (b) input signal – spectrogram; (c) speech presence probability after the voice activity detector; (d) suppression gain; (e) output signal – spectrogram; (f) output signal in the time domain



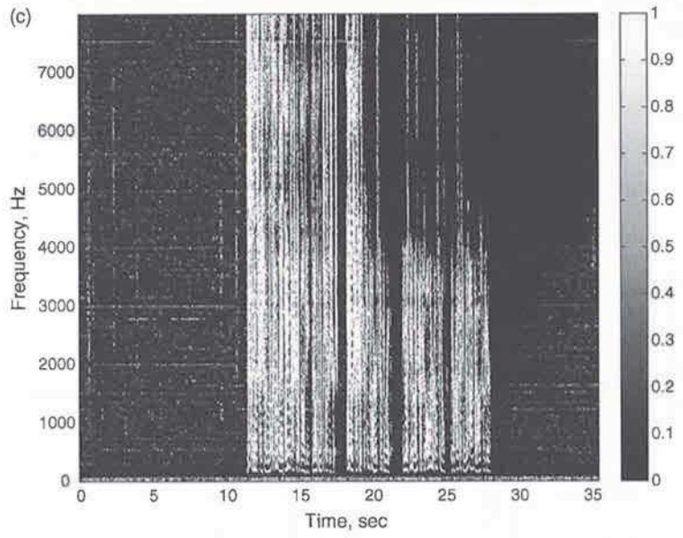
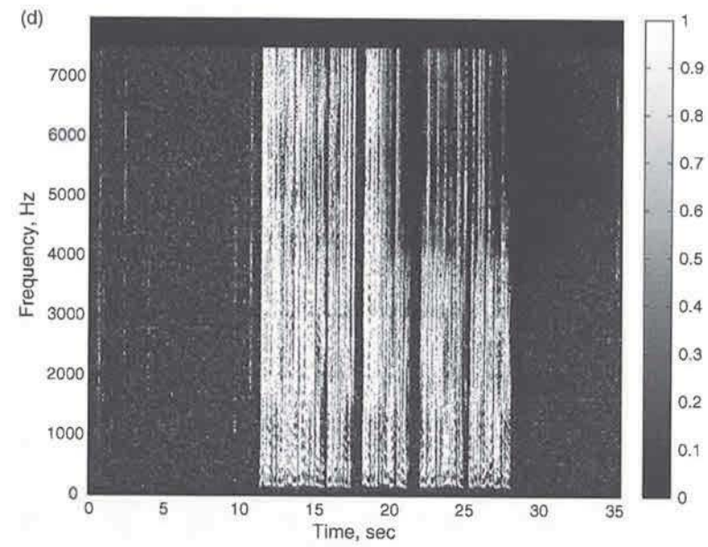


Figure 4.14 *(Continued)*



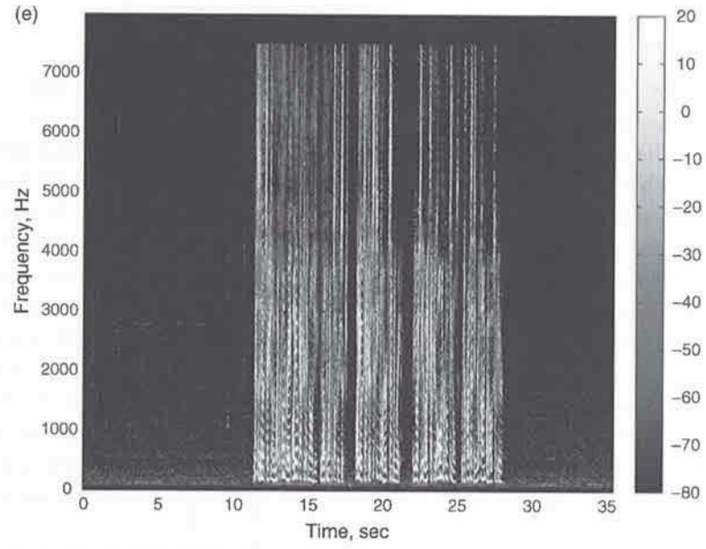


Figure 4.14 *(Continued)*

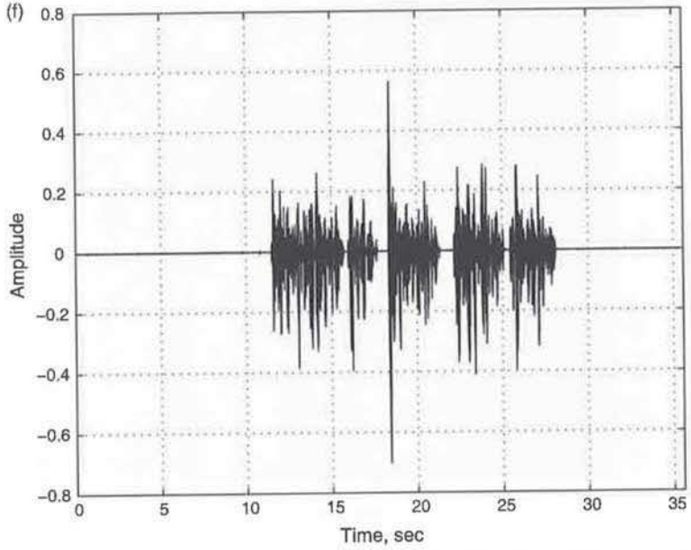


Figure 4.14 *(Continued)*

represented by horizontal lines, caused by computer fans and hard disks. We will discuss a more elegant way to deal with them later in this chapter. Figure 4.14c shows the output of the second voice activity detector. This is the speech presence probability per frequency bin, computed according to Equations 4.46, 4.51, and 4.60. The speech presence probability is used for updating the precise noise model – Equation 4.63. The suppression gain is presented in Figure 4.14c. It is computed according to Equation 4.20 and modified for the uncertain presence of a speech signal in the frame according to Equation 4.55. Note that to the first several frequency bins (below 100 Hz) some minimal gain value is assigned. The noise model there is very volatile and cannot provide a good enough signal estimation. The same is done for the last several frequency bins (above 7500 Hz) to remove aliasing effects. Figure 4.14d shows the spectrogram of the output signal. Note the minimal residual noise in the pause segments and compare this spectrogram with Figure 4.14b. The output signal in the time domain is shown in Figure 4.14e. Compare the noise level with the noise in the input signal – Figure 4.14a.

A comparison of the various suppression rules is presented in Table 4.3. These results are computed using the noise-suppressor implementation above by varying the algorithm for computing the suppression rule. This is similar to the comparison described in Section 4.2.11 and presented in Table 4.2.

From the MMSE perspective, the best results are achieved by the two probabilistic rules. They are apparently more robust to the inevitable errors in estimation of the noise models. At the other end is, again, the ML suppression rule.

Table 4.3 Comparison of various suppression rules for end-to-end noise suppressor

		0 dB				10 dB				20 dB				Average			
Algorithm	Equation	MSE	LSD	SNRI	MOS	MSE	LSD	SNRI	MOS	MSE	LSD	SNRI	MOS	MSE	LSD	SNRI	MOS
Do nothing		0.0429	0.886		2.40	0.0136	0.537		3.16	0.0043	0.301		3.93	0.0203	0.575		3.16
Uncertain presence	(4.54)	0.0184	0.727	7.12	2.62	0.0060	0.484	7.14	3.36	0.0023	0.401	6.85	3.98	0.0089	0.537	7.04	3.32
Wiener, a-posteriori	(4.14)	0.0097	0.516	14.64	2.93	0.0037	0.518	14.48	3.60	0.0018	0.621	13.15	3.96	0.0051	0.552	14.09	3.50
Wiener, a-priori	(4.12)	0.0110	0.497	17.51	2.93	0.0048	0.542	18.09	3.65	0.0022	0.665	16.20	4.05	0.0060	0.568	17.27	3.54
Maximum-likelihood	(4.17)	0.0152	0.655	8.91	2.72	0.0052	0.456	9.04	3.46	0.0021	0.425	8.62	3.99	0.0075	0.512	8.86	3.39
MMSE	(4.18)	0.0116	0.519	14.20	2.85	0.0048	0.468	14.79	3.60	0.0021	0.552	13.90	4.04	0.0062	0.513	14.30	3.50
Log-MMSE	(4.20)	0.0114	0.509	15.15	2.87	0.0049	0.482	15.78	3.62	0.0022	0.578	14.65	4.05	0.0061	0.523	15.19	3.51
Spectral subtraction	(4.16)	0.0115	0.529	13.16	2.88	0.0043	0.469	13.41	3.62	0.0019	0.546	12.62	4.03	0.0059	0.515	13.06	3.51
JMAP SAE	Table 4.1, 1	0.0112	0.513	14.41	2.86	0.0048	0.474	14.96	3.60	0.0021	0.566	14.02	4.04	0.0061	0.518	14.46	3.50
MAP SAE	Table 4.1, 2	0.0119	0.502	15.64	2.89	0.0048	0.496	16.22	3.63	0.0021	0.600	14.95	4.05	0.0060	0.533	15.60	3.52
MMSE SPE	Table 4.1, 3	0.0095	0.532	13.37	2.83	0.0049	0.458	13.91	3.58	0.0021	0.530	13.20	4.04	0.0063	0.506	13.49	3.48
Prob. Gaus-Gauss	(4.32)	0.0085	0.499	19.46	2.90	0.0036	0.559	19.55	3.62	0.0018	0.687	16.50	4.03	0.0049	0.582	18.50	3.52
Prob. Laplace-Gauss	(4.35)	0.0103	0.486	18.67	3.01	0.0034	0.540	18.53	3.68	0.0017	0.670	15.91	4.04	0.0045	0.565	17.70	3.57

From the LSD perspective, best results are achieved by MMSE and log-MMSE rules (optimal in this sense), closely followed by the group of the efficient alternatives. In general all rules achieve quite similar performance.

The two probabilistic suppression rules achieve best noise suppression, followed by the Wiener suppression rule. The ML rule lags behind, as in the clean experiment before.

There is no clear winner as far as the MOS results are concerned, but the two probabilistic suppression rules have a small advantage. Figure 4.15 shows the MOS results as a function of the improvement in SNR. Compare it with Figure 4.9. The first thing to notice is the substantially lower improvement in SNR. Under controlled conditions and precise a-priori and a-posteriori SNR estimations (from the noise-only and speech-only signals, known in this experiment) the improvement in SNR goes above 35 dB in some cases. Here all algorithms are grouped in the range from 9 to 19 dB improvement. Another interesting point is the MOS results. In the controlled experiment we were able to achieve MOS up to 4.1, where under real conditions the average MOS barely approaches 3.6. Note that the sound quality of the output signal does not depend heavily on the suppression rule used. The reliability of the VAD, how precise we can estimate the noise model, and the suppression rule modifier for the uncertain presence of a speech signal are equally important factors for building a good noise suppressor. Another interesting effect to notice is that, while the noise suppressor improves the perceptual quality of the output signal for input SNRs of 0 and 10 dB, the MOS results are much less improved for input SNRs of 20 dB. This is the reason why in some practical implementations the noise suppressor is turned off (or gradually reduced) above a certain input SNR.

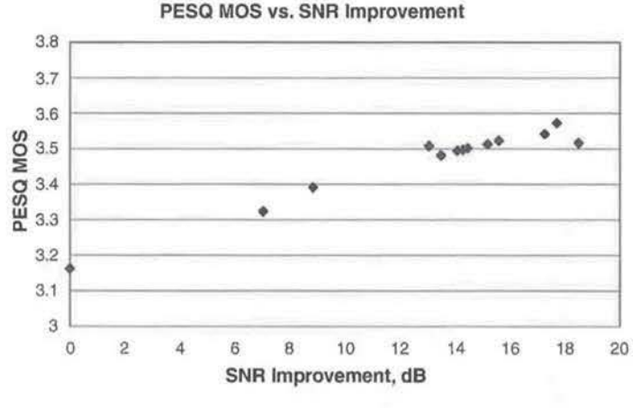


Figure 4.15 MOS results as a function of the SNR improvement for various suppression rules – end-to-end noise suppressor

EXERCISE

Create a MATLAB script for suppression of noise. The script should take the input and output file names as parameters. Use the provided *ProcessWAV.m* script as a template. Follow the steps and formulas above. Use the provided *SimpleVAD.m* for computing the rough noise model, and *SuppressionRule.m* for computing the chosen suppression rule.

Write a MATLAB script to generate a set of WAV files with given SNR (0, 5, 10, 15, 20, and 25 dB, for example) from clean speech and noise (record or use some of the provided). Evaluate the noise suppression script above by processing the generated set of files with noisy speech. Use the provided tool *SNRMeasurement* for evaluation, and, most importantly – listen, listen, listen to the output files.

4.6 Optimizing the Entire System

A good noise suppressor contains many parameters – the adaptation time constants and thresholds seen in many of the equations above. The tuning should start block by block.

The voice activity detectors should be tuned using the approach described at the end of Section 4.3.1.2. This means preparing a set of sound files with SNRs varying over the working range – say, from 0 to 30 dB in steps of 5 dB. This can be done by mixing various noises with a given magnitude to a set of clean speech recordings. Even the simplest VAD works well with a 30 dB SNR. It can be used to label the speech absence and presence frames. Then the mixed files are processed with the algorithm under tuning and the labeled data are used to compute the frame classification accuracy. Maximizing the accuracy can be done completely automatically by using the gradient-descent optimization algorithm.

Once we are sure that each of the blocks works well, it is time to tune the entire system. If the noise suppressor is part of a telecommunication system, the main optimization criterion is how it sounds. This means widely using objective quality measurements (PESQ, for example) to tune the entire system end-to-end. The improvement in SNR should be monitored, and solutions that sound the same but have higher improvement of the SNR should be preferred. The final several variants should go through extensive MOS tests with real human listeners.

If it is expected that speech-enhancement system output will be sent to an automatic speech recognizer, the system should be optimized to maximize the accuracy of the speech recognizer. Note that a speech enhancement system, optimized for speech recognition, may not be optimal for human listeners – and vice versa. If differences are substantial, then a separate processing should be done for the speech recognizer. Note that when doing this the noise-suppression procedure becomes specialized to the particular speech recognizer; that is, it cannot be generic. Different speech recognizers employ various techniques for increasing the noise robustness, and the preceding noise suppressor should not confuse them. If

the speech recognizer builds noise models, then applying the modified suppression rule usually does not work well. The reason for this is that the modified suppression rule suppresses more during the pauses – the time the next block will use to build the noise models. This will result in under-estimation of the noise floor and reduction of the efficiency of this block. Comparison of various noise-suppression algorithms from the speech recognition rate standpoint can be found in [18].

The speech recognition system's sensitivity to distortions and artifacts can affect even the suppression rule. In [19], the suppression rule, which is a function of two parameters, ξ_k and γ_k, is represented as a discrete 10×10 matrix. Fine estimation of the suppression rule values is done by bi-linear interpolation. Then the suppression rule can be subject to optimization with the goal of maximizing the speech recognition rate. As a corpus for speech recognition and optimization, the AURORA 2 test is used. After 15 iterations, the new suppression rule allows the speech recognizer to achieve a higher recognition rate. The starting point (Ephraim and Malah's MMSE estimator) and the resulting suppression rule after 25 iterations are shown in Figure 4.16. It is clear that, after the optimization, the new rule suppresses less noise. This is just an example of using an external optimization criterion (speech recognition error rate) to make one of the components of a speech enhancement system (the suppression rule) better. This technique is applicable to most of the parameters in the noise-suppression system.

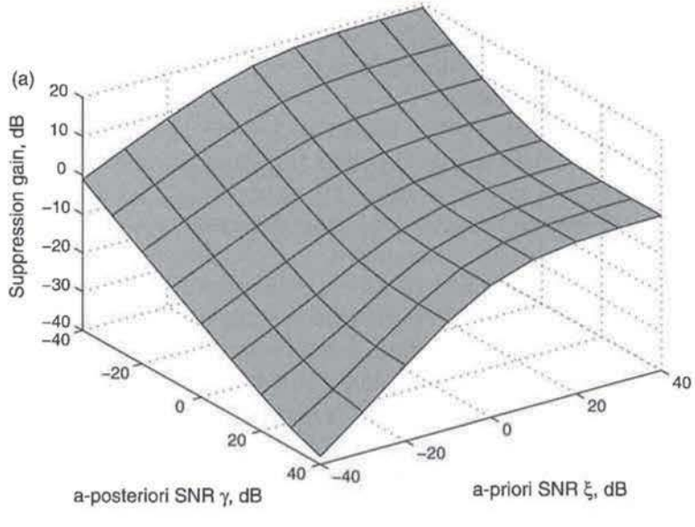


Figure 4.16 Speech-recognition friendly suppression rule: (a) starting point – MMSE suppression rule; (b) after 25 iterations of optimization to maximize the speech recognition rate

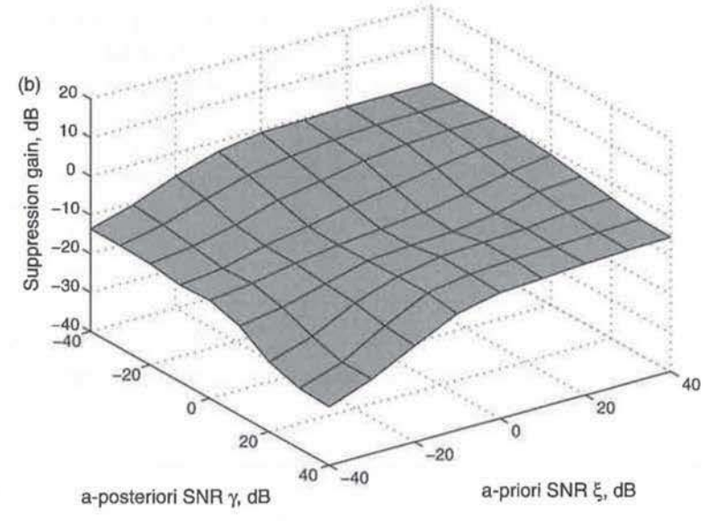


Figure 4.16 *(Continued)*

4.7 Specialized Noise-reduction Systems

4.7.1 Adaptive Noise Cancellation

The noise-suppression techniques described above depend heavily on the accuracy of the noise model. They cannot reconstruct the phase of the original signal, owing to the stochastic nature of the noise. What if we have a distorted copy of the original noise? A good example here is placing a microphone into the engine compartment of a car to pick up the engine noise. The noise, captured from a microphone inside the car cabin, will contain the driver's speech signal and the engine noise, distorted during its propagation to the car cabin. This approach is called "adaptive noise cancellation" and its block diagram is shown in Figure 4.17. The second microphone picks up the noise signal $\mathbf{N}^{(n)}$ – the vector representing the spectrum of this signal in audio frame n. The microphone inside the car cabin acquires the speech signal $\mathbf{X}^{(n)}$ and the engine noise, convolved with the engine-cabin impulse response $\mathbf{H}\cdot\mathbf{N}^{(n)}$:

$$\mathbf{Y}^{(n)} = \mathbf{X}^{(n)} + \mathbf{H}\cdot\mathbf{N}^{(n)}. \tag{4.69}$$

Note that in the frequency domain the convolution becomes multiplication. The active noise canceller tries to estimate the engine–cabin impulse response $\hat{\mathbf{H}}$ and to

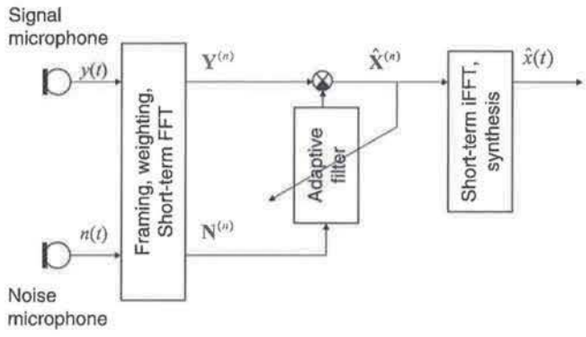


Figure 4.17 Block diagram of adaptive noise cancellation algorithm

calculate the speech signal by subtracting it from the mixture:

$$\hat{\mathbf{X}}^{(n)} = \mathbf{Y}^{(n)} - \hat{\mathbf{H}} \cdot \mathbf{N}^{(n)}. \tag{4.70}$$

Any adaptive filter techniques can be used, but one of the proven effective and most frequently used is the *normalized least-mean-squares* (NLMS). In each silent frame the impulse response is estimated for each frequency bin as

$$H_k^{(n+1)} = H_k^{(n)} + \mu_k \frac{E_k^{(n)} \cdot \bar{Y}_k^{(n)}}{|Y_k^{(n)}|^2 + \varepsilon} \tag{4.71}$$

where ε is a small number to prevent division by zero. $E_k^{(n)} = Y_k^{(n)} - H_k^{(n)} N_k^{(n)}$ is the error that the adaptive filter tries to minimize and that is supposed to be zero during *non-speech* intervals. Here $\bar{Y}_k^{(n)}$ denotes the complex conjugate of the k-th frequency bin in the n-th frame. In general, the NLMS filter is a minimum mean-square-error estimator using the steepest gradient descent. The adaptation time constant μ_k can vary with the frequency bins and the time. It should be $0 < \mu < 1/\lambda_{\max}$, where $\lambda_{\max}$ is the largest eigenvalue of the correlation matrix of the two input signals. In many implementations a fixed value for μ is used.

The adaptation process can be performed on every bin of every frame if we believe that the captured noise **N** does not contain a portion of the speech signal **X**. If there are leaks of **X** in **N**, then it will be better to do the adaptation only during the silence intervals. This will prevent cancellation of portions of the speech signal. If we have estimates of the speech absence probability for the frequency bin $P_k^{(n)}(\mathrm{H}_0||X_k|)$, provided by a VAD, then we can modify the adaptation step $\mu = P_k^{(n)}(\mathrm{H}_0||X_k|)\tilde{\mu}$. Here $\tilde{\mu}$ is the initial (non-variable) adaptation step. In silent frequency bins the adaptation goes with this step, while in frequency bins with speech present the adaptation slows down to prevent cancelling of the speech signal.

The theory of adaptive filters and algorithms is out of the scope of this book. There are numerous papers, books, and book chapters discussing the variable-size adaptation step, other adaptation algorithms, constraints to guarantee convergence, and so on. See [20] for an exhaustive theory of adaptive filters.

Only one dominant noise source is assumed in the adaptive noise cancellation. The delay between the noise-capturing microphone and the speech-capturing microphone should be less than one-quarter of the frame length. For the commonly used frame lengths of 10–50 ms, this means a distance between 3.4 and 17.15 m. In a car environment, for example, the distance between the microphone in the engine compartment and the microphone in the cabin is under 2 m, which means that any frame length above 25 ms will contain practically the same engine noise. If the delay between the noise-capturing and speech-capturing microphones is larger, and the frame length should not be increased for other reasons (latency of a real-time communication system, for example), then a specialized delay-estimation and time-shifting logic should be implemented for the signal from the noise microphone. This logic should delay the noise signal in a way so as to align it with the same noise component in the speech microphone.

The limitations of this type of noise cancelling are due to the presence of other noise sources. In the car example this can be the noise from the tires. In this case we have four additional and independent noise sources. The noise microphone in the engine compartment captures the noise from the engine and from all four tires with a given delay between the engine noise and the noise signal from each tire. The adaptive filter will try to subtract this mixture from the noise, captured by the speech microphone. As the delay from each tire to the speech-capturing microphone is different, there will be a residual noise even after the adaptive filter converges. The adaptive noise-cancellation system cannot cancel noises that are not presented in the noise-only signal – wind noise in the cabin, or other passengers talking. Note that for best results the two ADCs have to have synchronous clocks, otherwise tracking the clock drift will decrease the performance of the adaptive filter.

Overall this technique is linear and does not introduce distortions. It is complimentary to the classic noise-suppression techniques and should precede the non-linear noise suppressor. In many cases, adding a second microphone is justified by the speech enhancement this system can achieve. The general idea of having a noise-only signal, which can be subtracted from the mixture of useful signals and noise using an adaptive filter, will be explored further in the chapter on microphone arrays.

EXERCISE

Create a MATLAB script for adaptive noise cancellation. The script should take as parameters the input and output file names. Perform the processing in the frequency domain. Use the provided *ProcessWAV.m* script as a template. Follow the steps and formulas above. Use the provided *SimpleVAD.m* for controlling the adaptation process.

Evaluate the solution using the provided two-channel WAV file. The first channel is recorded with a microphone inside the car cabin, the second channel is the engine sound only. Measure the improvement in SNR and listen to the output.

4.7.2 Psychoacoustic Noise Suppression

4.7.2.1 Human Hearing Organ

The human hearing organ – the ear – consists of outer, middle, and inner parts. A very schematic diagram of the ear is shown in Figure 4.18. Note that not all parts are present in this figure. Part of the inner ear, for example, is the human vestibular system, which has nothing to do with hearing.

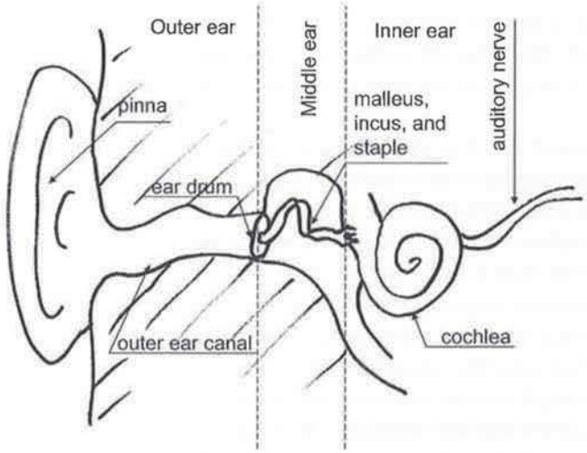


Figure 4.18 Human hearing organ

The outer ear – the pinna and outer ear canal – play a role in forming the directivity of the human hearing, together with the head and the shoulders. In addition, the pinna acts like a funnel, amplifying and directing the sound to the outer ear canal. It increases the sound level 10–15 dB in the frequency range between 1.5 and 7 kHz, where the energy of human speech is concentrated. The outer ear canal is a tube with a length of 30 mm, and a practically uniform diameter of around 7 mm. The ear canal has an acoustical resonance at approximately 3000 Hz, which is the reason for increased sensitivity of humans to these frequencies. The ear canal ends with an ear drum, which converts the changes in the air pressure to mechanical vibrations.

The middle ear is an air-filled cavity and contains the smallest bones in the human body: malleus, incus, and staple. They transfer the mechanical vibrations of the ear drum to the inner ear, which is filled with liquid. These three bones form an acoustical impedance transformer to maximize the energy transmission.

The inner ear is filled with liquid. The staple vibrations are transferred to the cochlea – a tubular organ with a spiral shape (the name comes from the Latin word for snail), and 2.5 turns. The tube is 32 mm long and has a diameter of 0.05 mm at the beginning and 0.5 mm at the end. Inside the cochlea is the Organ of Corti, which consists of thousands of hair cells. They sense the movements of the liquid in the inner ear and cause the neurons connected to them to fire. The auditory nerve transfers the electrical pulses to the brain for further processing. Without going into great detail, the specific shape and construction of the cochlea results in different frequencies agitating different sets of hair cells: for lower frequencies the ones at the end of the Organ of Corti, for higher frequencies the ones at the beginning. This is how humans can distinguish between frequencies.

Detailed description of the physiology and acoustics of the human hearing organ can be found in [21].

4.7.2.2 Loudness

Human hearing is not perfect; the same sound pressure level generates a different audio sensation for different frequencies. The perceived loudness of the sound is a function of both the frequency and sound pressure level. This is why we introduce the loudness level – the sound pressure level of a reference frequency that causes the same subjective loudness. The reference frequency is chosen to be 1000 Hz, and the measuring unit is called a "phon." This means that a sound with magnitude 40 dB SPL and frequency 1000 Hz will have loudness of 40 phons. The human ear is less sensitive towards the lower frequencies, which means that a sound with higher SPL will be necessary to cause the same audio sensation. These equal-loudness curves were studied in the early years by Fletcher and Munson [22] and later were replaced by more precise measurements done by Robinson and Dadson [23]. These measurements became the basis of the standard ISO 226. Later they were revised based on newer and more precise measurements, done by scientists from various countries, and the standard was updated in 2003 as ISO 226:2003.

The equal-loudness curves for several loudness levels are shown in Figure 4.19. The dashed line shows segments with low confidence, or not confirmed by many measurements. The role of the outer ear is clearly visible, as the human hearing has the highest sensitivity in the range 300–7000 Hz. The human sensitivity degrades smoothly for sounds with lower frequencies and rapidly for sounds with frequencies above 15 000 Hz. The upper threshold of human hearing is age-dependent. Young people at the age of 25 years can hear frequencies up to 19–20 kHz, while at the age of 45 years the upper frequency for individuals with normal hearing is down to 15–16 kHz. The curve for 0 phons is actually the threshold of human hearing as a function of the frequency.

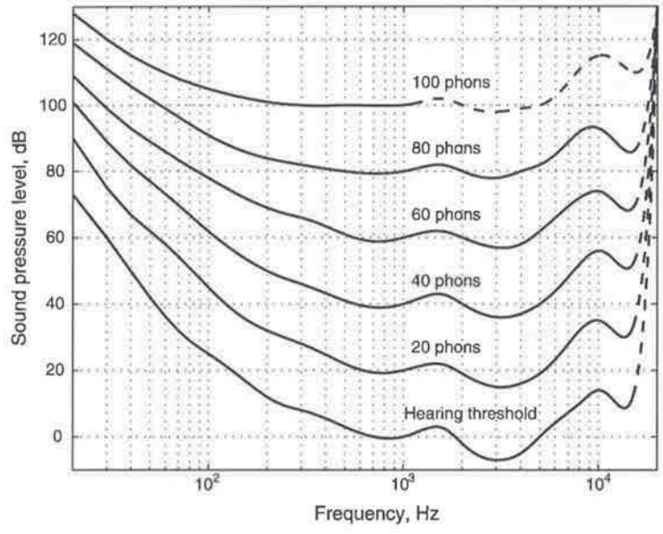


Figure 4.19 Equal-loudness curves for several loudness levels as a function of the frequency and sound pressure level

4.7.2.3 Masking Effects

The human ability to distinguish the frequencies of single tones is quite remarkable – we have a spectral resolution of about 4 Hz for frequencies below 500 Hz, which slowly degrades above this frequency, but remains better than 0.7%. In total, around 640 frequency steps can be distinguished in the audible range for humans.

The situation changes when several sinusoidal signals with different loudness are involved. Experiments and measurements found that the spectral selectivity of human hearing can be modeled as a set of filters with asymmetric shape and a frequency-dependent bandwidth – constant and around 100 Hz in the lower part of the frequency band, decreasing to ~20% of the center frequency above 500 Hz. This simply means that a tone with frequency 1000 Hz will cause a sensation to a group of hair cells in the cochlea around the ones responsible for detecting this frequency; that is, it will "leak" into the neighboring frequencies. Figure 4.20 shows the excitation of the hair cells in the cochlea for a triple-tone audio signal – 500, 2000, and 8000 Hz – according to [21]. The leakage of the signal in the neighboring frequencies is clearly visible.

To model the non-linear frequency resolution of the human ear, the *Bark scale* is introduced, named after the famous physicist H. G. Barkhausen. It converts the frequencies in hertz into a number in the range from 1 to 25 in a non-linear manner,

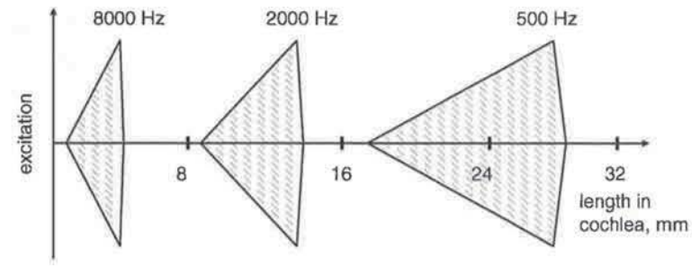


Figure 4.20 Excitation of the hair cells in the cochlea for a triple-tone audio signal – 500, 2000, and 8000 Hz with the same amplitude. Signal leakage towards neighboring hair cells is clearly visible. The horizontal axis is the length inside the cochlea in millimeters, the vertical is the relative magnitude of the excitation

approximated by the formula

$$Z_b(f) = 13 \arctan(0.00075f) + 3.5 \arctan\left[\left(\frac{f}{7500}\right)^2\right]. \tag{4.72}$$

Figure 4.21 shows the relationship between the linear frequency scale and the Bark scale. The dots mark the frequencies for Bark values of 0.5, 1.5, 2.5, and so on. The leaking of the single-tone frequencies is modeled as a triangle-shaped filter in the Bark scale with slopes +25 dB/Bark and −10 dB/Bark, respectively. The more convenient and smoother empirical model for the slope is given in [24]:

$$A(\Delta B) = 15.81 + 7.5(\Delta B + 0.474) - 17.5\sqrt{1 + (\Delta B + 0.474)^2}. \tag{4.73}$$

Here ΔB is the distance from the center frequency in Barks and $A(\Delta B)$ is the attenuation in decibels. This means that the human auditory system can be modeled as a set of overlapping filters with asymmetric triangle-shaped frequency responses. The center frequencies of these filters are chosen to be the dots in Figure 4.21. To simplify the model further, these filters are replaced with a set of rectangular non-overlapping filters with equivalent noise bandwidth; that is, under white-noise excitation they will provide the same average magnitude of the output signal as the corresponding triangle-shaped filter. For the band around 1000 Hz this is shown in Figure 4.22. Instead of using the triangle-shaped filter with a peak at 1000 Hz, the rectangular filter will be used, which passes all frequencies between 920 and 1080 Hz. Table 4.4 shows the center, beginning, and end frequency of the commonly used filter bank of rectangular filters; see [25] for more details. The widely used name for these filters is "critical bands."

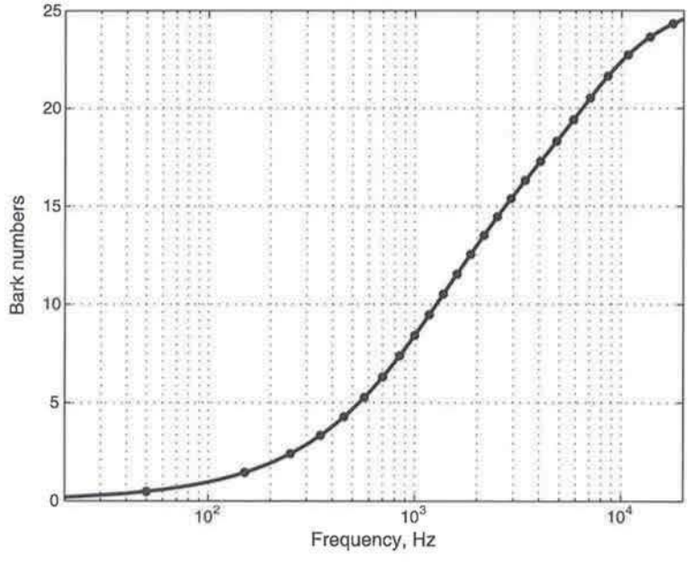


Figure 4.21 Conversion function from frequency to Bark scale. The dots show the positions of integer Bark numbers: 1, 2, 3, and so on

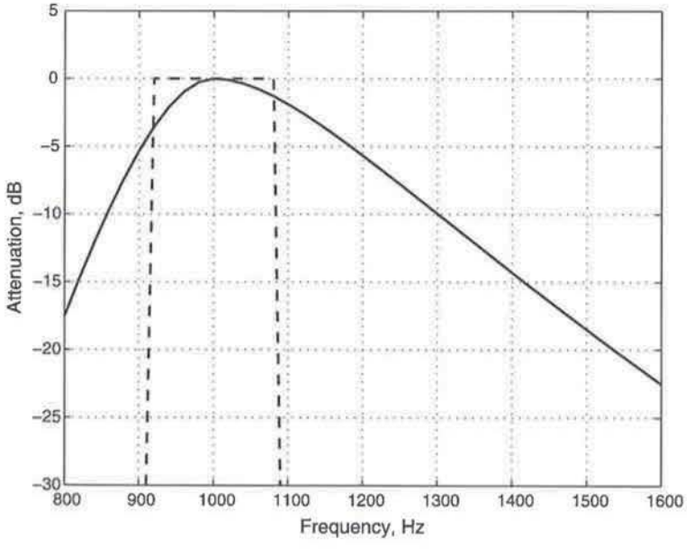


Figure 4.22 Equivalent rectangular bandwidth filter for the critical band with center 1000 Hz. The solid line is the critical band filter, the dashed line the equivalent rectangular bandwidth filter

Table 4.4 Center, beginning, and end of the critical bands (hertz)

Center	Beginning	End
50	0	100
150	100	200
250	200	300
350	300	400
450	400	510
570	510	640
700	630	775
840	765	920
1000	920	1095
1170	1075	1275
1370	1265	1490
1600	1480	1740
1850	1710	2010
2150	1990	2340
2500	2310	2725
2900	2675	3175
3400	3125	3750
4000	3650	4450
4800	4350	5350
5800	5250	6450
7000	6350	7900
8500	7600	9750
10 500	9250	12 250
13 500	11 750	20 000

The masking effects in the frequency domain are caused by the leaking. If we have additional signals with lower amplitude and close frequency, they may be below the hearing threshold. This is shown in Figure 4.23, where we have three signals with close frequencies. The 1000 Hz tone has the highest amplitude. The tone with frequency 1500 Hz and lower amplitude will not be audible, because it is masked by the leakage from the 1000 Hz tone. The third tone with frequency 3000 Hz and the same amplitude as the second one will be audible, since it is far enough to be masked. This simply means that the auditory nerves have a non-linear response to the audio sensation: there is an absolute hearing threshold, and there is a relative threshold, which indicates that the variation in excitation amplitude can be detected only if it is above this relative threshold. The absolute threshold is the 0 phones curve from Figure 4.21. Humans can detect the presence of an additional tone if it is ~8 dB above the masking threshold [26]. Multiple studies of human hearing have shown that a noise can mask a single tone if its level is ~4 dB above the tone, and a tone can mask noise if it has a level of ~24 dB above the noise in that Bark band [27]. These thresholds are used for designing more sophisticated audio compression algorithms.

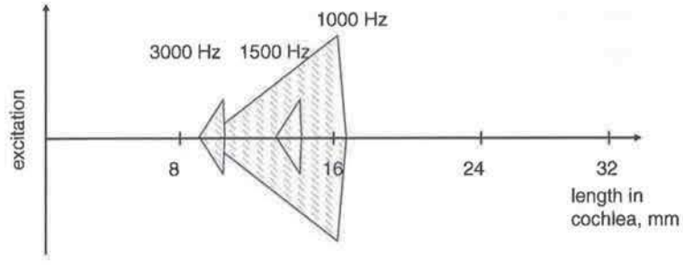


Figure 4.23 Excitation of the hair cells in the cochlea for a triple-tone audio signal – 1000, 1500, and 3000 Hz with different amplitudes. The signal with frequency 1500 Hz will be masked, while the signal with frequency 3000 Hz will be audible. The horizontal axis is the length inside the cochlea in millimeters, the vertical is the relative magnitude of the excitation

In addition to the masking effect in the frequency domain there is masking in the time domain as well, known as "non-simultaneous masking." Pre-masking appears when the masking tone starts some time after the masked tone or noise. This is shown in Figure 4.24. Measurable results registered for 20–30 ms, when the masking threshold is at −50 dB of its simultaneous masking threshold level. The effect is much more sensible in the post-masking case. Signals are still masked 100–200 ms after the masking tone is removed. It is assumed that the masking threshold is at −30, −40, and −45 dB of its simultaneous masking level after 50, 100 and 150 ms, respectively [28].

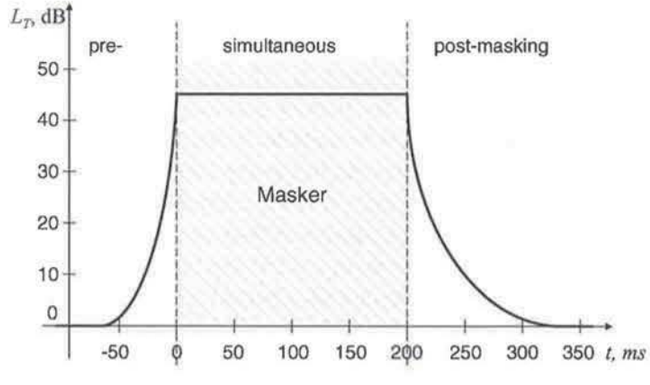


Figure 4.24 Masking in time: pre-, simultaneous, and post-masking effects. The horizontal axis is time, with masking white-noise signal between 0 and 200 ms. The vertical line is the necessary level L_T above the noise level for the audibility of a sinusoidal burst (test tone)

At this point we have all the necessary numbers and models to estimate the masking threshold $M_k^{(n)}$ in the frequency domain, given the input magnitude $|Y_k^{(n)}|$ for given frequency bin and frame. It gives the level below which humans cannot hear tones or noises. If we want to use the post-masking effects, we will have to have the magnitude of several previous frames – that is, $|Y_k^{(n-1)}|$, $|Y_k^{(n-2)}|$, $|Y_k^{(n-3)}|$, and so on. The algorithm is simple and starts with computing the average magnitude in each critical band. Then for each band the masking slopes towards the higher and lower frequencies per critical band are computed. The highest value from all the bands is compared with the hearing threshold. The higher value is the masking threshold for the current frame. Humans cannot hear the tones in each frequency bin if their magnitude is lower than the masking threshold.

The major application of psychoacoustics and masking effects is in audio compression. The principle is "do not store or transmit frequency bins that we cannot hear anyway". All major audio compression techniques today (MPEG-1, MPEG-2, MP3, MPEG-4, Windows Media Audio, Windows Real-Time Voice, etc.) are based on this principle. Discussing audio compression techniques is not in the scope of this book, but we will focus on the psychoacoustic approach for noise suppression.

4.7.2.4 Perceptually Balanced Noise Suppressors

We have seen that more suppression means more distortions in the estimated clean signal. In the psychoacoustic approach for noise suppression, the principle is "do not suppress noise below the level we can hear." In some papers it is defined as the principle of "least processing." Wolfe and Godsil [29] apply a perceptually modified suppression rule as follows:

$$\tilde{H}_k^{(n)} = \begin{cases} \left(1 - \dfrac{M_k^{(n)}}{|Y_k^{(n)}|}\right) H_k^{(n)} + \dfrac{M_k^{(n)}}{|Y_k^{(n)}|} & \text{for} \quad M_k^{(n)} < |Y_k^{(n)}| \\ 1 & \text{otherwise.} \end{cases} \tag{4.74}$$

Here, $\tilde{H}_k^{(n)}$ is the perceptually modified suppression rule and $H_k^{(n)}$ is the suppression rule estimated using any of the algorithms already discussed. In that particular paper the authors use the short-term MMSE rule from Ephraim and Malah. The ratio $M_k^{(n)}/|Y_k^{(n)}|$ is the relative masking level. Note that there is no processing done when the relative masking level is above 1; that is, on the masked frequency bins. The relative masking level is used as a limiting factor to the suppression gain, which does not allow suppression below the audible threshold.

The authors report a marginal decrease in the SNR improvement, which is expected, but increase in the user's preference.

Note that the perceptual-based noise suppressors are not compatible with any psychoacoustic-based audio compressor because they are created on antagonistic principles. It is pointless to have a perceptual-based noise suppressor, followed by a psychoacoustic compressor. The major application of perceptually balanced noise suppressors is for cleaning and restoring high-quality music recordings. By minimizing the intervention only to the hearable part of the noise and limiting the suppression to go no further than the masking threshold, the introduced distortion is minimized and the user perception of the restored records is improved.

4.7.3 Suppression of Predictable Components

Frequently speech recordings are contaminated by stationary tones. They usually come from power wiring, inadequate shielding, or grounding of the microphone cables, or placement of the microphones near power lines or transformers. In those cases the interference frequency is 50/60 Hz or 400 Hz and their harmonics. Other kinds of stationary-tone interference come from microphones positioned near TVs, monitors, or video cameras; the microphones can capture interference at frame or line frequencies acoustically from transformers or electronically from the cables. Yet another source of this kind of interference are noises coming from the acoustical environment, such as fans, computer hard drives, and air conditioning. Because of non-linearities and room reverberation, these signals behave mostly as random zero-mean Gaussian noise, but usually there are still predictable components. The frequencies of the predictable portion of these noises vary depending on the fan or hard drive spindle rotating speed. The common property of these signals is that they are practically stationary. In their time–frequency representations they show up as horizontal lines with constant amplitude.

The most intuitive approach to solve this problem and to clean up the contaminated signal is to apply band-pass filtering or notch filters tuned to the constant tones. These approaches remove speech signal components if the interfering frequencies are within the speech band. If the speech signal is contaminated by single-tone interference, then a notch filter works well and the missing frequency is usually inaudible. If the contaminating signal has multiple harmonics, then a set of notch filters or a comb filter may be needed to achieve significant filtering, and that can substantially distort the speech signal.

Classic noise suppressors, like the ones described above, assume that the noise is a stationary zero-mean Gaussian process and build a statistical model of the noise as a vector of variances per frequency bin. The stationary tones have a probability density function that is usually not Gaussian. Using a Gaussian PDF as a model of these signals and some of the known suppression rules (Wiener, or Ephraim and Malah, etc.) results in complete suppression of the speech signals in these frequency bins; that is, the noise suppressor converts to a notch filter for these frequencies.

The problem of tracking frequencies in a time-frequency representation is well studied. In [30], an ARCAP (autoregressive Capone algorithm) method is used to identify the spectral lines, followed by Kalman filtering to track their movement. The method is illustrated with a processing of avalanche signals. It is sensitive to noise and best results are obtained with a forward–backward Kalman filter, which makes it inapplicable for real-time algorithms where low latency is desired. Improving the algorithm further [31] by adding trajectory smoothing with a Fraser filter still keeps the algorithm good for off-line processing only. The birth/death time estimation of spectral lines is improved in [32] as well. In addition, a particle filter is used to perform optimal estimation in jump Markov systems for detection and tracking of spectral lines. The proposed time-varying autoregressive (TVAR) estimator is evaluated with synthetic signals. It is computationally expensive and sensitive to the times of birth/death of spectral lines. Andia [33] proposes image processing techniques to be used to detect, model and remove spectral lines from the time–frequency representation. All of these approaches solve problems that are more complex than necessary, and are mostly suitable for off-line processing of the contaminated signals.

One of the main properties of the predictable components is that they are stationary, or pseudo-stationary, and can be modeled as a linear combination of sinusoidal signals and a noise component:

$$z(t) = \sum_{i=1}^{L} A_i \sin(2\pi f_i t) + \mathbb{N}(0, \lambda) \tag{4.75}$$

where L is the number of stationary tones, each with frequency f_i. Converting this signal to the frequency domain yields the following model for the n-th audio frame:

$$Z_k^{(n)} = \sum_{i=1}^{L} W_T(k) * A_i e^{-j2\pi nTf_i} + \mathbb{N}(0, \lambda_N) \tag{4.76}$$

where W_T is the Fourier image of the frame weighting function, T is the audio frame step, n is the frame number, and k is the frequency bin.

We note the following:

- Due to the "smearing" of the spectral lines because of the weighting, bins neighboring the central bin (for each contaminating frequency) contain portions of the energy.
- These neighboring bins will rotate in the complex plane (phase shift) from frame to frame with the same speed, which can be different than the speed of each bin's central frequency $\exp(-j2\pi nTf_s/K)$.

These two aspects introduce additional complications in the extrapolation of the signal model for the next frame.

Assuming we have perfect estimation for the n-th frame $\hat{Z}_k^{(n)}$, then the extrapolation for the next frame will be

$$\hat{Z}_k^{(n+1)} = \hat{Z}_k^{(n)} \frac{\sum_{i=1}^{L} W_T(k) * A_i e^{-j2\pi(n+1)Tf_i}}{\sum_{i=1}^{L} W_T(k) * A_i e^{-j2\pi nTf_i}}. \tag{4.77}$$

The second term is a complex number that represents the "speed" of rotation of our complex model from frame to frame. As already noted, this "speed" can be different from the "speed" of the central frequency of the bin. Because $W_T(k)$ decays quickly with increasing k, we can assume that one frequency from the contaminating signal dominates in each frequency bin. In this case

$$\frac{\sum_{i=1}^{L} W_T(k) * A_i e^{-j2\pi(n+1)Tf_i}}{\sum_{i=1}^{L} W_T(k) * A_i e^{-j2\pi nTf_i}} \approx e^{-j2\pi Tf_I} + \mathbb{N}(0, \lambda_E). \tag{4.78}$$

where f_I is the dominant frequency and $\mathbb{N}(0, \lambda_E)$ is an error term, modeled as zero-mean Gaussian noise. As the dominant frequency is unknown, the extrapolation can be presented as

$$\hat{Z}_k^{(n+1)} = \hat{Z}_k^{(n)} \hat{Y}_k^{(n)} \tag{4.79}$$

where $\hat{Z}_k^{(n)}$ is the contaminating signal estimation for the n-th frame, and $\hat{Y}_k^{(n)}$ is the rotating speed of the model towards the next frame. Both components have additive Gaussian noise with variances λ_N and λ_E, respectively.

With the speech signal present, Equation 4.75 takes the form

$$x(t) = s(t) + \sum_{i=1}^{L} A_i \sin(2\pi f_i t) + \mathbb{N}(0, \lambda) \tag{4.80}$$

and the representation in the frequency domain of the n-th frame is

$$X_k^{(n)} = S_k^{(n)} + \sum_{i=1}^{L} W_T(k) * A_i e^{-jnTf_i} + \mathbb{N}(0, \lambda_N). \tag{4.81}$$

In this case our estimation of the speech signal is

$$\hat{S}_k^{(n)} = X_k^{(n)} - Z_{est}^{(n)}(k); \tag{4.82}$$

that is, we just subtract our estimation of the contaminating signal

$$Z_{\text{est}}^{(n)}(k) = \hat{Z}_k^{(n-1)} \cdot \hat{Y}_k^{(n-1)}. \tag{4.83}$$

The speech signal estimation contains the captured noise $\mathbb{N}(0, \lambda_N)$ and the cancellation adds an additional noise component $\sim \mathbb{N}(0, \lambda_E)$ due to the approximations in the model and estimation errors.

In parallel with the contaminating signal cancellation, we should constantly update the contaminating signal model, which for each frequency bin consists of four elements: $\hat{Z}(k)$, $\hat{Y}(k)$, $\lambda_N(k)$, and $\lambda_E(k)$ (from which only the first two are involved in the constant-tones cancellation process). The contaminating signal model is updated as follows:

$$\hat{Z}_k^{(n)} = (1-\alpha)\hat{Z}_k^{(n-1)} + \alpha\left(p_k^{(n)} X_k^{(n)} + \left(1-p_k^{(n)}\right)\hat{Z}_k^{(n-1)}\right) \tag{4.84}$$

where $\alpha = T/\tau_Z$, τ_Z is the adaptation time constant, and $p_k^{(n)}$ is the probability that we have only contaminating signal in $X_k^{(n)}$ – that is, the probability of speech absence. It can be provided by any voice activity detector (VAD), which produces per-bin probability estimation of speech presence. The additive noise variance is updated as follows:

$$\lambda_N^{(n)} = (1-\alpha)\lambda_N^{(n-1)} + \alpha\left(p_k^{(n)}\delta_k^{(n)} + \left(1-p_k^{(n)}\right)\lambda_N^{(n-1)}\right) \tag{4.85}$$

where $\delta_k^{(n)} = ||X_n^{(n)} - Z_{est}^{(n)}(k)||^2$. The rotating speed estimation is updated in the same way:

$$\hat{Y}_k^{(n)} = (1-\beta)\hat{Y}_k^{(n-1)} + \beta\left(p_k Y_{est}^{(n)}(k) + (1-p_k)\hat{Y}_k^{(n-1)}\right) \tag{4.86}$$

where

$$Y_{est}^{(n)}(k) = \frac{Y_k}{||Y_k|| + \varepsilon}$$

is the normalized rotation speed estimation

$$Y_k = \frac{X_k^{(n)}}{X_k^{(n-1)} + \varepsilon}$$

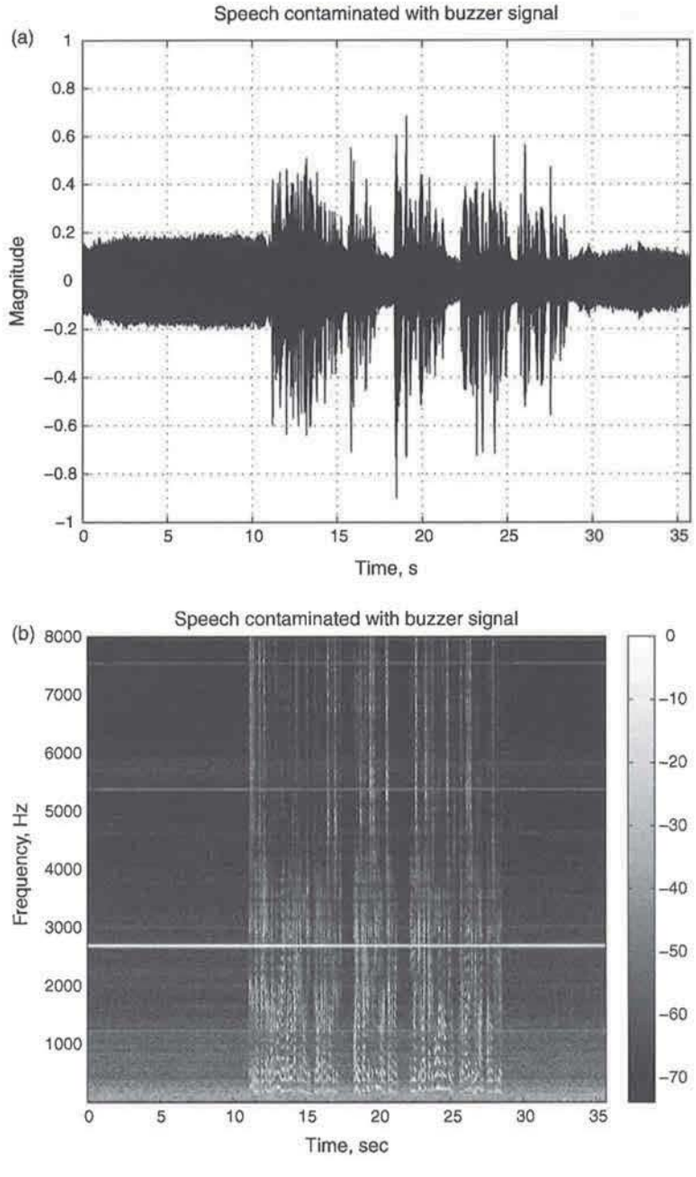


Figure 4.25 Cancellation of predictable contaminating signals: (a) speech signal, contaminated with buzzer signal; (b) spectrogram of the contaminated signal, the buzzer signal being visible as horizontal lines; (c) cleaned speech signal; (d) spectrogram of the cleaned speech signal

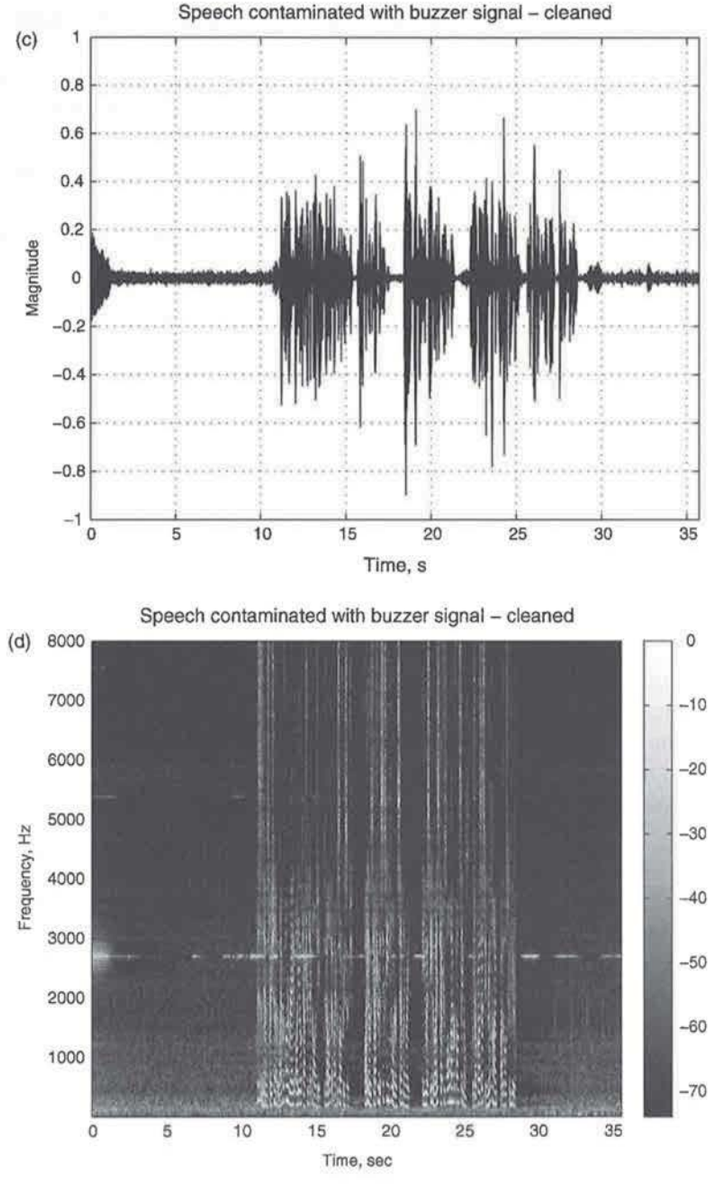


Figure 4.25 *(Continued)*

for the current frame, ε is a small number, $\beta = T/\tau_Y$, and τ_Y is the adaptation time constant.

The proposed method is evaluated in [34] with a speech signal contaminated with several noises: white noise, office noise, and two buzzer sounds with a different number of harmonics. The improvement in SNR is shown in Table 4.5. There is no suppression for white noise and clean speech, as expected. For office noise (three computers with their fans and hard drives, air conditioning) the algorithm improves the SNR by almost 3 dB, removing the predictable components from the noise. The proposed algorithm suppresses the signals from the two buzzers up to 15 dB.

Table 4.5 Improvements in SNR with predictable signals compensation (decibels)

	Input			Output			
Recording	Signal	Noise	SNR	Signal	Noise	SNR	Improvement
White noise		−13.43			−13.43		0.00
Clean speech	−25.37	−60.44	35.07	−25.38	−60.75	35.37	0.30
Office noise	−34.55	−44.62	10.07	−35.02	−47.98	12.96	2.89
Buzzer 1	−21.42	−21.69	0.27	−23.19	−39.35	16.16	15.89
Buzzer 2	−18.56	−20.52	1.96	−24.21	−39.96	15.75	13.79

Figure 4.25 shows the contaminated and cleaned signals and their spectrograms. The contaminating signal is visible as three horizontal lines. This is a real recording in a room where people move, changing the reverberation and interference patterns. After each change the algorithm has to adapt to the new signals. During this time we see the bright traces in the spectrogram. Their magnitude is still much lower than the captured signal, which is visible in the time domain representation of the output signal.

This type of processing is suitable as a pre-processor, before a classic noise suppressor. It removes the predictable part without artifacts and musical noise, leaving the noise suppressor less to suppress, which in general means less musical noise and artifacts. It is computationally inexpensive and even in office conditions reduces almost 3 dB of the noise, which is well audible. It is a safety net when the microphone is accidentally placed near sources of predictable noises. It successfully removes most of the hard drive spindle noise, captured by the microphone in a laptop, but signal and audio processing are not fixes for a not very thoughtful design – the microphones should be kept away from such noise sources.

EXERCISE

Create a MATLAB script for cancellation of predictable components of the noise. The script should take as parameters the input and output file names. Perform the processing in the frequency domain. Use the provided *ProcessWAV.m* script as a template. Follow the steps and formulas above. Use the provided *SimpleVAD.m* for controlling the adaptation process.

Evaluate the solution using the provided *SpeechBuzzer.WAV* file. Measure the improvement in SNR and listen to the output.

4.7.4 *Noise Suppression Based on Speech Modeling*

The algorithms discussed so far assume one model of the speech signal, usually statistical with Gaussian, gamma, or Laplace distribution. In Chapter 2 we saw that speech is a complex signal and consists of segments with quite different characteristics. Apparently each of the segments can have an optimal suppression filter that is quite different from the optimal filters for the other types of segments.

The idea of different filters for different segments of the speech signal was first proposed by Drucker [35]. In his paper he groups the approximately 40 phonemes in the English language into five broad classes: stops, fricatives, glides, vowels, and nasals. Each phoneme is processed by a separate filter, designed to eliminate the intra-class confusion – that is, the error of assigning a class sound from the same class – and the use of a different filter eliminates interclass confusion. The proposed algorithm works as follows (Figure 4.26). The input speech plus noise is segmented into phonemes; a decision device determines which of the five classes of sounds the phoneme belongs to, and then routes the speech plus noise segment to the appropriate filter. According to the paper, the proposed algorithm increases the intelligibility around 25% in the input SNRs ranging from −8 dB to 0 dB; above

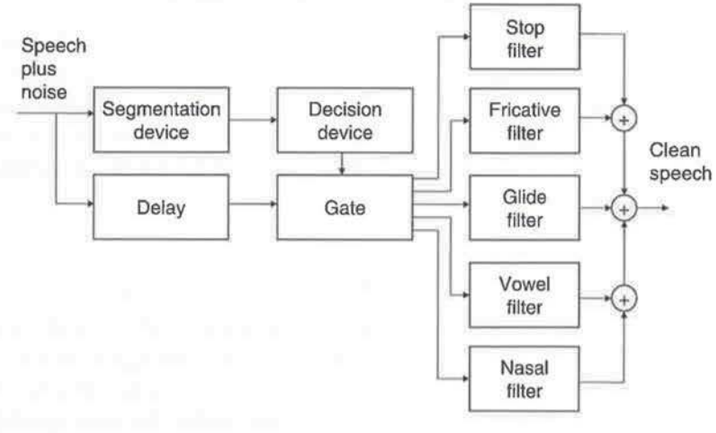


Figure 4.26 Block diagram of a noise suppressor based on speech modeling

0 dB the improvement in the intelligibility decreases to ~7%. The major source of potential problems in this approach is the classifier.

Many researchers later tried either to reduce the classification error, or to synthesize more robust filters to such errors. Algorithms for precise measurement and tracking of the pitch frequency and synthesis of the proper comb filters proved to be very efficient for denoising of the vowels.

In a later work, Ephraim and Cohen [36] proposed parallel processing, based on the probability of classification. In general the processing model with certain probability of speech presence is a multi-class model. From this perspective a useful modeling apparatus can be a *hidden Markov process* (HMP). The classes are not a-priori defined, but they are created in a learning process from training data of clean speech signals. It is in fact a clustering process that can be performed by using vector quantization techniques. Each class may contain spectrally similar vectors of the signal, which can be parameterized as an autoregressive process. Transformations from one spectral prototype to another can be modeled by the HMP. The various noises can be processed in a similar manner. If we have M speech classes and N noise classes, then we have $M \times N$ combinations; that is, estimators for enhancing the speech signals. For a given sequence of noise speech vectors $\mathbf{Y}^t = \{Y^{(n)}, Y^{(n-1)}, \ldots, Y^{(n-t)}\}$, and the probability $p(i, j|\mathbf{Y}^t)$ of the signal being in state i and the noise being in a state j given $\mathbf{Y}^t$, the MMSE estimator of the clean speech signal $\mathbf{x}$ is

$$\mathrm{E}\{\mathbf{X}^t|\mathbf{Y}^t\} = \sum_{i=1}^{M}\sum_{j=1}^{N} p((i,j)|\mathbf{Y}^t)\mathrm{E}\{\mathbf{X}^t|\mathbf{Y}^t,(i,j)\}. \tag{4.87}$$

4.8 Practical Tips and Tricks for Noise Suppression

While the mathematical models above are correct and valid to describe the behavior of captured and processed signals, there are some additional tips and tricks that can and should be used for successful implementation of a real noise suppression and reduction system. Audio processing is a science, an art, and a craft. So far we have covered the science part and touched on the art part when talking about human perception. This section is about the craft.

4.8.1 Model Initialization and Tracking

Each time the processing starts there can be different noise levels, or rotational speed. Using some default initial value and adaptation equations like Equation 4.63 leads to slow adaptation in the initial phase. On the other hand, decreasing the time constants for faster adaptation leads to less stable values of the model. To adapt quickly at the beginning and keep the values stable during normal working, we can use a variable time

constant in the initial phase:

$$\lambda_d^{(n)} = \begin{cases} |Y_k^{(n)}|^2 & \text{if} \quad n = 1 \\ \left(1-\frac{1}{n}\right)\lambda_d^{(n-1)} + \frac{1}{n}|Y_k^{(n)}|^2 & \text{if} \quad 1 < n < \frac{\tau_d}{T} \\ \left(1-\frac{T}{\tau_d}\right)\lambda_d^{(n-1)} + \frac{T}{\tau_d}|Y_k^{(n)}|^2 & \text{for} \quad n \geq \frac{\tau_d}{T}. \end{cases} \tag{4.88}$$

In the first frame we initialize with the value of the current frame; the second frame is the average between the first and second frames; and so on until we reach the number of frames big enough to use as an adaptation factor T/τ_d.

4.8.2 Averaging in the Frequency Domain

The noise amplitude in each of the frequency bins varies substantially. To build a more precise model we should increase the adaptation time constant. Unfortunately this leads to slower adaptation to changes in the noise. A good trade-off is to use smoothing towards the neighboring frequency bins as well. Weighting causes leakage, so even a single sinusoidal signal will be represented in several frequency bins. Usually just a moving average of three to seven frequency bins does a sufficient job to stabilize the rough noise model, used in VADs.

4.8.3 Limiting

For robustness to accidental spikes and errors, the values of the measured parameters should be kept within reasonable limits. The standard deviations cannot go below certain minimal values. The mean parameter in a gamma distribution should not go below the standard deviation of the Gaussian noise. Every probability has values between $0 + \varepsilon$ and $1 - \varepsilon$. Likelihood should be limited to be above $0 + \varepsilon$ and some certain number that is not too big – 1000 is a good practical value. Good limitation for signal-to-noise ratios of any kind is in the range −60 dB to + 60 dB. Suppression gains should be limited to the range 0–1. Proper limiting of the value range in the real-time execution code allows the algorithm to be more robust to unexpected input data or computational errors.

4.8.4 Minimal Gain

Zeroing some frequency bins is never a good idea. It is a harsh operation for the sound which results in musical noise and unpleasant distortions for the human ear. The

background noise in the silent segments is distorted and chopped. To prevent this from happening, a minimal gain should be applied after estimating the final suppression rule:

$$H_{\text{final}}^{(n)}(k) = (1-G_{\min})H_k^{(n)} + G_{\min}. \tag{4.89}$$

The most frequently used value for the minimum gain is 0.1, which limits the suppression to 20 dB. The level of musical noises is negligible and the background noise, while suppressed, is not distorted.

4.8.5 *Overflow and Underflow*

The noise suppression and all other audio processing algorithms are usually implemented as real-time processing modules. There is no time to handle all potential overflow and underflow exceptions; this is why proper measures to avoid them should be taken. Earlier in this chapter it was said that, in real-time implementations, $1/x$ becomes $1/(x + \varepsilon)$, where ε is a small positive number. The same is true for computing logarithms (log-likelihood, for example). Then $\log(x)$ becomes $\log(\max(\varepsilon, x))$ or $\log(x + \varepsilon)$. In many cases we compute exponents and large arguments can cause exceptions or undefined results. For double precision, floating-point numbers $\exp(x)$ overflows when x is slightly greater than 700. For secure execution we should use $\exp(\min(700, x))$.

4.8.6 *Dealing with High Signal-to-Noise Ratios*

The noise suppression is a trade-off between suppression and introduced distortions. Most of the algorithms are optimized for SNRs of 5–15 dB – the most common when capturing sound in homes, offices, and conference rooms. This means that these algorithms are suboptimal for high-quality input, when the SNR is 30–50 dB. This can happen when using a headset, for example. Then the noise-suppression algorithms just introduce distortions, decreasing the actual quality and the overall MOS results. In such cases, instead of making the algorithms more complex, it is a better idea just to turn the noise suppression off.

The noise suppressor has a signal/pause classifier anyway. For each frame we can compute the level and add it to our estimation of the signal or of the noise using a certain time constant. With the signal and noise level estimations, computing the average SNR is trivial. If it is high enough we can turn off the noise suppression. Actually the suppression rules work well under very high SNR conditions. They stay around 1 and do not harm the signal quality.

The suppression rule modifier for the uncertain presence of a speech signal (see Equation 4.54) is usually the source of increased distortions under high SNR conditions. In most cases it makes sense to turn only this feature off. This should be done with a certain hysteresis; that is, turn the feature off when the average SNR goes above a high threshold and turn it back on when the average SNR goes below a lower threshold. Typical values for these two thresholds are 20 dB and 25 dB average SNR.

4.8.7 *Fast Real-time Implementation*

The processing algorithms described so far require computation of many probabilities, exponents, gamma functions, and so on. While the processors used in modern personal computers have the integrated ability to work with floating-point numbers, the operations they can do beyond the four arithmetic operations are limited to exponent and tangent. This means that everything else has to be computed in a programmatic way. Not many digital signal processors have the capability to perform operations with floating-point numbers at all. This makes performance optimizations critical for the success – and even applicability – of given algorithms.

Fortunately, the majority of these computations can be performed off-line and kept as a set of tables. Most of the distributions (Gaussian, gamma, Laplace) can be tabulated with steps. In real time, a linear interpolation can be used to obtain the exact value. In many cases the nearest-neighbor algorithm provides sufficient precision. The suppression rule itself is a function of two parameters. It can be discretized as a matrix, computed off-line and used in real time. This can save computations and make the algorithm run faster.

Of course, before going to performance optimization of the execution code, the normal software engineering rules should be followed. The process starts with performance profiling, which provides the time used by the CPU to execute any of the functions in the code. Then the performance optimization starts with the functions with highest execution times. It is pointless to optimize the performance of a function or operation that takes 0.1% of the CPU time; even completely removed it will reduce the execution time only by 0.1%.

In many cases the most computationally expensive part is the conversion to the frequency domain and back – that is, FFT and iFFT functions. From this perspective, the manufacturers of most DSPs provide implementation of the FFT algorithm optimized for their processor. Using frame size, which is a power of two, decreases the execution time of these two operations as well.

4.9 Summary

This chapter has discussed algorithms and approaches for single-channel noise reduction. Clean signal estimation from the mixture of signal and noise is a gain-based process and the algorithms belong to the group of noise suppressors. The process applies a time-varying, real-valued gain to each frame in the frequency domain. The computation of this gain, called a suppression rule, is based on the statistical parameters of both the noise and clean signal. The suppression gain is a

function of the a-priori and a-posteriori SNRs and is optimal in one way or another: the MMSE of the magnitudes, ML, and so on. Applying the suppression rule reduces the noise component, but introduces distortion and artifacts, called musical noise.

An important part of each noise suppressor is the voice activity detector. In its simplest form this is a two-way classifier: the current audio frame contains only noise, or it is a mixture of noise and speech. The most complex VADs provide a speech presence probability for each frequency bin. In some cases this probability is used to modify the suppression rule – the uncertain presence of speech signal approach.

Part of the noise suppression algorithm builds the noise and the speech models. Based on the VAD output, the noise model variance is updated from frame to frame. For estimation of the speech signal statistical parameters, the decision-directed approach is commonly used, which assumes high correlation of the speech signal in consequent frames and uses the previous output frame to estimate the a-priori SNR.

The goal of the noise suppressor is not to remove the noise, but to make the output sound better to humans. Therefore, optimizing the noise suppressor as a system targets maximization of the MOS results, not improvement in the SNR.

To reduce distortion and artifacts, other approaches are used such as adaptive noise cancellation (with a secondary channel for capturing just the noise) or a stationary-tones canceller (which estimates and subtracts the non-random components of the noise). A separate group are perceptually based noise suppressors, which use the masking effects in human hearing to suppress less noise – the parts we cannot hear anyway.

The noise-reduction system in modern communication devices is a real-time running complex program. It can and should be optimized for better initialization and tracking, faster adaptation, and faster execution.

In general, noise reduction is a science, an art, and a craft. It is a science because it deals with mathematical models and has reproducible results. It is an art because it is about the human perception of sounds, where not everything can be modeled with numerical models. It is a craft because there are always better implementations of the same algorithm, some "secret sauce" which makes the entire system work well. This chapter has discussed all three aspects of good noise-reduction systems.

Bibliography

[1] Wiener, N. (1949) *Extrapolation, Interpolation, and Smoothing of Stationary Time Series, with Engineering Applications*, MIT Press, Cambridge, MA.

[2] Wang, D. and Lim, J. (1982) The unimportance of phase in speech enhancement. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, **30**, 679–681.

[3] Ephraim, Y. and Malah, D. (1984) Speech enhancement using a minimum mean-square error short-time spectral amplitude estimator. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, **32**, 1109–1121.

[4] McAulay, R. and Malpass, M. (1980) Speech enhancement using a soft-decision noise suppression filter. *IEEE Transactions on Acoustics, Speech, and Signal, Processing*, **28**, 137–145.

[5] Boll, S. (1979) Suppression of acoustic noise in speech using spectral subtraction. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, **26**, 113–120.

[6] Ephraim, Y. and Malah, D. (1985) Speech enhancement using a minimum mean-square error log-spectral amplitude estimator. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, **33**, 443–445.

[7] Wolfe, P. and Godsill, S. (2001) Simple alternatives to the Ephraim and Malah suppression rule for speech enhancement. Proceedings of 11th IEEE Workshop on Statistical Signal Processing, pp. 496–499.

[8] Martin, R. (2002) Speech enhancement using MMSE short-time spectral estimation with gamma distributed speech priors. Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP), Orlando, FL, pp. 253–256.

[9] Gazor, S. and Zhang, W. (2003) Speech probability distribution. *IEEE Signal Processing Letters*, **10**(7), 204–207.

[10] Van Trees, H.L. (1968) *Detection, Estimation and Modulation Theory* (Part I), MIT Press, Cambridge, MA.

[11] ITU-T (1996) *Recommendation G.729 Annex B: A Silence Compression Scheme for G.729 Optimized for Terminals Conforming to Recommendation V.70*, ITU-T, Geneva, Switzerland.

[12] Sohn, J., Kim, N. and Sung, W. (1999) A statistical model based voice activity detector. *IEEE Signal Processing Letters*, **6**(1), 1–3.

[13] Davis, A. and Nordholm, S. (2003) A low complexity statistical voice activity detector with performance comparisons to ITU-T/ETSI voice activity detectors. Information, Communications and Signal Processing, 2003 and the Fourth Pacific Rim Conference on Multimedia, Dec. 2003, vol. 1, pp. 119–123.

[14] Haykin, S. (1994) *Communication Systems*, 3rd edn, John Wiley & Sons, Chichester, England.

[15] Middleton, D. and Esposito, R. (1968) Simultaneous optimum detection and estimation of signals in noise. *IEEE Transactions on Information Theory*, **14**, 433–443.

[16] Martin, R. (2001) Noise power spectral density estimation based on optimal smoothing and minimum statistics. *IEEE Transactions on Speech and Audio Processing*, **9**, 504–512.

[17] Hendriks, R., Heusdens, R. and Jensen, J. (2005) Forward-backward decision directed approach for speech enhancement. Proceedings of International Workshop on Acoustics, Echo and Noise Control (IWAENC), Eindhoven, The Netherlands.

[18] Song, M.-S., Lee, C.-H. and Kang, H.-G. (2006) Performance of various single channel speech enhancement algorithms for automatic speech recognition. Proceedings of International Conference on Spoken Language Processing (Interspeech), Pittsburg, PA.

[19] Tashev, I., Droppo, J. and Acero, A. (2006) Suppression rule for speech recognition friendly noise suppressors. Proceedings of Eight International Conference Digital Signal Processing and Applications (DSPA), Moscow.

[20] Haykin, S. (2002) *Adaptive Filter Theory*, 4th edn, Prentice-Hall, Upper Saddle River, NJ.

[21] Zwicker, E. and Fast, H. (1999) *Psychoacoustics: Facts and Models*, Springer-Verlag, Berlin.

[22] Fletcher, H. and Munson, W. (1933) Loudness, its definition, measurement and calculation. *Journal of the Acoustical Society of America*, **5**, (2), 82–108.

[23] Robinson, D. and Dadson, R. (1956) A re-determination of the equal loudness relations for pure tones. *British Journal of Applied Physics*, **7**, (5), 166–181.

[24] Schroeder, M., Atal, B. and Hall, J. (1979) Optimizing digital speech coders by exploiting masking properties of the human ear. *Journal of the Acoustical Society of America*, **64**(S1), S139.

[25] Schraf, B. (1970) Critical bands, in *Foundations of Modern Auditory Theory*, Academic Press, New York.

[26] Malvar, H. (2004) Auditory masking in audio compression, in *Audio Anecdotes: Tools, Tips, and Techniques for Digital Audio* (eds K. Greenbaum and R. Basel), A.K. Peters, Natick, MA.

[27] Spanias, A., Painter, T. and Atti, V. (2007) *Audio Signal Processing and Coding*, John Wiley & Sons, Hoboken, NJ.

[28] Vary, P. and Martin, R. (2006) *Digital Speech Transmission: Enhancement, Coding and Error Concealment*, John Wiley & Sons, Chichester, England.

[29] Wolfe, P. and Godsil, S. (2003) A perceptually balanced loss function for short-time spectral amplitude estimator. Proceedings of IEEE ICASSP, Hong Kong, China, **5**, 425–428.

[30] Roguet, W., Martin, N. and Chehikian, A. (1996) Tracking of frequency in a time-frequency representation. Proceedings of IEEE International Symposium on TFTS, pp. 341–344.

[31] Davy, M., Leprette, B., Doncarli, C. and Martin, N. (1998) Tracking of spectral lines in ARCAP time-frequency representation. Proceedings of EUSIPCO, Rhodes Island, Greece.

[32] Andrieu, C., Davy, M. and Doucet, A. (2003) Efficient particle filtering for jump Markov systems: application to time-varying autoregressions. *IEEE Transactions on Signal Processing*, **51**, 1762–1770.

[33] Andia, B. (2006) Restoration of speech signals contaminated by stationary tones using an image perspective. Proceedings of IEEE ICASSP, Toulouse, France.
[34] Tashev, I. and Malvar, H. (2007) Stationary-tones Interference Cancellation Using Adaptive Tracking. Proceedings of IEEE ICASSP, Honolulu, HI.
[35] Drucker, H. (1968) Speech processing in a high ambient noise environment. *IEEE Transactions on Audio and Electroacoustics*, **16**, 165–168.
[36] Ephraim, Y., Cohen, I. (2006) Recent advancements in speech enhancement in *The Electrical Engineering Handbook, Circuits, Singals, and Speech and Image Processing*, Richard C. Drof (ed.), Third Edition, CRC Press, Boca Raton, FL, pp. 15-12–15-26.
[37] Benyassine, A., Shlomot, E., Su, H. *et al.* (1997) ITU-T recommendation G.729 annex B: a silence compression scheme for use with G.729 optimized for V.70 digital simultaneous voice and data applications. *IEEE Communications Magazine*, **35**(9), 64–73.
[38] Cho, Y.D., Al-Naimi, K. and Kondoz, A. (2001) Improved voice activity detector based on a smoothed statistical likelihood ratio. Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP), Salt Lake City, UT, vol. 2, pp. 737–740.
[39] Fawsett, T. (2006) An introduction to ROC analysis. *Pattern Recognition Letters*, **27**, 861–874.
[40] Sohn, J. and Sung, W. (1998) A voice activity detection employing soft decision based noise spectrum adaptation. Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP), Seattle, WA, pp. 365–368.

5

Sound Capture with Microphone Arrays

This chapter is dedicated to sound capture with systems of multiple microphones, called "microphone arrays." The sound propagation itself is a three-dimensional process, so that capturing it in one single point, with one microphone, is not sufficient to deal with 3D processes like ambient noise, reverberation, and multiple sound sources. Using several closely positioned microphones allows listening to the sound coming from one direction, while suppressing the noises and interference sounds coming from other directions. In addition, microphone arrays allow estimation of the direction of arrival – that is, sound source localization.

There are two major groups of microphone-array processing algorithms: time-invariant and adaptive. The first are optimal under the assumption of isotropic ambient noise, and they are fast and simple to implement for working in real time. Adaptive processing algorithms shine when we have point noise sources in low reverberant conditions. They require more CPU resources and are more complex to implement. Both approaches assume identical capturing channels and are affected by mismatch, caused mainly by the manufacturing tolerances of the microphones used. This is handled by creating a robustness to manufacturing tolerances in the algorithms, by manufacturing time-calibration procedures, or by real-time autocalibration algorithms. Multiple channels allow the creation of multichannel noise suppressors and spatial filters, which further improve the quality of the sound and increase suppression of unwanted sounds and noise.

5.1 Definitions and Types of Microphone Array

5.1.1 Transducer Arrays and their Applications

The concept of using a set of antennas for directional radio transmission and receiving has been known since World War I, but was employed in practice in radars used during

Sound Capture and Processing Ivan J. Tashev

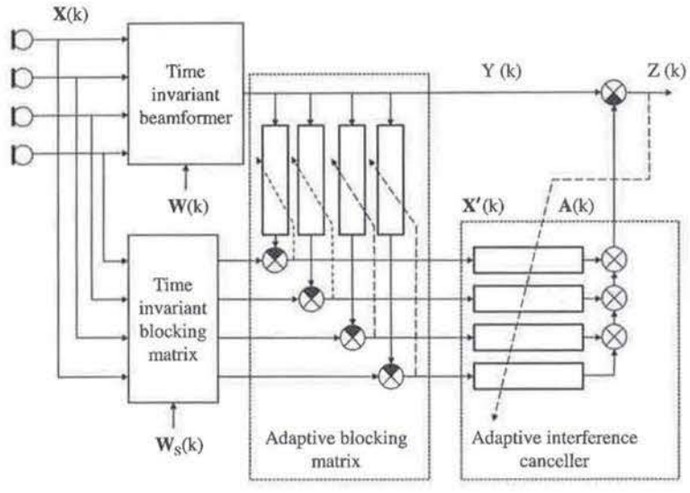


Figure 5.33 Robust generalized side-lobe canceller

5.6.4 Adaptive Algorithms for Microphone Arrays – Summary

Adaptive array algorithms were initially designed for antenna arrays. There they are considered more efficient and suppress more noise due to the fact that they place nulls towards the undesired signal sources. In audio, the efficiency of the adaptive microphone array algorithms is reduced mostly owing to the reverberation. In reverberant conditions there are no point sources – desired and undesired sound sources are smeared. At a critical distance, where the received energy directly from the sound source equals the reverberant energy, even if we place a perfect null towards the unwanted sound source the best we can achieve is a 6 dB attenuation – the direct path. The reverberation is practically isotropic ambient noise. For most rooms and offices the critical distance is close to the work distance for hands-free sound capture – around 1.5 m. From this perspective, an isotropic ambient noise model is closer to reality than a point noise source model. The time-invariant beamformers were optimal exactly for isotropic ambient noise. Under these conditions, a time-invariant beamformer with perfect channel matching is as efficient as an adaptive beamformer; that is, the adaptive beamformer replaces the autocalibration procedure. Still, the adaptive beamformers provide slightly better results in a reverberant environment, with the price of increased CPU and memory use.

5.7 Microphone-array Post-processors

Both time-invariant and adaptive beamformers are linear processors and compute the output signal as a linear combination of the input signals. Even if the weights of

different microphones change, which is the case with adaptive beamformers, they do it much slower than the audio frame rate. Microphone-array post-processors apply real-valued gain which varies from frame to frame in the same way as the static noise suppressors do. The difference is that the gain estimation is based on the additional information about the positions of the desired and undesired sound source, which we have from the multiple channels and eventually the microphone positions.

5.7.1 Multimicrophone MMSE Estimator

Assume that the source signal $S_c(f)$ in Equation 5.27 has variance $\lambda_c(f)$. The noise contains correlated and uncorrelated components, with spectral matrix presented in Equation 5.42:

$$\mathbf{\Phi}_{N'N'}(f) = \mathbf{\Phi}_{NN}(f) + \lambda_{NC}(f)\mathbf{I} \tag{5.94}$$

where $\lambda_{NC}(f)$ is the variance of the uncorrelated noise and $\Phi_{NN}(f)$ is the spectral matrix of the correlated (spatial) noise. Then the spectral matrix of the input signals is

$$\mathbf{\Phi}_{XX}(f) = \lambda_c(f)\mathbf{D}_c(f)\mathbf{D}_c^{\mathrm{H}}(f) + \mathbf{\Phi}_{N'N'}(f). \tag{5.95}$$

Let the estimation of the desired signal be provided by a matrix processor $\mathbf{H}(f)$, which is an M-element complex vector. The derivation provided here follows [2]. Then the mean square error is

$$\begin{aligned} \varepsilon &= \mathrm{E}\{|S_c(f) - \mathbf{H}(f)\mathbf{X}(f)|^2\} \\ &= \mathrm{E}\{(S_c(f) - \mathbf{H}(f)\mathbf{X}(f))(S_c^*(f) - \mathbf{X}^{\mathrm{H}}(f)\mathbf{H}^{\mathrm{H}}(f))\}. \end{aligned} \tag{5.96}$$

Taking the complex gradient with respect to $\mathbf{H}^{\mathrm{H}}(f)$ and setting the result equal to zero gives

$$\begin{aligned} &\mathrm{E}[S_c(f)\mathbf{X}^{\mathrm{H}}(f)] - \mathbf{H}(f)\mathrm{E}[\mathbf{X}(f)\mathbf{X}^{\mathrm{H}}(f)] = 0 \\ &\text{or} \quad \mathbf{\Phi}_{dX^{\mathrm{H}}}(f) = \mathbf{H}_0(f)\mathbf{\Phi}_{XX}(f). \end{aligned} \tag{5.97}$$

From Equation 5.27 and the assumption for uncorrelated signal and noise components:

$$\mathbf{\Phi}_{dX^{\mathrm{H}}}(f) = \lambda_c(f)\mathbf{D}_c^{\mathrm{H}}(f). \tag{5.98}$$

This leads to the solution for the optimal MMSE estimator:

$$\mathbf{H}_0(f) = \lambda_c(f)\mathbf{D}_c^{\mathrm{H}}(f)\mathbf{\Phi}_{xx}^{-1}(f). \tag{5.99}$$

On inverting Equation 5.95 using the matrix inversion formula, we obtain

$$\mathbf{\Phi}_{xx}^{-1} = \mathbf{\Phi}_{N'N'}^{-1} - \mathbf{\Phi}_{N'N'}^{-1}\lambda_c\mathbf{D}_c(1+\mathbf{D}_c^{\mathrm{H}}\mathbf{\Phi}_{N'N'}^{-1}\lambda_c\mathbf{D}_c)^{-1}\mathbf{D}_c^{\mathrm{H}}\mathbf{\Phi}_{N'N'}^{-1} \tag{5.100}$$

where the frequency indices are suppressed for simplicity. After defining

$$\mathbf{\Lambda}^{-1}(f) = \mathbf{D}_c^{\mathrm{H}}(f)\mathbf{\Phi}_{N'N'}^{-1}(f)\mathbf{D}_c(f) \tag{5.101}$$

and on substituting (5.99) in (5.100) we obtain the optimal solution:

$$\mathbf{H}_0(f) = \frac{\lambda_c(f)}{\lambda_c(f)+\mathbf{\Lambda}(f)}\mathbf{\Lambda}(f)\mathbf{D}_c^{\mathrm{H}}\mathbf{\Phi}_{NN}^{-1}(f). \tag{5.102}$$

This MMSE processor is practically a multichannel Wiener filter. Its block diagram is shown in Figure 5.34. Taking a closer look, it is easy to see that it consists of an MVDR beamformer – compare the right part of Equation 5.102 with Equation 5.38, and something close to the single-channel Wiener filter, described in Chapter 4. The noise variance $\lambda_d(f)$ is computed by $\mathbf{\Lambda}(f)$, defined in Equation 5.101. In this chapter we ignore the beamformer and focus on the post-processor only.

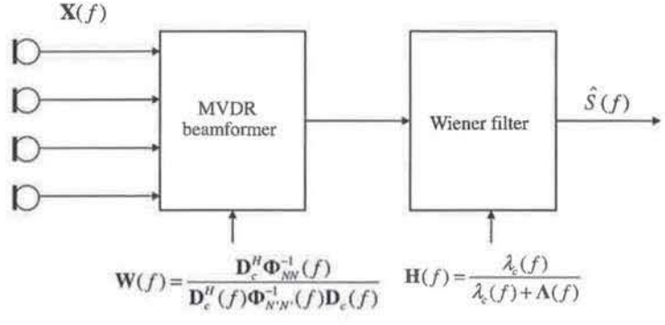


Figure 5.34 Multimicrophone MMSE estimator

5.7.2 *Post-processor Based on Power Densities Estimation*

Equation 5.102 provides the optimal solution under the assumption of the known desired signal variation $\lambda_c(f)$ and the noise cross-power matrix $\mathbf{\Phi}_{XX}(f)$. Estimation of these is not trivial in real conditions, where all we know is the input signals and the microphone-array geometry. One of the first practical applications of post-processor for microphone arrays was published by Zelinski [23]. Calculating the auto- and cross-spectral densities of the aligned (i.e., properly delayed) channels i and j leads to (all

frequency indices omitted for simplicity)

$$\phi_{X_iX_i} = \phi_{SS} + \phi_{N_iN_i} + 2\Re\{\phi_{SN_i}\} \tag{5.103}$$

$$\phi_{X_iX_j} = \phi_{SS} + \phi_{N_iN_j} + \phi_{SN_j} + \phi_{N_iS}. \tag{5.104}$$

Zelinski makes the following assumptions:

- The signal and noise are uncorrelated ($\phi_{N_iS} = 0, \forall i$), which is in general true unless we do not consider the early reverberation of the desired signal as noise.
- The noise power spectrum is the same on all sensors, ($\phi_{N_iN_i} = \phi_{NN}, \forall i$), which restricts the algorithm to microphone arrays with the same type of sensors.
- The noise is uncorrelated between sensors, ($\phi_{N_iN_j} = 0, \forall i \neq j$), which again excludes the early reverberation.

Under these assumptions, Equations 5.103 and 5.104 are reduced to

$$\phi_{X_iX_i} = \phi_{SS} + \phi_{N_iN_i} \tag{5.105}$$

$$\phi_{X_iX_j} = \phi_{SS}. \tag{5.106}$$

They can be estimated using a standard smoothing in time:

$$\hat{\phi}_{X_iX_j}^{(n)} = (1-\alpha)\hat{\phi}_{X_iX_j}^{(n-1)} + \alpha X_iX_j^* \tag{5.107}$$

where $\alpha = T/\tau$, T is the frame direction, and τ is the update time constant. Then the numerator and denominator in the post-processor part of Equation 5.102 can be estimated more robustly by averaging the spectral densities over all the possible channel combinations, resulting in the post-filter estimator (frame indices omitted for simplicity):

$$\hat{H}_{\mathrm{PF}} = \frac{\frac{2}{M(M-1)}\sum_{i=1}^{M-1}\sum_{j=i+1}^{M}\Re\{\hat{\phi}_{X_iX_j}\}}{\frac{1}{M}\sum_{i=1}^{M}\hat{\phi}_{X_iX_i}}. \tag{5.108}$$

The real operator $\Re\{\cdot\}$ is used because the term, estimated in the nominator, is required to be a real.

The Zelinksi post-processor is a good approximation and works well for the lower part of the frequency band, where the assumptions above hold better, but it is less

7

Acoustic Echo-reduction Systems

Acoustic echo-reduction systems have been an integral part of telephones from very early, and the quality of the echo reduction is critical for the performance of every communication system that works in speakerphone mode. Acoustic echo cancellation (AEC) was one of the earliest applications of adaptive filters and one of the most studied. The purpose of the acoustic echo canceller is to remove from the microphone signal the sounds from the local loudspeaker. This is done by employing an adaptive filter to estimate the transfer function between the loudspeaker and the microphone. The adaptive filter processes the signal sent to the loudspeakers. Its output is subtracted from the microphone signal. This is why in many cases the entire acoustic echo-reduction system is called an acoustic echo canceller.

However, we shall see later in this chapter that AEC is just one of several processing stages. Owing to noise and reverberation, the transfer function estimation is not exact and the adaptive filter cannot remove completely the captured loudspeaker sound, called "echo." The second stage in acoustic echo-reduction systems is a suppression-based non-linear processor. It suppresses the residual energy similarly to a noise suppressor – by applying a time-varying real gain to each frequency bin.

Removing the captured sound from stereo and surround-sound audio systems is challenging owing to the non-uniqueness of the solution for the transfer function. There are several methods to mitigate this problem. In many practical echo-reduction systems additional blocks are used to quickly handle the case when feedback occurs, to track sampling rate drifts, and so on. They are described towards the end of the chapter.

7.1 General Principles and Terminology

7.1.1 Problem Description

The effect of sound reflection from walls and objects is called "reverberation." A human voice recorded in a studio with a closely positioned microphone has no

Sound Capture and Processing Ivan J. Tashev

reverberation. It sounds unnatural and usually is called "dry." With some reverberation the human voice sounds warmer and more natural. Echoes are distinct copies of the reflected sound. Humans can hear echoes when the difference between arrival times of the direct signal and the reflection is more than 0.1 s, but even with differences of 0.05 s the audio sounds echoic. In 0.1 s the sound travels 34.3 m, which means that if the object reflecting the sound is further away than 17 m (the sound travels to the object and back) the reflection will be heard as echo. Acoustic echo-reduction does not suppress the echoes in the room. It actually suppresses the *effect* when the local sound source is captured by the microphone, transmitted through the communication line, reproduced by the loudspeaker in the receiving room, captured by the microphone there, returned back through the communication line, reproduced from the local loudspeaker, and so on. This creates an echoic sound and in some cases causes feedback – that is, the entire system converts to a generator, reproducing an annoying constant tone. In the context of acoustic echo-cancellation, echo is the sound from the local loudspeaker captured by the local microphone.

These echo effects were a problem in telephones even before their official discovery. The first prototypes used four wires. They were practically two independent sets, each consisting of microphone, battery, two wires, and headphone. The patent application filed by Antonio Meucci on 28 December 1871 uses four wires in order to eliminate the "local effect," which is nothing but hearing your own voice in the headset. After failing to pay the patent application fee, two years later Meucci abandoned his patent application. This allowed Alexander Graham Bell to file, on 14 February 1876, his patent application for the invention we call today the "telephone." Later, the four wires were replaced by two wires; that is, the same two wires are used to carry the electrical signal from both sides. The first telephones actually used one wire and closed the circuit through the ground in a similar way that telegraphs were doing at that time. Mixing the incoming and outgoing signals caused problems with the "local effect," which was resolved by using a Wheatstone bridge to separate the two signals in telephone circuitry. As the telephone line impedance participates in the bridge, any change in the impedance impairs the suppression of the local effect. With increasing length of the telephone lines there appeared signal reflections when the impedances of the line and the telephone were not balanced. This required the presence of an echo suppressor in each telephone, implemented initially as passive circuitry that inserted signal losses (a pair of diodes which open when signal levels exceed a certain threshold). Later this circuitry was replaced by the "line echo canceller" (sometimes called a "network echo canceller"). Since the mid-1960s the line echo canceller has been implemented as an electronic adaptive filter. When speakerphones appeared they required suppression of the acoustic echo as well. This block is called the *acoustic echo canceller* and is the subject of this chapter.

The theory of acoustic echo cancellation was initially developed by AT&T Bell Labs [1] but was deployed only in the late 1970s owing to performance limitations of electronic blocks of that time. They become cost-effective in the 1990s, and currently

adaptive echo cancellers are part of practically every mobile or stationary telephone with speakerphone capabilities.

7.1.2 Acoustic Echo Cancellation

In telecommunications, *near-end* denotes our end of the communication chain: microphone, loudspeaker, and sound. On the other side are the *far-end* microphone, loudspeaker, and sound. The schematic diagram is shown in Figure 7.1. The near-end speaker talks to the microphone and the audio signal is sent to the far-end room. There the near-end speaker's voice is reproduced by the far-end loudspeaker. The far-end microphone captures the sound from this loudspeaker and the voice of the far-end speaker. When both the local and the remote speakers talk simultaneously we have so-called *double talk*. The signal captured by the far-end microphone is transmitted through the communication line and reproduced by the near-end loudspeaker. Without acoustic echo cancellers the near-end speaker will reproduce the delayed and decayed copy of the near-end speech, which will be captured by the near-end microphone and the entire process will be repeated many times, causing annoying echoes. This system with feedback under certain conditions can become unstable and convert itself into a generator of a specific audio frequency, making communication impossible. Both near- and far-end stations have to have acoustic echo cancellers to remove the sound captured from the local loudspeakers. Then each station will transmit only the local voice. This breaks the feedback chain, the echoes are gone, and audio feedback is not possible.

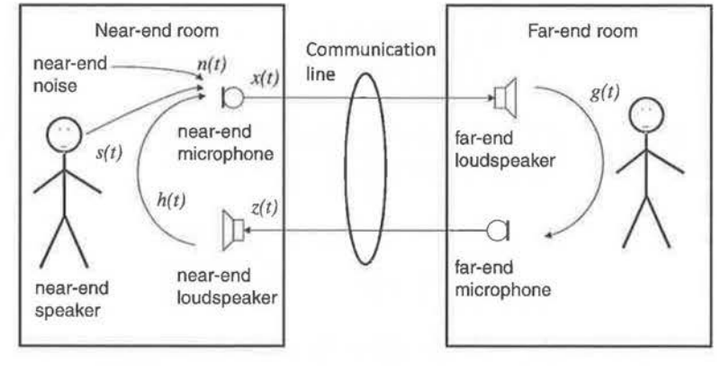


Figure 7.1 Speakerphone telecommunication system

The schematic diagram of the acoustic echo canceller is shown in Figure 7.2. The far-end signal $z(t)$ is sent to the loudspeaker. The microphone captures this signal convolved with the impulse response of the transfer path speaker-microphone $h(t)$. It captures the local voice $s(t)$ and noise $n(t)$. The transfer path speaker-microphone is

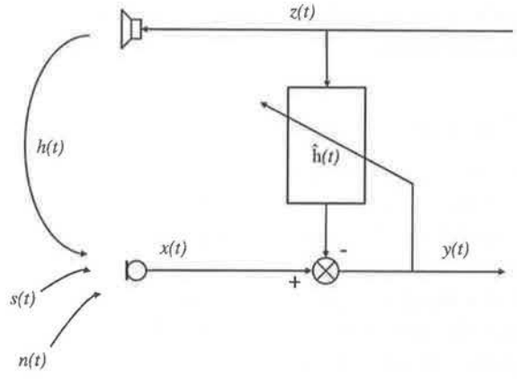


Figure 7.2 Schematic diagram of acoustic echo canceller

omitted for simplicity, as we will not deal with it in this chapter. The microphone signal is

$$x(t) = z(t)*h(t) + s(t) + n(t). \tag{7.1}$$

The acoustic echo canceller estimates the transfer path loudspeaker-microphone $\hat{h}(t)$ and subtracts the estimated portion of the loudspeaker signal from the microphone signal. At the acoustic echo canceller output we have

$$\begin{aligned} y(t) &= x(t) - z(t)*\hat{h}(t) \\ &= z(t)*h(t) - z(t)*\hat{h}(t) + s(t) + n(t). \end{aligned} \tag{7.2}$$

As usual we will illustrate the algorithms with processing in the frequency domain. Then the convolution converts to multiplication and we have

$$\begin{aligned} Y_k^{(n)} &= Z_k^{(n)} H_k^{(n)} - Z_k^{(n)} \hat{H}_k^{(n)} + S_k^{(n)} + N_k^{(n)} \\ &= Z_k^{(n)} (H_k^{(n)} - \hat{H}_k^{(n)}) + S_k^{(n)} + N_k^{(n)}. \end{aligned} \tag{7.3}$$

The modeling described so far assumes that the audio frame is longer than the reverberation process, which is incorporated in $h(t)$, and we model it with one tap filter for each frequency bin. The reverberation in a normal office or conference room lasts 200–400 ms to the moment that the reverberation energy goes below −60 dB. On the other hand, processing in the frequency domain and the overlap–add process, described in Chapter 2, adds one frame delay, increasing the latency of the entire system. A latency above 100 ms in communications is considered inconvenient for users, and a latency above 250 ms breaks the dialog. In most cases the overall latency of the

communication channel should be kept below 50 ms. This leaves less than 20 ms for the audio frame; in many cases the audio frame duration is 10 ms. At 16 kHz sampling rate this is a 160-sample frame length. To accommodate the longer impulse response, the acoustic echo canceller uses a finite impulse response (FIR) filter with multiple taps for each frequency bin. This converts Equation 7.3 to

$$\begin{aligned} Y_k^{(n)} &= \sum_{i=0}^{L-1} Z_k^{(n-i)} H_k^{(n-i)} - \sum_{i=0}^{L-1} Z_k^{(n-i)} \hat{H}_k^{(n-i)} Z_k^{(n)} + S_k^{(n)} + N_k^{(n)} \\ &= \sum_{i=0}^{L-1} Z_k^{(n-i)} \left(H_k^{(n-1)} - \hat{H}_k^{(n-1)} \right) + S_k^{(n)} + N_k^{(n)} \end{aligned} \tag{7.4}$$

where L is the number of taps in the FIR filter. Denoting

$$\begin{aligned} \mathbf{Z}_k^{(n)} &= [Z_k^{(n)}, Z_k^{(n-1)}, \ldots, Z_k^{(n-L+1)}]^T \\ \mathbf{H}_k^{(n)} &= [H_k^{(n)}, H_k^{(n-1)}, \ldots, H_k^{(n-L+1)}]^T \\ \mathbf{X}_k^{(n)} &= [X_k^{(n)}, X_k^{(n-1)}, \ldots, X_k^{(n-L+1)}]^T \end{aligned} \tag{7.5}$$

the equation can be rewritten in vector form:

$$Y_k^{(n)} = [\mathbf{H}_k^{(n)}]^T \mathbf{Z}_k^{(n)} - [\hat{\mathbf{H}}_k^{(n)}]^T \mathbf{Z}_k^{(n)} + S_k^{(n)} + N_k^{(n)}. \tag{7.6}$$

The total number of filter coefficients is $K \times L$, where K is the number of frequency bins and L is the number of taps for each frequency bin. The goal of the acoustic echo canceller is to estimate these coefficients as close as possible to the actual transfer function and to track eventual changes. If someone moves in the room, or a door opens, the transfer function between the loudspeaker and the microphone will change. This requires the use of adaptive filters in the acoustic echo cancellation.

7.1.3 Acoustic Echo Suppression

If we have perfect estimation of the transfer function, the signal captured from the loudspeaker will be completely suppressed. Owing to the near-end noise, shorter filters than the actual reverberation, and estimation errors, a portion of the captured loudspeaker signal will remain. This portion is called the *echo residual*:

$$\Re_k^{(n)} = Z_k^{(n)} (H_k^{(n)} - \hat{H}_k^{(n)}). \tag{7.7}$$

Assuming that the adaptive filtering did its best, whatever phase information is left behind will be very difficult to track. Then the way to reduce the residual is to use

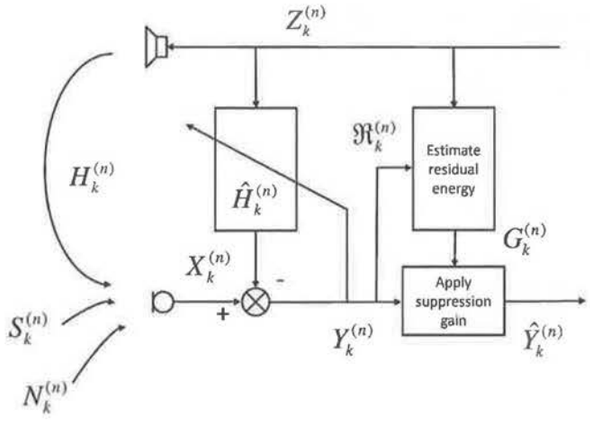


Figure 7.3 Acoustic echo canceller and acoustic echo suppressor

real-gain-based approaches similar to single-channel noise suppression (discussed in Chapter 4), as shown in Figure 7.3. If we can estimate the amount of residual energy $|\hat{\Re}_k^{(n)}|^2$ in each frequency bin, then we can estimate a real-valued suppression gain $G_k^{(n)}$ such as

$$\hat{Y}_k^{(n)} = G_k^{(n)} Y_k^{(n)} \approx S_k^{(n)} + N_k^{(n)} \tag{7.8}$$

that is optimal in one or another way. This technique is called *acoustic echo suppression*. The goal of the acoustic echo suppressor is to estimate the residual energy, to compute a suppression gain, and to apply it to the acoustic echo canceller output. It leads to improved suppression of the captured loudspeaker signal, but introduces distortion and artifacts typical of noise suppressors.

7.1.4 Evaluation Parameters

There is no way to optimize and tune the performance of any engineering system without specifying proper evaluation parameters. One commonly used parameter is the *system distance* [2]:

$$D^{(n)} = 10\log_{10}\left(\frac{||\mathbf{H}_k^{(n)} - \hat{\mathbf{H}}_k^{(n)}||^2}{||\mathbf{H}_k^{(n)}||^2}\right) \tag{7.9}$$

where $||\cdot||$ denotes the l^2-norm, $||\mathbf{H}_k^{(n)}||^2 = (\mathbf{H}_k^{(n)})^T \mathbf{H}_k^{(n)}$. This parameter measures the difference between the estimated and actual transfer functions. It is useful when the real transfer function is known – usually when the acoustic echo canceller is evaluated with simulated signals. A smaller difference means better estimation. In some sources the same parameter is called "normalized misalignment" [3]. This parameter is suitable only for evaluation of acoustic echo cancellers.

For evaluation of the residual energy, *echo return loss enhancement* (ERLE) is used. It is defined as

$$ERLE^{(n)} = 10\log_{10}\left(\frac{E\{(X_k^{(n)})^2\}}{E\{(\Re_k^{(n)})^2\}}\right) \tag{7.10}$$

and is the ratio of the signal and residual energies. The residual can be measured directly only if there is no local speech and the noise level is very low. Corrections for the noise energy should be made if this is not the case. A higher ERLE means better echo suppression.

One indirect parameter to measure the performance of an acoustic echo-reduction system is the perceptual sound quality. As discussed in Chapter 2, this is expressed with MOS points. Also, besides averaging the estimations from a large number of human listeners, the measurement can be done by using some of the algorithms for perceptual evaluation of sound quality. One of the standardized and frequently used algorithms is PESQ (perceptual evaluation of sound quality). It requires knowledge of the clean speech signal, which means that the voice of the local speaker should be recorded in parallel using a close-talk microphone. Another approach is a pre-recorded speech signal to be reproduced by either a loudspeaker of a head-and-torso simulator. The latter is better as it will create more realistic reverberation for the local speech.

An important part of the acoustic echo canceller evaluation is the *convergence time*. This is the time for estimation of the new filters $\hat{\mathbf{H}}_k^{(n)}$ after change in $\mathbf{H}_k^{(n)}$ or at the beginning of the process. As the convergence process may be considered complete when the residual reaches the noise level, then it will depend on the noise level and on the magnitude of the loudspeaker signal. To eliminate these dependencies, the converging process is often modeled with an exponential curve and a single parameter – the time constant.

7.2 LMS Solution for Acoustic Echo Cancellation

The block diagram in Figure 7.2 shows a classic application of an adaptive filter. At each audio frame the input signals are processed with the current filter and then the filter coefficients are updated. One of the potential goals is to minimize the least mean-square error – the classic LMS adaptive filter [4]. The gradient of the mean-square error is

$$\begin{aligned}\nabla^{(n)} &= \frac{\partial \mathrm{E}\{|\mathfrak{R}^{(n)}|\}}{\partial \mathbf{H}^{(n)}} \\ &= 2\mathrm{E}\left\{\mathfrak{R}^{(n)}\frac{\partial \mathfrak{R}^{(n)}}{\partial \mathbf{H}^{(n)}}\right\} \\ &= -2\mathrm{E}\{\mathfrak{R}^{(n)}\mathbf{Z}^{(n)}\}.\end{aligned} \tag{7.11}$$

Note that the frequency-bin index k is omitted for simplicity. Here we replaced the error with the residual, which means that the gradient estimation is correct when there is no near-end speech and local noise. With estimation of the momentary gradient

$$\hat{\nabla}^{(n)} = \mathfrak{R}^{(n)}\mathbf{Z}^{(n)} \tag{7.12}$$

we can update the filter coefficients in the frames with no local talk:

$$\mathbf{H}^{(n+1)} = \mathbf{H}^{(n)} + \mu\mathfrak{R}^{(n)}\mathbf{Z}^{(n)}. \tag{7.13}$$

Here, μ is the step size and determines the adaptation speed. In most applications the step size is variable to provide faster adaptation. To guarantee convergence of the filter it should be smaller than [4]:

$$0 < \mu < \frac{2}{\lambda_{\max}} \tag{7.14}$$

where $\lambda_{\max}$ is the largest eigenvalue of the input correlation matrix. The convergence time constant is given by

$$\tau_{\mathrm{conv}} = \frac{1}{2\mu\lambda_{\mathrm{av}}}. \tag{7.15}$$

In summary, the LMS acoustic echo canceller should adapt only during frames without local speech and can use variable step size to converge faster. We will continue to use the notation μ in this chapter, but in most of the cases it is variable and dynamically computed; that is, it is actually $\mu^{(n)}$. In the literature there are many algorithms for step-size control, suitable for acoustic echo cancellation, so they are not discussed in detail here.

EXERCISE

Find the .WAV files *FarEndMono.WAV* and *AEC_Mono.WAV*. The first is the loudspeaker signal, the second is recorded in normal noise and reverberation conditions

(a small office) without near-end speech. Modify the script *ProcessWAV.m* from Chapter 2 to have three input parameters: far-end file name, recorded file name, and output file name. Add reading of the second file, modify the conversion to the frequency domain to do conversion of the two files. Add the voice activity detector from Chapter 4 (*SimpleVAD.m*) to work on the far-end frames. Add the LMS adaptive filter to compensate the echo according to Equation 7.6, using $L = 10$. Do adaptation only when there is far-end speech (detected by the VAD) according to Equation 7.13. Evaluate the results by computing the ERLE (plot it as a function of time) and the convergence time. Adjust the adaptation speed by changing the value of μ. Save the script as *MonoAEC.m*.

Find a paper discussing dynamic step size and implement it. Compare the results.

7.3 NLMS and RLS Algorithms

The LMS adaptive filter has one known caveat. Assuming that the residual is proportional to the far-end signal, the adaptation rate will be proportional to the power of the far-end signal because it participates in the gradient estimation. To overcome this highly variable adaption speed, the *normalized least-mean-square* (NLMS) algorithm is preferred. It just adds normalization by the l^2 norm of the input vector:

$$\mathbf{H}^{(n+1)} = \mathbf{H}^{(n)} + \mu\frac{\mathfrak{R}^{(n)}\mathbf{Z}^{(n)}}{||\mathbf{Z}^{(n)}||^2}. \tag{7.16}$$

In real implementations a small number is added to the denominator to prevent division by zero. This is one of the most commonly used adaptive filters in acoustic echo cancellers.

NLMS adaptive filtering is well covered in the literature as well. Its convergence speed in the context of acoustic cancellers is critical for the quality of the cancellation and is well studied. Various modifications of the algorithm have been designed: proportionate NLMS (PNLMS) for better behavior with sparse impulse responses [5], and improved PNLMS (IPNLMS) for better convergence of the PNLMS algorithm [6].

Among other adaptive filter algorithms with application in acoustic echo cancellation should be mentioned the affine projection algorithm (APA), which is a further generalization of the NLMS [7]. The much faster adaptation comes at the cost of a substantial increase in computations, which led to creation of the fast affine projection (FAP) algoriothm [8].

The *recursive least-squares* (RLS) algorithm uses a recursive way to update the filter coefficients in each step and converges much faster than NLMS, but is computationally very expensive.

EXERCISE

Modify the script *MonoAEC.m* from the previous exercise to use the NLMS algorithm. Evaluate and compare the results with LMS.

Implement and compare with NLMS the performances of RLS, APA, and FAP algorithms.

7.4 Double-talk Detectors

The NLMS and other adaptive filters handle well the pauses in the far-end speech signal – the filter does not adapt when $\|\mathbf{Z}(n)\|^2 \rightarrow 0$. Still, in practical realizations of acoustic echo cancellers a voice activity detector (VAD) is used for the far-end signal and filters do not adapt when there is no far-end speech activity. The presence of near-end speech, however, can divert the adaptive filter to wrongly estimate the transfer function. A voice activity detector for the microphone signal can give an indication when there is far-end or near-end speech. To block the adaptation when there are both far-end and near-end speech signals we need a double-talk detector. The reaction of the adaptive acoustic echo canceller in both cases (no far-end speech or double talk) can be in a soft way; that is, instead of not adapting at all it can be implemented as reduction of the adaptation step size μ.

7.4.1 Principle and Evaluation

Double talk detectors (DTD) have the same evaluation parameters as those discussed in Chapter 4 for evaluation of voice activity detectors: true positive rate, false positive rate, and accuracy. The generic DTD computes a statistical parameter ξ, preferably data-independent, which is compared with a threshold η. If the value is higher than the threshold, double talk is detected; if it is below, there is no double talk. The threshold value can be adjusted using the ROC curves discussed in the same chapter. A good published paper about DTD evaluation criteria is [9].

Several improvements can be made to the classic comparison with a threshold:

- **Adding hysteresis**. Switch the state from "no double talk" to "double talk" when $\xi > \eta + \Delta\eta/2$, return to "no double talk" when $\xi < \eta - \Delta\eta/2$. Here, $\Delta\eta$ is the hysteresis and its value is adjusted together with the threshold η to be optimal in some way – best accuracy, minimal sum of the squares of false positives and false negatives, and so on. The hysteresis prevents frequent switching of the state when ξ is close to the threshold η.
- **Adding timing restrictions**. The speech signal has its own dynamics: probabilities to switch from pause to speech and from speech to pause, average duration of the speech segments, and so on. The simplest improvement is after switching to "double talk" state to stay there a certain minimal time.

Many DTD algorithms are developed for processing in the time domain. As we are here describing processing algorithms in the frequency domain, we will provide all equations using notation for that. Processing in the frequency domain uses L tap filters for each frequency bin, so the DTD algorithms designed for the time domain can be easily adapted to work for each frequency bin. The DTD output will be noisier owing to the shorter filter – in the time domain the number of taps is in the range of a couple of thousands, while in the frequency domain it is usually under ten. This is why the statistical parameter ξ, computed for each frequency bin (i.e., $\xi_k^{(n)}$), are combined to form the per-frame parameter $\xi^{(n)}$, which is compared with the threshold η. Combining is usually as a weighted sum:

$$\xi^{(n)} = \sum_{k=1}^{K} w_k \xi_k^{(n)}. \tag{7.17}$$

The weights are selected to be higher where there is more speech energy and lower where there is more noise. A typical shape of this frequency weighting is a band-pass filter in the range 200–3000 Hz. Some standardized weightings (C-message for example, see Chapter 3) can be used as well.

A good overview of various algorithms for double-talk detectors is given in Chapter 6 of [3]. When describing algorithms further we will omit the frequency bin indices whenever possible.

7.4.2 Geigel Algorithm

One of the earliest DTDs is the Geigel algorithm:

$$\xi^{(n)} = \frac{\max\{|\mathbf{X}^{(n)}|\}}{|Z^{(n)}|}. \tag{7.18}$$

This evaluates the ratio of the largest magnitude of the microphone signal $\mathbf{X}^{(n)}$ (see Equation 7.5) for the last L frames to the magnitude of the far-end speech $Z^{(n)}$. It assumes that the near-end speech is typically stronger in the microphone signal. The number of evaluated previous values is usually assumed the same as the length of the adaptive filter. This algorithm was designed for network echo cancellers where it works best. For acoustic echo cancellers the variability of the optimal threshold is higher and the Geigel algorithm works less reliably.

7.4.3 Cross-correlation Algorithms

Cross-correlation function based DTDs are considered more robust and reliable. The cross-correlation vector of $\mathbf{X}^{(n)}$ and $Z^{(n)}$ is

$$\mathbf{C}_{XZ} = \frac{\mathrm{E}\{\mathbf{X}^{(n)}Y^{(n)}\}}{\sqrt{\mathrm{E}\{|\mathbf{X}^{(n)}|^2\}\mathrm{E}\{|Y^{(n)}|^2\}}} = \frac{\mathbf{R}_{XZ}}{\sigma_X\sigma_Z}. \tag{7.19}$$

The statistical variable for comparing with a threshold can be either the l^κ-norm ($\kappa = 1, 2, \ldots$) of the correlation vector or the maximal value for the last L frames:

$$\begin{aligned}\xi_{\mathrm{CC}}^{(n)} &= ||\mathbf{C}_{XZ}||^\kappa \\ &= \max\{\mathbf{C}_{XZ}\}.\end{aligned} \tag{7.20}$$

The problem with this algorithm is that the cross-correlation function is not very well normalized. It is not quite robust when near-end noise is present. The *normalized cross-correlation* method is derived in [10]. In the absence of near-end speech and noise:

$$\sigma_X^2 = \mathbf{H}^{\mathrm{T}}\mathbf{R}_{ZZ}\mathbf{H} \tag{7.21}$$

where $\mathbf{R}_{ZZ} = E\{\mathbf{Z}^{(n)}(\mathbf{Z}^{(n)})^{\mathrm{T}}\}$. Since $X^{(n)} = \mathbf{H}^T\mathbf{Z}^{(n)}$, then $\mathbf{R}_{XZ} = \mathbf{R}_{ZZ}\mathbf{H}$ and Equation 7.21 can be rewritten in the form

$$\sigma_X^2 = \mathbf{R}_{ZX}^T\mathbf{R}_{ZZ}^{-1}\mathbf{R}_{ZX}. \tag{7.22}$$

When we have near-end noise and speech present, this converts to

$$\sigma_Y^2 = \mathbf{R}_{ZX}^T\mathbf{R}_{ZZ}^{-1}\mathbf{R}_{ZX} + \sigma_V^2, \tag{7.23}$$

where $V = S + N$. The statistical parameter ξ for the DTD is the square-root of (7.22) divided by (7.23):

$$\begin{aligned}\xi_{\mathrm{NCC}} &= \frac{\sqrt{\mathbf{R}_{ZX}^T\mathbf{R}_{ZZ}^{-1}\mathbf{R}_{ZX}}}{\sqrt{\mathbf{R}_{ZX}^T\mathbf{R}_{ZZ}^{-1}\mathbf{R}_{ZX} + \sigma_V^2}} \\ &= \sqrt{\mathbf{R}_{ZX}^T(\sigma_X^2\mathbf{R}_{ZZ})^{-1}\mathbf{R}_{ZX}} \\ &= ||C_{ZX}||^2.\end{aligned} \tag{7.24}$$

Here, $C_{ZX} = (\sigma_X^2\mathbf{R}_{ZZ})^{-1/2}\mathbf{R}_{ZX}$ is the normalized cross-correlation function.

The DTD as described can be computationally expensive. Later, a faster version of this algorithm was developed [11]. It is based on recursively updating $\mathbf{R}_{ZZ}^{-1}\mathbf{R}_{ZX}$ using the Kalman gain $\mathbf{R}_{ZZ}^{-1}\mathbf{Z}$. Then Equation 7.24 can be rewritten as

$$\xi_{\mathrm{FNCC}}^2 = \frac{\mathbf{R}_{ZX}^T\mathbf{R}_{ZZ}^{-1}\mathbf{R}_{ZX}}{\sigma_Y^2(n)} = \frac{\chi^2(n)}{\sigma_Y^2(n)} \tag{7.25}$$

where the statistic parameter is squared for simplicity. Then for each frame the estimation of the correlation variables is as follows:

$$\begin{aligned}\sigma_X^2(n) &= \lambda\sigma_X^2(n-1) + |X^{(n)}|^2 \\ \chi^2(n) &= \lambda\chi^2(n-1) + |X^{(n)}|^2 - \varphi(n)|\Re^{(n)}|^2 \\ \Re^{(n)} &= X^{(n)} - \mathbf{H}^{\mathrm{T}}\mathbf{Z}^{(n)} \\ \varphi(n) &= \frac{\lambda}{\alpha(n)} \\ \alpha(n) &= \lambda + (\mathbf{Z}^{(n)})^{\mathrm{T}}(\mathbf{R}_{ZZ}^{-1}(n-1))\mathbf{Z}^{(n)} \\ \mathbf{R}_{ZZ}^{(n)} &= \lambda\mathbf{R}_{ZZ}^{(n-1)} + \mathbf{Z}^{(n)}(\mathbf{Z}^{(n)})^{\mathrm{T}}.\end{aligned} \tag{7.26}$$

Here, λ is a forgetting factor. These two methods are among the most frequently used algorithms for double-talk detection.

7.4.4 Coherence Algorithms

Instead of using the cross-correlation function as a statistical variable we can use the squared magnitude of the coherence function [12]. If the coherence between $\mathbf{Z}^{(n)}$ and $\mathbf{X}^{(n)}$ is close to 1, then there is no double talk; if it goes below a certain threshold, then there is double talk. The squared magnitude of the coherence function for the frequency bin k is

$$\gamma_{ZX}^2(k) = \frac{|S_{ZX}(k)|^2}{S_{ZZ}(k)S_{XX}(k)}. \tag{7.27}$$

The statistics function then can be used per bin or as a weighted average of all frequency bins for a per-frame decision:

$$\xi_{\mathrm{COH}} = \sum_{k=1}^{K} w_k\gamma_{ZX}^2(k). \tag{7.28}$$

The weighting is usually by a band-pass filter in the range 200–2000 Hz with smooth slopes – see the beginning of this section.

EXERCISE

Copy the script *MonoAEC.m* from the previous exercise as *MonoAEC_DTDeval.m*. Add the double-talk detector. Implement all four algorithms from this subsection. Find and use for evaluation files *FarEndMono.WAV*, *AEC_Mono_wDoubleTalk.WAV*, and *NearEndMono.WAV*.

The first is the loudspeaker signal, the second is recorded in normal noise and reverberation conditions (small office) with near-end speech, and the third is a clean

version of the near-end speech. The second file is recorded with near-end speech played by a head-and-torso simulator.

Add the near-end speech as a fourth parameter; add reading and conversion to the frequency domain for the near-end speech. Add a second simple VAD to work on the near-end speech. We have double talk when both VADs (far- and near-end) indicate speech activity. Compare this with the output of the DTD above. Build a table comparing the true positives, false negatives, and accuracies of the four algorithms. Use ROC curves to find the best thresholds for each algorithm.

Add the best DTD to the script *MonoAEC.m* for further use. Use the NLMS algorithm with a variable step size. Modify it to adapt only when there is far-end speech and no double talk. At this point you should have a decent MATLAB® implementation of a mono acoustic echo canceller. Evaluate the ERLE and convergence time.

7.5 Non-linear Acoustic Echo Cancellation

7.5.1 Non-linear Distortions

The adaptive filter assumes a linear transfer function between the far-end signal and the microphone. If the loudspeaker is not perfect (none is, but it is more valid for small loudspeakers) it will introduce non-linear distortions. Another potential source of non-linear distortions is clipping in the output amplifier. The effect of these distortions is that the reproduction tract adds harmonics of the far-end signal. The measure for harmonic distortions is called *total harmonic distortion* (THD) and is defined as the proportion of the power of all harmonics to the power of the first harmonic:

$$THD = \frac{\sum_{i=2}^{N} A_i^2}{A_1^2} = \frac{A_{\text{RMS}}^2 - A_1^2}{A_1^2} \tag{7.29}$$

Here, A_1 is the amplitude of the sinusoidal signal sent to the loudspeaker, and A_i is the amplitude of the i-th harmonic that appears because of the non-linear distortions. This means that the microphone will capture signals (the harmonics) for which the acoustic echo canceller is not set up because they are not in the far-end signal. A small loudspeaker can have 10% THD at maximal power. This means that at least 10% of the echo energy, captured by the microphone, will not be cancelled, which is limiting factor to the performance of the acoustic echo canceller. ITU-T standards require reduction of the acoustic echo by at least 30 dB, which cannot be achieved even with a perfect echo canceller if the loudspeaker introduces more than 3% harmonic distortion. The quality requirements for loudspeakers used in systems employing acoustic echo cancellation are usually higher, but there are algorithmic ways to mitigate this problem as well. Birkett and Goubran [13] use a neural network and a second microphone (to provide an error signal) to achieve considerable improvement. With a very simple

delay and saturation model, Stenger and Rabenstein [14] achieve ERLE improvement of 4 dB.

Practically all non-linear AEC algorithms proposed in the literature work in the time domain. Unfortunately their conversion to the frequency domain is not trivial owing to their complex structure. In the following subsections the notation is changed: $x[k]$ is the far-end signal and $y[k]$ is the microphone signal at moment kT.

7.5.2 Non-linear AEC with Adaptive Volterra Filters

Volterra series is a model for non-linear behavior, similar to the Taylor series. The difference is that with a Taylor series the output at any given moment depends only on the input at that moment, while in the Volterra series the output depends on the input at all times. This "memory" effect allows modeling of complex non-linear systems, containing capacitors and inductances. Initially defined with integrals, the Volterra series can be converted in discrete form and pruned to order M:

$$y[k] = \sum_{r=0}^{N} \sum_{k_1=0}^{M} \cdots \sum_{k_r=k_{r-1}}^{M} h_r[k_1, \cdots, k_r] x[k-k_1] \cdots x[k-k_r], \tag{7.30}$$

where h_r are r-th-order Volterra kernels. As the Volterra kernels are symmetric, in Equation 7.30 only coefficients $k_r \geq k_{r-1}$ are used. An acoustic echo-cancellation algorithm using a second-order adaptive Volterra filter is proposed by Stenger *et al.* [15]. Defining

$$\begin{aligned} \mathbf{x}_1[k] &= (x[k], x[k-1], \cdots, x[k-M+1]) \\ \hat{\mathbf{h}}_1 &= (\hat{h}_1[0], \hat{h}_1[1], \cdots, \hat{h}_1[M-1]) \end{aligned} \tag{7.31}$$

for the first-order and

$$\begin{aligned} \mathbf{x}_2[k] = (&x^2[k], x[k]x[k-1], \cdots, x[k]x[k-M+1], x^2[k-1], x[k-1]x[k-2], \cdots, \\ &x[k]x[k-M+1], \cdots x[k-M+1]x[k-M+1]) \\ \hat{\mathbf{h}}_2 = (&\hat{h}_2[0,0], \hat{h}_2[0,1], \cdots, \hat{h}_1[0,M-1], \hat{h}_2[1,1], \cdots, \hat{h}_1[M-1,M-1]) \end{aligned} \tag{7.32}$$

for the second-order Volterra kernel, the LMS adaptive Volterra filter is defined as

$$r[k] = y[k] - \hat{\mathbf{h}}_1[k]\mathbf{x}_1^T[k] - \hat{\mathbf{h}}_2[k]\mathbf{x}_2^T[k] \tag{7.33}$$

$$\hat{\mathbf{h}}_1[k+1] = \hat{\mathbf{h}}_1[k] + \mu_1 r[k]\mathbf{x}_1^T[k] \tag{7.34}$$

$$\hat{\mathbf{h}}_2[k+1] = \hat{\mathbf{h}}_2[k] + \mu_2 r[k]\mathbf{x}_2^T[k]. \tag{7.35}$$

This can easily be converted to an NLMS algorithm by normalizing the step size:

$$\mu_1 = \frac{\alpha_1}{||\mathbf{x}_1[k]||^2} \qquad \mu_2 = \frac{\alpha_2}{||\mathbf{x}_2[k]||^2} \tag{7.36}$$

where α_1 and α_2 are the step-size parameters. To reduce computational complexity, the authors of the paper propose using different lengths of first- and second-order filters, $M_1 = 50$ taps and $M_1 = 25$ taps. With this second-order Volterra filter they report an ERLE improvement of 5.5 dB compared to a linear NLMS adaptive filter. This algorithm is applicable for systems working in the time domain. For systems using more processing steps (usually in the frequency domain) it does not fit well in the overall architecture.

7.5.3 Non-linear AEC Using Orthogonalized Power Filters

The non-linear transfer function can be modeled with power filters. These filters represent the output signal as a linear combination of a certain number of samples of the input signal (what linear filters do) and their square, their third power, and so on; that is, using the Taylor series. A power filter of P-th order is shown in Figure 7.4 and defined as follows:

$$y[k] = \sum_{p=1}^{P} \sum_{l=0}^{N-1} h[p,l] x^p[k-l] = \sum_{p=1}^{P} \mathbf{h}_p^{\mathrm{T}} \mathbf{x}_p[k] \tag{7.37}$$

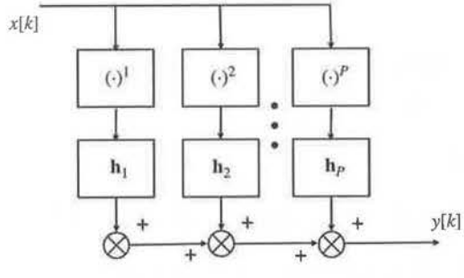


Figure 7.4 Power filter

where the vectors are defined as

$$\begin{aligned} \mathbf{x}_p[k] &= [x^p[k], x^p[k-1], \ldots, x^p[k-N+1]]^T \\ \mathbf{h}_p &= [h_{p,0}, h_{p,1}, \ldots, h_{p,N-1}]^T. \end{aligned} \tag{7.38}$$

Direct adaptation of the proposed power filter structure will be slow owing to the high correlation of the input signals (i.e., $x[k], x^2[k], \ldots x^P[k]$) for these filters working

in parallel. To improve the convergence speed, Kuech *et al.* [16] introduce a new set of mutually orthogonal input signals:

$$\begin{aligned} x_{o,1}[k] &= x[k] \\ &\ldots \\ x_{o,p}[k] &= x^p[k] + \sum_{i=1}^{p-1} q_{p,i} x^i[k] \end{aligned} \tag{7.39}$$

for $1 < p < P$. The orthogonalization coefficients $q_{p,i}$ are chosen such that

$$\mathrm{E}\{x_{o,i}[k] x_{o,j}[k]\} = 0, \quad \text{for} \quad i \neq j \tag{7.40}$$

and determined using the Gram–Schmidt orthogonalization method [17]. After modifying the filter structure and using bias correction, the authors of [16] present experimental results showing improvement in ERLE. This method has the same problem as the previous one – it fits with difficulty in a large signal processing system operating in the frequency domain.

7.5.4 Non-linear AEC in the Frequency Domain

For acoustic echo cancellers working in the frequency domain the compensation for non-linearity of the loudspeaker is more complex. An interesting algorithm is proposed by Bendersky *et al.* [18]. After the linear AEC is placed a block that adaptively estimates the magnitudes of harmonics for each frequency bin and uses suppression methods to reduce the echo residual magnitude. This naturally leads us to the next section.

7.6 Acoustic Echo Suppression

The acoustic echo suppressor usually follows the acoustic echo canceller. Assuming that the adaptive filtering has already cancelled the trackable part of the echo signal, whatever phase information is left behind will be very difficult to estimate. This is why the next step is to remove the residual by using suppression techniques described in Chapter 4. The problem remains: we have a mixture of statistically independent signals (echo residual, local speech, and local noise). The goal is to estimate a real-valued suppression gain which, applied to the output of the acoustic echo canceller, suppresses the residual and lets the local speech pass undistorted.

7.6.1 Estimation of the Residual Energy

In noise suppressors, one of the most important components was building the noise model – that is, the noise variance for each frequency bin, $\lambda_d(k)$. It is estimated during the pauses of the speech signal, indicated by a voice activity detector (VAD). We assumed that the noise is almost stationary, so that the average noise power in each

frequency bin changes much more slowly than the speech power. In acoustic echo suppressors, the equivalent of the noise model is estimation of the residual power. None of the assumptions for the noise above is valid. We can only assume that the residual power is a function of the far-end speech. We should estimate this function during frames when we have far-end speech (using a VAD on the far-end signal) and when there is no double talk.

Enzner *et al.* [19] propose using the coherence function for estimation of the residual energy. Given an acoustic echo-canceller output as a sum of the statistically independent near-end speech, noise, and AEC residual $Y_k^{(n)} = S_k^{(n)} + N_k^{(n)} + \Re_k^{(n)}$, then the squared coherence function is

$$C_{ZY}(k,n) = \frac{|\Phi_{ZY}(k,n)|^2}{\Phi_{ZZ}(k,n)\Phi_{YY}(k,n)} \tag{7.41}$$

and we can estimate the power spectral density of the residual energy as

$$\Phi_{\Re\Re}(k,n) = C_{ZY}(k,n)\Phi_{ZZ}(k,n). \tag{7.42}$$

This method uses a single tap filter in the frequency domain and underestimates the residual echo because the reverberation takes longer than the acceptable frame size. The authors generalize the residual energy estimation as

$$\Phi_{\Re\Re}(k,n) = \sum_{i=0}^{L-1} C_{ZY}(k,n-i)\Phi_{ZZ}(k,n-i). \tag{7.43}$$

The derivations in Equations 7.41 and 7.42 are valid under the assumption of uncorrelated echo and background noise signals, which is true in the long term. In the short term, in one frame, they are correlated and Equation 7.43 will overestimate the residual power. The authors propose a technique to compensate for the bias, which requires additional computational resources.

Instead of increasing the complexity of the model, Chhetri *et al.* [20] propose a direct regression model:

$$|\hat{\Re}_k^{(n)}| \approx \sum_{i=0}^{L-1} w_i |Z_k^{(n-i)}|. \tag{7.44}$$

On squaring Equation 7.44 we have

$$\begin{aligned} |\hat{\Re}_k^{(n)}|^2 &\approx \left(\sum_{i=0}^{L-1} w_i |Z_k^{(n-i)}|\right)^2 \\ &= \sum_{i=0}^{L-1}\sum_{j=0}^{L-1} w_i w_j |Z_k^{(n-i)}||Z_k^{(n-j)}| \end{aligned} \tag{7.45}$$

which, according to the authors, is more powerful as it contains the cross-power terms that are missing in the power regression model

$$|\hat{\Re}_k^{(n)}|^2 \approx \sum_{i=0}^{L-1} w_i |Z_k^{(n-i)}|^2 \tag{7.46}$$

close to the estimation based on power spectral density in Equation 7.43. To compute the regression coefficients, the authors propose an adaptive algorithm that assumes knowledge of the noise magnitude. In each frame the residual magnitude is estimated according to Equation 7.44. Then the error signal and smoothed far-end power are computed:

$$E_k^{(n)} = \max(|Y_k^{(n)}| - \Re_k^{(n)}, N_k^{(n)}) \tag{7.47}$$

$$P_k^{(n)} = \alpha P_k^{(n-1)} + (1-\alpha)||Z_k^{(n)}||^2. \tag{7.48}$$

The adaptation happens after computing the normalized gradient:

$$\begin{aligned} \nabla_k^{(n)} &= -\frac{2E_k^{(n)}|\mathbf{Z}_k^{(n)}|}{P_k^{(n)}} \\ \mathbf{w}_k^{(n+1)} &= \mathbf{w}_k^{(n)} - \frac{\mu}{2}\nabla_k^{(n)}. \end{aligned} \tag{7.49}$$

Here, $\mathbf{w}_k^{(n)}$ is the vector of regression coefficients from Equation 7.44. The adaptation is performed only in the absence of a near-end speech signal and the presence of a far-end speech signal. The authors use a variable step size μ, adjustable to ensure the positivity of $|Y_k^{(n)}| - \Re_k^{(n)}$ as much as possible, regardless of preventing $E_k^{(n)}$ to fall below the noise floor in Equation 7.47. The number of regression coefficients varies with the frequency and the room size.

7.6.2 *Suppressing the Echo Residual*

The suppression is based on estimation and applying a real-valued and time-varying suppression gain, usually between 0 and 1. The most straightforward approach is the Wiener suppression rule

$$V_k^{(n)} = \frac{\max(||Y_k^{(n)}||^2 - ||\hat{\Re}_k^{(n)}||^2, 0)}{||Y_k^{(n)}||^2} Y_k^{(n)} \tag{7.50}$$

with all the caveats discussed in Chapter 4. Many of the suppression rules from this chapter can be adapted and used for suppressing the echo residual.

An interesting approach is proposed by Madhu *et al.* [21]. They use an EM learning algorithm to estimate directly the probability of the presence of a residual signal. This probability is used as a suppression rule, in the same way as described in the sections on probability-based suppression rules. Besides the traditional ERLE as quality measurement, PESQ MOS is used. The paper reports better echo suppression and better perceptual sound quality than the regression AES.

The presence of near-end noise affects estimation of the echo residual and a noise model should be estimated, as described in the previous subsection. Most sound capture systems include a stationary noise suppressor immediately after the acoustic echo-reduction block. This justifies merging the noise and echo suppressors. Then the Wiener gain should be

$$H_k^{(n)} = \frac{||Y_k^{(n)}||^2 - ||\hat{\Re}_k^{(n)}||^2 - ||N_k^{(n)}||}{||Y_k^{(n)}||^2}. \tag{7.51}$$

Most of the a-priori and a-posteriori SNR estimators can be adapted and more sophisticated suppression rules used. Good results are achieved using probability as the suppression rule:

$$H_k^{(n)} = \frac{P_S^{(n)} p(S_k^{(n)}|\mu_Y, \sigma_Y^2)}{P_S^{(n)} p(S_S^{(n)}|\mu_Y, \sigma_Y^2) + P_N^{(n)} p(N_k^{(n)}|\mu_N, \sigma_N^2) + P_{\Re}^{(n)} p(\hat{\Re}_k^{(n)}|\mu_{\Re}, \sigma_{\Re}^2)} \tag{7.52}$$

where $P_X^{(n)}$ is the prior probability for the presence of X in the n-th frame, and $p(X|\mu, \sigma)$ is the PDF value for X given the mean and variance and know distribution. Practically this is a faster way to compute the probability than described in [21].

Overall, AES is an important part in acoustic-reduction systems. It provides 5–10 dB additional echo suppression. When it is well tuned, most of the artifacts, typical for every suppression algorithm, can be negligible. In such cases AES actually increases the perceptual sound quality. One more important thing – the AES is the safety net for the acoustic echo canceller. When the linear adaptive filter is not converged after rapid change in the transfer function, AES should be able to adapt faster and suppress the increased echo residual. The speech signal is quite sparse and the probability of having far- and near-end speech in the same frequency bin is relatively low. This means that, even under double-talk conditions, the quality of the output signal should be acceptable and in all cases better than no echo suppression at all.

EXERCISE

Copy the script *MonoAEC.m* from the previous exercise as *MonoAEC_AES.m*. Implement the two algorithms for estimation of the echo residual presented above.

Use the files with double talk. Compare the results using the same suppression rule. Select the better algorithm by comparing ERLE and performing listening tests. Not always does higher suppression mean better listening results. Adapt and implement some of the suppression rules from Chapter 4 and select the one that gives best results.

Experiment with an echo-reduction system with AES only by disabling the AEC part. Evaluate the results: echo suppression, and quality of the near-end sound.

Copy the script *MonoAEC.m* as *MonoAER.m* and add the best AES. At this point you should have a good working MATLAB implementation of a mono acoustic-reduction system.

7.7 Multichannel Acoustic Echo Reduction

7.7.1 The Non-uniquenes Problem

Most of communication systems operate with a mono far-end signal. Building a high-end telecommunication system with stereo sound would allow more comfortable communication and better perception of the positions of the different sound sources in the far-end room. Attempts to design a stereophonic acoustic echo canceller started in the early 1990s [22]. The first results were unsatisfactory and this raised the interest of the signal processing community. There followed many efforts to study the problem and offer solutions. At the beginning of the next decade there emerged scenarios such as controlling the stereo and surround-sound equipment with human voice and speech recognition, which requires stereo and a multichannel acoustic echo-reduction system.

When we refer to a stereo and multichannel acoustic echo-reduction system we mean highly correlated speaker channels. The most intuitive approach is to build the stereo acoustic echo canceller by chaining two mono units, as shown in Figure 7.5. This scheme will work flawlessly if the two loudspeaker channels are not correlated. Each of the filters will adapt independently from the other. Unfortunately this is not the case

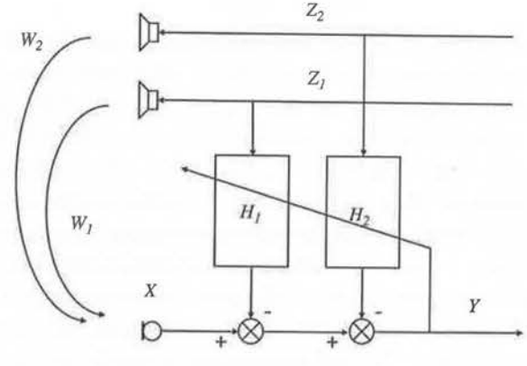


Figure 7.5 Stereo acoustic echo canceller with two mono acoustic echo cancellers

with stereo sound where the two channels are highly correlated. The problem is well described in [23].

Assume for a moment that we have a frame long enough to handle the reverberation; that is, we have one tap filter for each frequency bin (omitted in this section for simplicity). Then the sound source S in the far-end room is captured by the two microphones as follows:

$$\begin{aligned} Z_1 &= G_1 S \\ Z_2 &= G_2 S. \end{aligned} \tag{7.53}$$

Here, G_1 and G_2 are the transfer functions from the sound source to each of the microphones in the far-end room. These two channels are reproduced by the speakers in the near-end room and captured by one of the microphones (no near-end speech and noise presented):

$$X = W_1 Z_1 + W_2 Z_2. \tag{7.54}$$

Here, W_1 and W_1 are the transfer functions from each of the loudspeakers to the microphone. We apply two acoustic echo cancellers with filters H_1 and H_2 and have on the output

$$\begin{aligned} Y &= W_1 Z_1 + W_2 Z_2 - H_1 Z_1 - H_2 Z_2 \\ &= W_1 G_1 S + W_2 G_2 S - H_1 G_1 S - H_2 G_2 S \end{aligned} \tag{7.55}$$

which, considering that $S \neq 0$ and completely converged canceller, leads to the equations

$$\begin{aligned} W_1 G_1 + W_2 G_2 - H_1 G_1 - H_2 G_2 &= 0 \\ G_1 (W_1 - H_1) + G_2 (W_2 - H_2) &= 0. \end{aligned} \tag{7.56}$$

The first thing to note is that we have two unknowns and one equation, which leads to an infinite number of solutions when $G_1 \neq 0$ and $G_2 \neq 0$, which is true if we have stereo sound capture. The two adaptive filters can converge to any of the infinite number of solutions. Unfortunately all of them depend on G_1 and G_2, except one: $W_1 = H_1$ and $W_2 = H_2$, which is the "true" solution – each adaptive filter converged to the corresponding transfer function. Any other solution is correct for the current situation, but changes in the far-end room – the speaker moves or another speaker starts to talk – will cause loss of convergence, appearing as echo on the output and the two adaptive filters will have to re-converge again. This is called the "non-uniqueness" problem.

In general, two adaptive filters, working in parallel, perform poorly if their input signals are highly correlated. Several approaches to create stereo and multichannel acoustic echo-reduction systems are discussed next.

7.7.2 *Tracking the Changes*

The general idea is to keep the structure in Figure 7.5 and to build an acoustic echo canceller that converges fast enough to track the changes in the far-end room, including changes of speaker position. This requires computationally complex adaptive filters such as RLS. Additional measures are taken to stabilize the convergence process. One example of such an approach is presented in [24].

7.7.3 *Decorrelation of the Channels*

The non-uniqueness problem occurs because the loudspeaker channels are highly correlated. If there is a way to de-correlate them, the structure with two adaptive filters in parallel will work. This can be achieved by adding non-linear elements, different for each channel, as shown in Figure 7.6. This idea is proposed in [25]. Unfortunately, to achieve stable working of the stereo acoustic echo canceller it is necessary to add a level of non-linear distortions, which are clearly audible and objectionable in high-end communication systems. Other attempts to decorrelate loudspeaker signals by adding uncorrelated noise to each channel, using comb filtering, using time-varying filters, and so on, either destroy the stereo picture or introduce unacceptable distortions and delays.

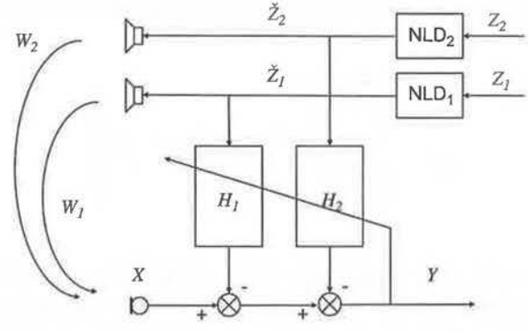


Figure 7.6 Stereo acoustic echo canceller with non-linear distortion of the far-end signal

The idea is developed further by Herre *et al.* [26], whereby the non-linear distortions are introduced accounting for psychoacoustics of human hearing. The authors achieve enough decorrelation to allow adaptive filters to converge in a surround-sound echo-cancellation system. On the other hand, their user studies show that human listeners cannot hear these distortions. The paper is a good overview of the approaches for multichannel acoustic echo cancellation and provides many literature references.

7.7.4 Multichannel Acoustic Echo Suppression

The energy transfer function is much less affected by correlation of the loudspeaker channels. Considering the fact that the speech signal is sparse in both time and frequency domains, building a stereo acoustic echo canceller based entirely on methods of suppression may not be such a bad idea as it seems at first look. Faller and Tournery [27] model the echo path as a delay and a single tap coloration filter for each frequency bin. The delay is computed and introduced before converting to the frequency domain. The coloration filter is estimated as

$$
\begin{aligned}
w_k^{(n)} &= \left| \frac{\mathrm{E}\{Z_k^{(n)^*} X_k^{(n)}\}}{\mathrm{E}\{X_k^{(n)^*} X_k^{(n)}\}} \right| \\
E\{X_k^{(n)^*} X_k^{(n)}\} &= \frac{T}{\tau} |X_k^{(n)^*} X_k^{(n)}| + \left(1 - \frac{T}{\tau}\right) E\{X_k^{(n-1)^*} X_k^{(n-1)}\} \\
E\{Z_k^{(n)^*} X_k^{(n)}\} &= \frac{T}{\tau} |Z_k^{(n)^*} X_k^{(n)}| + \left(1 - \frac{T}{\tau}\right) E\{Z_k^{(n-1)^*} X_k^{(n-1)}\}.
\end{aligned}
\tag{7.57}
$$

Here, T is the frame duration and τ is the adaptation time constant. The authors propose $\tau = 1.5$ s. Then the estimations of the echo residual and the suppression gain are

$$
\begin{aligned}
\hat{\Re}_k^{(n)} &= w_k^{(n)} |Z_k^{(n)}| \\
H_k^{(n)} &= \left[\frac{\max(|Z_k^{(n)}|^\alpha - \beta |\hat{\Re}_k^{(n)}|^\alpha, 0)}{|Z_k^{(n)}|^\alpha} \right]^{\frac{1}{\alpha}}.
\end{aligned}
\tag{7.58}
$$

Here, α and β are design parameters. If $\alpha = 2$ the formula converts to a spectral subtraction suppression rule; $\beta < 1$ is used if the echo is underestimated, $\beta > 1$ otherwise. In the multichannel case (in the paper are discussed multiple reproduction and capture channels), the single-channel AES described above is used. The microphone and speaker energies are combined as follows:

$$
\begin{aligned}
|Z_k^{(n)}| &= \left(\sum_{l=1}^{L} g_{Zl} |Z_k^{(n)}(l)|^\theta \right)^{\frac{1}{\theta}} \\
|X_k^{(n)}| &= \left(\sum_{m=1}^{M} g_{Zm} |X_k^{(n)}(m)|^\lambda \right)^{\frac{1}{\lambda}}.
\end{aligned}
\tag{7.59}
$$

The authors use weightings $g_{Zl} = g_{Xm} = 1$ for all l and m, and $\theta = \lambda = 2$, which means that all loudspeaker and microphone channels are treated equally and Equation 7.59 combines the channel powers. Then the gain computed in Equation 7.58 is applied to all microphone channels.

7.7.5 Reducing the Degrees of Freedom

Looking at Equation 7.56 again, we can conclude that if we have one equation then we should use a single adaptive filter. A similar idea is proposed by Hirano and Sugiyama [28]. The stereophonic acoustic cancellation is for telecommunications and assumes that the sound source is captured by the two microphones according to Equation 7.53. The adaptive filters for both channels use the same input signal – the speaker channel which arrives earlier. One microphone channel of the proposed stereo echo-canceller structure is shown in Figure 7.7. Then Equation 7.55 changes to

$$
\begin{aligned}
Y &= W_1 Z_1 + W_2 Z_2 - H Z_1 \\
&= W_1 G_1 S + W_2 G_2 S - H G_1 S.
\end{aligned}
\tag{7.60}
$$

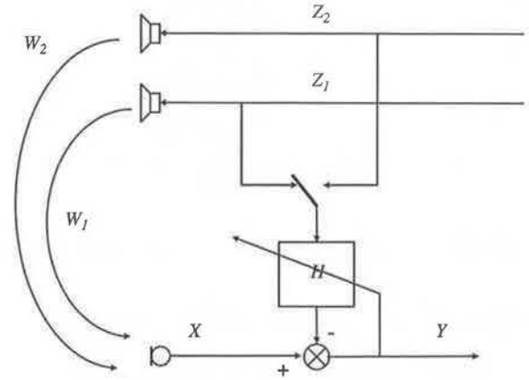


Figure 7.7 Stereo acoustic echo canceller with one adaptive filter

Assuming a converged filter and $S \neq 0$, Equation 7.56 in this case will look like

$$
\begin{aligned}
W_1 G_1 + W_2 G_2 - H G_1 &= 0 \\
H &= (W_1 G_1 + W_2 G_2) G_1^{-1}.
\end{aligned}
\tag{7.61}
$$

The filter will have a solution and will converge if the far-end room impulse response is invertible. While the assumption $S \neq 0$ is safe, as we can adapt the filter only when it is met (i.e., we have speech activity at the far end) and the situation is in our control, this

is not the case with the far-end room impulse response, which may not be invertible. Another caveat of this structure is that if the far-end speaker changes (rapid change of G_1 and G_2) the stereo echo canceller will have to reconverge.

To resolve these issues, the adaptive filter architecture shown in Figure 7.8 is proposed. Assume for a moment that we have good initial estimations of W_1 and $W_2 - H_{01}$ and H_{02}, respectively. Equation 7.55 with this structure looks like

$$\begin{aligned} Y &= W_1Z_1 + W_2Z_2 - H(H_{01}Z_1 + H_{02}Z_2) \\ &= W_1G_1S + W_2G_2S - H(H_{01}G_1S + H_{02}G_2S) \end{aligned} \tag{7.62}$$

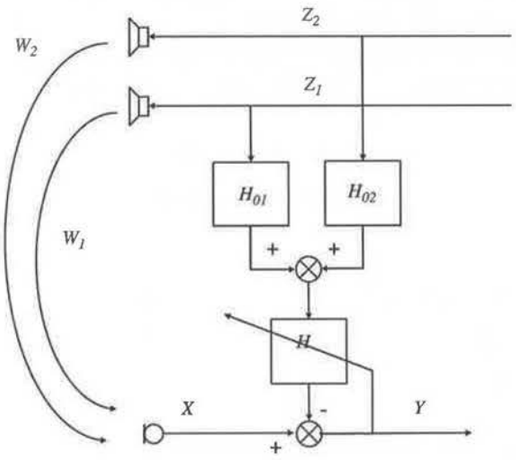


Figure 7.8 Stereo acoustic echo canceller with two fixed and one adaptive filters

and initially the adaptive filter is converged at $H = 1$; that is

$$W_1G_1 + W_2G_2 - H(H_{01}G_1 + H_{02}G_2) = 0. \tag{7.63}$$

Now let something change in the far-end room (speaker move or change) and the impulse responses there change to $G_1 + g_1$ and $G_2 + g_2$. Then the echo canceller will remain converged because

$$\begin{aligned} &W_1(G_1 + g_1) + W_2(G_2 + g_2) - H(H_{01}(G_1 + g_1) + H_{02}(G_2 + g_2)) \\ &\quad = W_1G_1 + W_2G_2 - H(H_{01}G_1 + H_{02}G_2) \\ &\quad + W_1g_1 + W_2g_2 - H(H_{01}g_1 + H_{02}g_2) \\ &\quad = 0. \end{aligned} \tag{7.64}$$

This filter structure is robust to changes in impulse responses in the far-end room. If the transfer functions in the near-end room change to $W_1 + w_1$ and $W_2 + w_2$, the adaptive filter will have to converge to $H + h$ to compensate for the changes. Is this compensation possible at all? We will have

$$\begin{aligned} &(W_1 + w_1)G_1 + (W_2 + w_2)G_2 - (H + h)(H_{01}G_1 + H_{02}G_2) \\ &\quad = [W_1G_1 + W_2G_2 - H(H_{01}G_1 + H_{02}G_2)] + [w_1G_1 + w_2G_2 - h(H_{01}G_1 + H_{02}G_2)] \\ &\quad = w_1G_1 + w_2G_2 - h(H_{01}G_1 + H_{02}G_2) \end{aligned} \tag{7.65}$$

as the first part of the second equation is equal to zero. Then the adaptive filter change is

$$h = (w_1G_1 + w_2G_2)(H_{01}G_1 + H_{02}G_2)^{-1}. \tag{7.66}$$

This solution exists because from Equation 7.63 we know that $H_{01}G_1 + H_{02}G_2$ is not zero. The adaptive filter can converge and compensate for the changes in the echo path. The two fixed filters thus reduce the degrees of freedom of the entire system; they act as constraints and hold the adaptive filter in a position where it can find the true solution. It is trivial to prove that, if there are changes in echo paths in both far- and near-end rooms, the adaptive filter can converge and the solution is Equation 7.66.

This structure will work well if we have good initial estimations of the echo paths in the near-end room. One easy way to do this is to play a short chirp signal from each loudspeaker consecutively at the beginning of each telecommunication session. Of course the chirp signals can be converted to something more melodic. It is important to have decorrelated wideband signals emitted from all loudspeakers for a short time. The entire initial estimation can take less than a second, including pauses before and after the calibration signal.

Another advantage of the proposed multichannel acoustic echo canceller is that the adaptive filter deals only with the small changes in the echo path. Most of the suppression comes from the fixed filters – the direct path and a substantial portion of the reverberation. People moving around in the near-end room cause relatively small increases in the residual, which the adaptive filter compensates for.

It is not a problem to extend this structure to surround-sound systems (five or seven loudspeaker channels) and for use with microphone arrays. Note that we have one adaptive filter per microphone channel, which means that this approach scales well for use with microphone arrays. With the traditional approach shown in Figure 7.5, a sound capture system with an eight-element microphone array and seven-channel surround-sound system will have to run 56 adaptive filters; while the approach in Figure 7.8 requires only eight. In addition, these 56 adaptive filters (if such a system can be made at all) should be computationally expensive RLS filters, while the eight can be regular NLMS filters with variable adaptation step.

7.8 Practical Aspects of the Acoustic Echo-reduction Systems

Building a robust acoustic echo-reduction system is a complex combination of research and engineering solutions. One of the most critical issues is guaranteeing that the adaptive filters will converge in all usage scenarios. Acoustic echo-reduction systems are no longer just for high-end professional equipment. Now personal computers are used for audio (and video) communication, with acoustic echo cancellation and suppression being part of the audio stack. People move their speakers during a session (which drastically changes the transfer function), they adjust the volume up and down using the knob on their loudspeakers or sound system (something AEC and AES are not aware of), and they place the microphones close to the loudspeakers, so the echo signal exceeds the local speaker voice by 20 dB or more. Laptop designers do the same (perhaps because both the microphone and the loudspeaker are part of the audio system and they should be together). All these factors require the addition of more processing blocks, many of which act as safety nets and engage only in critical situations – feedback, lost of convergence, and so on.

7.8.1 Shadow Filters

The basic idea is to have one fixed and one adaptive filter. The fixed filter is used to process the microphone signals. The adaptive filter works in parallel and adapts to changes in the transfer paths. When the adaptive filter starts to produce systematically better output its coefficients are copied to the fixed filter. The advantage here is that we can use more aggressive step sizes for faster convergence as we do not have to worry about the intermediate results during the convergence process. More details about shadow filtering can be found in [29].

7.8.2 Center Clipper

Humans can hear well even low-level signals with some organization. The echo residual is audible even when it is 20–30 dB below the level of the speech signal. For processing in the time domain the center clipper tracks the residual level and sets to zero all samples that are below this level:

$$\tilde{y}[k] = \begin{cases} y[k] & |y[k]| \geq \mathbb{R} \\ 0 & \text{otherwise.} \end{cases} \tag{7.67}$$

The clipping value $\mathbb{R}$ should be as small as possible and can be adaptive to track the residual.

The center-clipping equivalent in the frequency domain is the zeroing of all frequency bins with magnitude below the estimated residual level for that bin. In Chapter 4 it was explained that zeroing a frequency bin is never a good idea. This very old type of processing has been superseded by more sophisticated acoustic echo-suppression systems. Its advantage, however, is simplicity and ease of implementation, especially when the acoustic echo canceller runs in the time domain on low-power processors.

7.8.3 Feedback Prevention

Feedback is not only annoying, it prevents the adaptive filter readapting to the changed transfer functions. The problem occurs when for a certain frequency the phase shift is close to $(2n + 1)$ times 180° and the gain is larger than 1 for the entire loop: far end + near end + far end.

One of the simplest ways to prevent feedback is to apply a variable gain to the signal that goes to the local loudspeakers when feedback is detected. Some professional echo-cancellation devices use a set of adjustable notch filters. Once feedback is detected they engage and suppress the feedback frequency. The missing narrow frequency band cannot be detected by humans and this approach is applicable even for high-end systems. Another potential solution is to put in the input processing chain a constant tone suppressor like the one described in Chapter 4. It will detect and suppress the feedback signal, which will allow the adaptive filter to converge. Another frequently used approach is so-called "frequency shift." The general idea is to translate the spectrum of the input signal 5–10 Hz. The technique was designed in the mid 1960s for public address systems; for more details see [30].

7.8.4 Tracking the Clock Drifts

This problem is typical for personal computers. In devices such as speakerphones or mobile telephones, the sampling frequencies of the analog-to-digital and digital-to-analog converters are synchronized by using the same clock generator. In personal computers we can have a loudspeaker connected to the output of the sound card and external USB microphone (usually together with the web camera), or we can have USB speakers and USB microphone, and so on. In general, even when the sampling frequency is set to be the same for both devices (say 16 kHz), the sampling does not happen synchronously and the sampling rates are different (within certain limits) owing to different clock generators. The adaptive filter should constantly readapt to track the sampling rate drift, which reduces its suppression abilities. This is why acoustic echo-reduction systems for personal computers have an integrated block that estimates the delay between the loudspeaker and microphone signals and constantly adjusts the delay of the loudspeaker signal. Usually this happens by shifting the weighting window for frame extraction, before conversion in the frequency domain.

7.8.5 *Putting Them All Together*

In various chapters of this book we have discussed multiple audio processing techniques. Some of them are linear (AEC, beamforming), some of them are not (noise suppressor, AES). Building an end-to-end audio processing stack requires a proper sequencing of these processing blocks. The general rules are linear processing first, slower blocks first.

An example of a multichannel sound capture system for an advanced telecommunication system or for voice control of multimedia equipment is shown in Figure 7.9. The system has stereo or surround-sound playback and uses a microphone array for sound capture. The multichannel acoustic echo cancellation is first and works on each microphone channel. The microphone-array beamformer follows and combines the signals from all microphones into one, filtering the noise and reducing the reverberation from the local room. In addition it does some suppression of the echo residuals. The adaptive filters in the AEC converge independently, under slightly different reverberation and noise conditions (both near and far end). We can say that the echo residuals have low correlation in the short term. From this perspective, the echo residuals behave as uncorrelated noise for the microphone array. This type of noise is suppressed by the instrumental gain. The increased level of uncorrelated noise should be accounted for during the microphone array design if it uses a time-invariant beamformer. The AEC is placed before the beamformer because the microphone array acts as a highly directional microphone. Changing the beam direction causes rapid change of the transfer function between the loudspeakers and the beamformer output, which the AEC cannot follow if placed after the beamformer. It is possible to have the AEC after the beamformer only if the microphone array works with a fixed (non-steerable) beam.

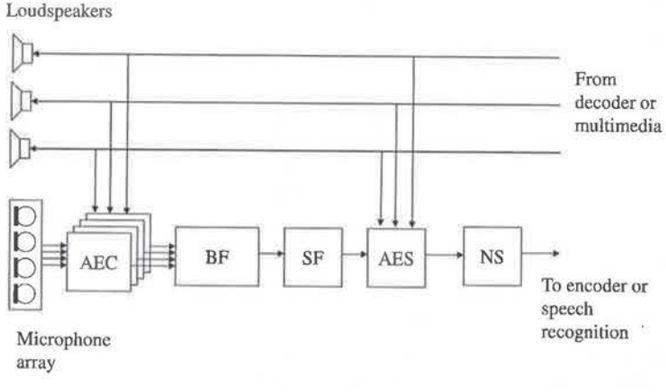


Figure 7.9 Block diagram of enhanced sound capture system with multichannel acoustic echo cancellation and microphone array

The beamformer output contains the enhanced local speech signal, decreased local noise levels, and still some echo residual. To increase the suppression from the microphone array, in Chapter 5 we proposed using a spatial filter, which in general is a sound source localizer per bin and suppressor for the bins where we have signals coming from an undesired direction. The echo residual should be processed by the acoustic echo suppressor discussed in this chapter. The stationary part of the local noise should be processed by one of the speech-enhancement algorithms discussed in Chapter 4. All of these algorithms compute and apply a time-varying real gain to the signal spectrum. They can be applied in the sequence above, or the gain computed jointly in a similar manner to Equation 7.52.

The output of this system goes either to the speech-recognition engine (if we want to do voice control of multimedia equipment) or to the encoder, which compresses the audio signal and sends it to the far-end room. The signal to the loudspeakers is either from the multimedia equipment (cable TV, DVD player, CD player, TV set, VCR, etc.) or from the decoder, converting the compressed audio from the far-end room to a waveform.

EXERCISE

Figure 7.8 is not complete. Identify the best places for the processing blocks discussed in this section. Where should the stationary tones compensation, or frequency shift, be placed? What about the sound source localizer from Chapter 6?

7.9 Summary

This chapter has discussed acoustic echo-reduction systems. They remove the sound from the loudspeakers which is captured by the microphone or microphones and is called echo. Such systems are part of all communication equipment and personal computers with speakerphone mode of operation. There are two major approaches for removing the echo: by cancellation and by suppression. Both are used in sequence to ensure the echo removal and maximal quality of the captured local speech signal.

Acoustic echo cancellers use adaptive filters to estimate the transfer path between the loudspeaker and the microphone. Then the signal sent to the loudspeakers is filtered and subtracted from the microphone signal. They should adapt to eventual changes in the transfer path. The adaptation process should happen when there is a loudspeaker signal (which can be detected by a voice activity detector) and there is no local speech signal, which is detected by a block called a double-talk detector. This block is an important part of each acoustic echo canceller.

Echo suppressors deal with the echo residual left after the acoustic echo canceller. They work in a similar to noise suppressors manner and use real-valued gain to reduce the residual echo. Estimating this residual is critical for acoustic echo suppressors and they use various adaptive algorithms to track and predict its power.

Multichannel echo reduction is part of high-end telecommunication systems and speech-controlled multimedia equipment. The main problem here is the non-uniqueness of the adaptive filter solution. This means that at any moment there are an infinite number of solutions, but they are different when some of the transfer paths change, except one – the true solution. This problem is resolved by faster adaptation, or advanced suppression algorithms, or by reducing the degrees of freedom of the adaptive system.

Building a robust system for echo reduction is a challenging research and engineering problem. Besides AEC and AES blocks, such a system may contain additional blocks such as algorithms for preventing feedback, and so on. An example of an end-to-end sound capture system has been described in this chapter.

Bibliography

[1] Sondhi, M. (1967) An adaptive echo canceller. *Bell System Technical Journal*, **46**, pp. 497–511.

[2] Vary, P. and Martin, R. (2006) *Digital Speech Transmission*, John Wiley & Sons, New York.

[3] Huang, Y. and Benesty, J. (eds) (2004) *Audio Signal Processing for Next-generation Multimedia Communication Systems*, Kluwer Academic, Norwell, MA.

[4] Haykin, S. (2002) *Adaptive Filter Theory*, 4th edn, Prentice-Hall, Upper Saddle River, NJ.

[5] Duttweiler, D. (2000) Proportionate normalized least-mean-squares adaptation in echo cancellers. *Transactions on Speech and Audio Processing*, **8**, 508–518.

[6] Benesty, J. and Gay, S. (2002) An improved PNLMS algorithm. Proceedings of International Conference on Acoustics, Speech, and Signal Processing, ICASSP '02. Orlando, FL.

[7] Ozeki, K. and Umeda, T. (1984) An adaptive filtering algorithm using an orthogonal projection to an affine subspace and its properties. *Electronic Communications of Japan*, **67-A**, 19–27.

[8] Gay, S. (2000) The fast affine projection algorithm, in *Acoustic Signal Processing for Telecommunication* (eds S. Gay and J. Benesty), Kluwer Academic, Boston, MA.

[9] Cho, J., Morgan, D. and Benesty, J. (1999) An objective technique for evaluating doubletalk detectors in acoustic echo cancellers. *IEEE Transactions on Speech and Audio Processing*, **7**, 6.

[10] Benesty, J., Morgan, D. and Cho, J. (2000) A new class of doubletalk detectors based on cross-correlation. *IEEE Transactions on Speech and Audio Processing*, **8**(2), pp. 168–172.

[11] Benesty, J. and Gänsler, T. (2006) The fast cross-correlation double-talk detector. *Signal Processing*, **86**, 1124–1139, Elsevier.

[12] Gänsler, T., Hansson, M., Invarsson, C.-J. and Salomonsson, G. (1996) A double-talk detector based on coherence. *IEEE Transactions on Communications*, **44**(11), 1241–1247.

[13] Birkett, A. and Goubran, R. (1995) Acoustic echo cancellation using NLMS-neural network structures. Proceedings of International Conference on Audio, Speech and Signal Processing ICASSP'95, Detroit, MI, pp. 2035–3038.

[14] Stenger, A. and Rabenstein, R. (1998) An acoustic echo canceller with compensation of nonlinearities. Proceedings of EUSIPCO 98, Isle of Rhodes, Greece.

[15] Stenger, A., Trautmann, L. and Rabenstein, R. (1999) Nonlinear acoustic cancellation with 2nd order adaptive Volterra filters. Proceedings of International Conference on Audio, Speech and Signal Processing ICASSP'99, Phoenix, AZ.

[16] Kuech, F., Mitnacht, A. and Kellermann, W. (2005) Nonlinear acoustic echo cancellation using adaptive orthogonalized power filters. Proceedings of International Conference on Audio, Speech and Signal Processing (ICASSP), Philadelphia, PA.

[17] Moon, T. and Stirling, W. (2000) *Mathematical Methods and Algorithms for Signal Processing*, Prentice-Hall, Englewood Cliffs, NJ.

[18] Bendersky, D., Stokes, J. and Malvar, H. (2008) Nonlinear residual acoustic echo suppression for high levels of harmonic distortion. Proceedings of International Conference on Audio, Speech and Signal Processing (ICASSP), Las Vegas, NV.

[19] Enzner, G., Martin, R. and Vary, P. (2002) Unbiased residual echo power estimation for hands-free telephony. Proceedings of International Conference on Acoustics, Speech, and Signal Processing (ICASSP), Orlando, FL.

[20] Chhetri, A., Surendran, A., Stokes, J. and Platt, J. (2005) Regression based acoustic echo suppression. Proceedings of International Workshop on Acoustic Echo and Noise Control (IWAENC), Eindhoven, The Netherlands.

[21] Madhu, N., Tashev, I. and Acero, A. (2008) An EM-based probabilistic approach for acoustic echo suppression. Proceedings of International Conference on Acoustics, Speech, and Signal Processing (ICASSP), Las Vegas, NV.

[22] Sondhi, M. and Morgan, D. (1991) Acoustic echo cancellation for stereophonic teleconferencing. Proceedings of Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA), Mohonk Mountain House, New Paltz, NY.

[23] Sondhi, M., Morgan, D. and Hall, J. (1995) Stereophonic acoustic echo cancellation: an overview of the fundamental problem. *IEEE Signal Processing Letters*, **2**(8), 148–151.

[24] Stokes, J. and Platt, J. (2006) Robust RLS with round robin regularization including application to stereo acoustic echo cancellation. Proceedings of International Conference on Multimedia and Expo (ICME), Toronto, Canada.

[25] Benesty, J., Morgan, D. and Sondhi, M. (1998) A better understanding and an improved solution to the specific problems of stereophonic acoustic echo cancellation. *IEEE Transactions on Speech and Signal Processing*, **6**(2), pp. 156–165.

[26] Herre, J., Buchner, H. and Kellermann, W. (2007) Acoustic echo cancellation for surround sound using perceptually motivated convergence enhancement. Proceedings of International Conference on Audio, Speech and Signal Processing (ICASSP), Honolulu, HI.

[27] Faller, C. and Tournery, C. (2006) Stereo acoustic echo control using a simplified echo path model. Proceedings of International Workshop on Acoustic Echo and Noise Control (IWAENC), Paris.

[28] Hirano, A. and Sugiyama, A. (1992) A compact multi-channel echo canceller with a single adaptive filter per channel. Proceedings of the IEEE International Symposium on Circuits and Systems, San Diego, CA, pp. 1922–1925.

[29] Hänsler, E. and Schmidt, G. (2004) *Acoustic Echo and Noise Control: A Practical Approach*, John Wiley & Sons, New York.

[30] Schroeder, M. (1964) Improvement of acoustic-feedback stability by frequency shifting. *Journal of Acoustic Society of America*, **36**, 1718–1724.

[31] Benesty, J., Amand, F., Gilloire, A. and Grenier, Y. (1995) Adaptive filter for stereophonic acoustic echo cancellation. Proceedings of International Conference on Audio, Speech and Signal Processing (ICASSP), Detroit, MI, pp. 3099–3102.

[32] Benesty J., Sondhi M. and Huang Y. (eds) (2008) *Speech Processing*, Springer-Verlag, Berlin, Germany.

[33] Buchner, H., Herbordt, W. and Kellermann, W. (2001) An efficient combination of multi-channel acoustic echo cancellation with a beamforming microphone array. Proceedings of International Workshop on Hands-free Speech Communication, Kyoto, Japan, pp. 55–58.

[34] Carini, A. (2001) The road of an acoustic echo controller for mobile telephony from product definition till production. Proceedings of International Workshop on Acoustic Echo and Noise Control (IWAENC), Darmstadt, Germany.

[35] Iqbal, M., Stokes, J., Platt, J. *et al.* (2006) Double-talk detection using real time recurrent learning. Proceedings of International Workshop on Acoustic Echo and Noise Control (IWAENC), Paris.

[36] Jiang, G. and Hsieh, S. (2004) Nonlinear acoustic echo cancellation using orthogonal polynomial. Proceedings of International Conference on Audio, Speech and Signal Processing (ICASSP), Montreal, Canada.

[37] Kim, S., Kim, J. and Yoo, C. (2003) Accuracy improved double-talk detector based on state transition diagram. Proceedings of Eurospeech 2003, Geneva, Switzerland, pp. 1421–1424.

[38] Kuech, F. and Kellermann, W. (2006) Ortogonalized power filters for nonlinear acoustic echo cancellation. *Signal Processing*, **86**, 1168–1181, Elsevier.

[39] McLachlan, G. and Krishnan, T. (1997) *The EM Algorithm and Extensions*, John Wiley & Sons, New York.

[40] Stenger, A. and Rabenstein, R. (1999) Adaptive Volterra filters for nonlinear acoustic echo cancellation. Proceedings of Nonlinear Signal and Image Processing (NSIP), Antalya, Turkey.

[41] Ye, H. and Wu, B. (1991) A new double-talk detection algorithm based on the orthogonality theorem. *IEEE Transactions on Communications*, **39**(11), 1542–1545.

EXERCISE

Compute and plot the Schroeder function for the impulse response, computed during the previous exercise. Find the T_{60} for this room.

8.1.5 Modeling

One of the most popular methods for synthesizing reverberated signals is the so-called "image method" [3]. The reflected sound from a wall can be modeled as an additional sound source, situated on the other side of the wall as an image of the original. Then given a room to model, the absorption coefficient of the walls, and the number of reflections, it is easy to represent the sound captured by the microphone in a given position as the sum of the sound sources (Figure 8.6). Each time a wall is crossed the sound magnitude is decreased by the absorption ratio and the phase inverted. (For non-rigid walls the reflected image will not be a point source.) The image method uses an angle-independent pressure wall reflection coefficient β, which leads to the energy absorption coefficient $\alpha = 1 - \beta^2$. Improvement of the image method can be found in [4]; the algorithm has been implemented in MATLAB and is available to use.

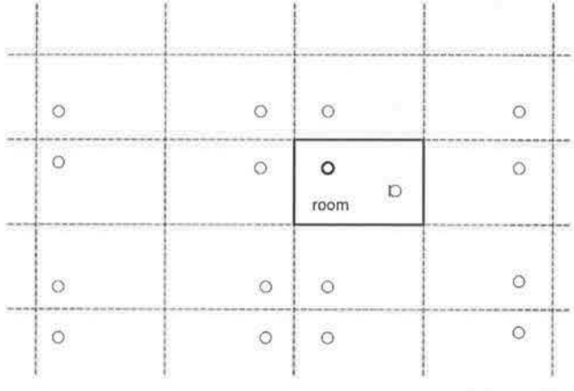


Figure 8.6 Image method for modeling of reverberation

The image method models the initial set of reflections well, but the reverberation tail is sparse and quite different from the actual measurements. This is due to a not very precise reflection model, which is frequency-independent and does not include energy diffusion, and to the fact that the room is modeled as empty with only eight reflective surfaces – the walls. Regardless of this, the image method is commonly used for generation of synthetic room impulse responses.

Using the idea of the image method we can do some interesting computations. The sound sources when a sound wave has already reached the microphone are inside a

sphere of radius $R = ct$, where c is the speed of sound. In this sphere we have

$$N = \frac{\frac{4}{3}\pi R^3}{V} = \frac{4\pi c^3 t^3}{3V} \tag{8.12}$$

sound sources where V is the volume of the room. The rate at which reflected waves arrive at the microphone is the first derivative:

$$\frac{dN}{dt} = \frac{4\pi c^3 t^2}{V} [\text{reflections/sec}]. \tag{8.13}$$

For a room of size 4 m × 3 m × 3 m ($V = 36\,\text{m}^3$), the moment when the reflections will arrive with a rate equal to the sampling rate of 16 kHz is 33.7 ms. This is roughly the moment where we can say that for that room the reverberation transforms from a discrete to a stochastic process.

8.2 De-reverberation via De-convolution

The general idea is to estimate an L-tap filter $\mathbf{G}_k$ applied to the captured signal to give us an estimation of the source:

$$\tilde{S}_k^{(n)} = \mathbf{G}_k^{\mathrm{T}} \mathbf{X}_k^{(n)} = \mathbf{G}_k^{\mathrm{T}} (\mathbf{H}_k^{\mathrm{T}} \mathbf{S}_k^{(n)}) + \mathbf{G}_k^{\mathrm{T}} \mathbf{N}_k^{(n)} \approx S_k^{(n)}. \tag{8.14}$$

In the ideal case $\mathbf{G}_k^{\mathrm{T}} \mathbf{H}_k^{\mathrm{T}} = 1$ and we achieve full restoration of the speech signal. There are two potential problems visible at first glance. The first is that the room impulse response $h(t)$ may not be invertible and the algorithm will have to find an approximate solution. A stable and causal system such as the transfer function $h(t)$ has a stable and causal inverse $g(t)$ only if it is a minimum-phase system. Unfortunately the room impulse response is almost never a minimum-phase system [12]. The second problem is that, even if it is invertible, we do not have access to the original speech, which was the case with acoustic echo-cancellation. Indirect criteria or properties of the speech signal have to be used as a criterion for updating the adaptive filter. This makes de-reverberation via direct estimation of the inverse filter a much more complex problem than acoustic echo cancellation.

EXERCISE

Use the .WAV file from the previous exercise. Even with complete knowledge of the sound source, can you find a good de-convolution filter? Evaluate the solution by filtering the microphone signal and comparing it to the speech signal.

8.2.1 De-reverberation Using Cepstrum

The main idea is to use the mean cepstrum as an estimate of the transfer function [5]. Once we have an estimation of H_{CC} from Equation 8.10, then it is converted back to the time domain, truncated to an appropriate length, and the de-convolution filter $g(t)$ is designed. Other algorithms try to use filtering in the cepstral domain based on empirical data. Most of the algorithms of this group are not seriously evaluated for improving speech-recognition results or human perception.

8.2.2 De-reverberation with LP Residual

Human speech production is modeled as an all-pole filter, while the reverberation adds only zeros. This means that the *linear prediction* (LP) coefficients will remain intact and only the LP residual will be affected. An entire group of algorithms use this fact. The general idea is to compute and subtract the LP and to work only with the residual. Once the reverberation effects are removed from the residual we can combine it back with the LP and resynthesize the speech signal. Such an algorithm is proposed by Yegnanarayana and Satyanarayana Murthy [6]. The LP residual is processed using the fact that the entropy is higher in the reverberant segments. Proper weighting is estimated and applied to regions with a high level of reverberation. The speech synthesis from the LP coefficients and processed residual inevitably introduces some distortions.

Similar LP properties are used by Gillespie *et al.* [7]. The authors use the fact that the reverberated LP residual has more of a Gaussian-shaped distribution, while the clean-speech LP residual is peakier; that is, it has higher kurtosis. To reduce the LP reconstruction artifacts, the authors propose a parallel structure (Figure 8.7). The adaptive filter works on the LP residual and maximizes the kurtosis. A copy of the filter works in parallel on the input signal. The algorithm works well in conditions of strong reverberation, where the difference is larger, but is weaker when the reverberation is

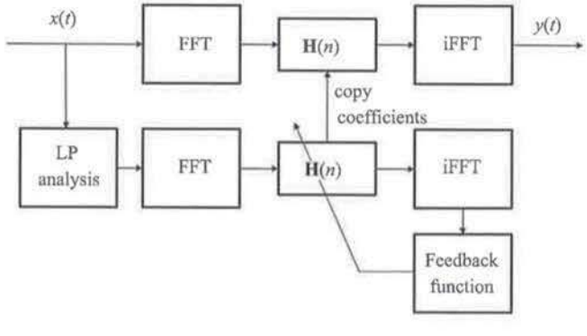


Figure 8.7 Block diagram of de-reverberator via maximizing kurtosis

low. The last requires longer adaption times, which harms the ability to track changes in the impulse response in the volatile tail.

8.2.3 De-reverberation Using Speech Signal Properties

Speech signals roughly consist of harmonic parts and noise-like segments. Nakatani *et al.* [8] claim that the second group is in general less affected by the reverberation, and a perceptual improvement can be obtained if the harmonic segments of the speech signal can be cleaned. The idea is to recognize these segments, to estimate the pitch, and to resynthesize the harmonic structure. Then the speech signal is assembled and evaluated. The algorithm is called HERB; it is iterative and computationally expensive which makes it more suitable for offline processing.

EXERCISE

Read one of the papers cited above and implement the algorithm. Record your own reverberant files and evaluate the algorithm. MATLAB provides good support for LP analysis. Process the .WAV file from the previous exercise and plot the Schroeder function. Compare it with the same function before the reverberation. Where are the strongest and weakest areas of the implemented algorithm?

8.3 De-reverberation via Suppression

The general idea is similar to the way noise suppressors and acoustic echo suppressors work. If we can estimate the reverberation energy in the signal we can use suppression techniques to estimate a real-valued gain for each frequency bin and reduce the bin magnitude proportionally to the portion of the reverberation energy. It is obvious that this approach will work better on the reverberation tail and will be weaker on the first reflections. On the other hand, most of the methods in the previous section can better estimate the first part of the correction filter, but usually fail to deal well with the volatile reverberation tail. From this perspective, the suppression group of methods provides good post-processors for the de-convolution methods. For speech recognition, reverberation suppressors can be the only de-reverberation algorithms used, as the cepstral mean normalization (CMN) in the front end deals well with the first reflections.

Essential for good reverberation suppression is a good estimate of the reverberation power. For the reverberation tail, a common model is the exponential energy decay both in the time domain and per frequency bin:

$$E_{\text{Rev}} = E_0 \exp\left(-\frac{t}{\tau}\right) \tag{8.15}$$

where E_0 is the initial signal power and τ is the time constant directly related to T_{60}. This simple model with only one parameter is very convenient to estimate, especially when the estimation is done per frequency bin and signals are quite noisy. Tashev and Allred [9] divide the frequency band into four or eight sub-bands and the decay time constant is estimated for each sub-band at the falling slope of the utterance, indicated by a voice activity detector. Via linear interpolation, this parameter is propagated for each frequency bin. It is trivial to convert the time constant to a single tap IIR (infinite impulse response) filter, which is used to process the input signal power and to receive the reverberation estimate. After this, an enhanced Wiener filter is used to suppress the reverberation. The algorithm's purpose is to improve the hands-free speech-recognition results, and it does not try to work with the first part of the impulse response.

More sophisticated methods for reverberation spectral variance estimation can be found in [10].

EXERCISE

Use the estimated room impulse response to compute the decaying time constant in several frequency bands. Model the reverberation spectral variance as a function of the input signal. Implement a simple Wiener suppression algorithm based on this estimation. Process the .WAV file from the previous exercise and plot the Schroeder function. Compare it with the same function before the reverberation. Where are the strongest and weakest areas of the implemented algorithm? Evaluate the results by listening to the output. Are there artifacts or musical noises?

8.4 De-reverberation with Multiple Microphones

8.4.1 Beamforming

In general, the more directional a microphone is the less reverberation it captures and the greater is the critical distance. From this perspective, microphone arrays that via beamforming can achieve a high directivity index are performing de-reverberation implicitly. Refer to Chapter 5 for the beamforming algorithms and microphone-array processing. This group of algorithms to date actually remains the most efficient, reliable, and commonly used method for de-reverberation.

8.4.2 MINT Algorithm

The authors of Chapter 12 in [11] propose this algorithm, based on the Bezout theorem. This says that, if the impulse responses from the sound source to each of the microphones have no common zeros, it is possible to perfectly compensate for the reverberation. This is a much lighter constraint than the one in single-channel de-convolution for minimum

phase. In essence, this approach combines the signals from all the microphone channels using the assumption that the information lost in one channel (i.e., zero for this frequency) is available in another channel (no common zeros requirement). The authors derive the estimation of the compensation filter. The approach should theoretically give perfect compensation of the reverberation, but it is sensitive to noise.

EXERCISE

Use some of the multichannel .WAV file from Chapter 5 or Chapter 6 to process with some of the beamforming algorithms from Chapter 5. Compute and compare the Schroeder function for one of the microphone channels and for the beamformer output. Where is the beamformer more efficient, and what is left from the reverberation?

8.5 Practical Recommendations

De-reverberation is a complex problem that is not yet completely solved. Before employing some of the algorithms from the literature, a good evaluation should be conducted on why de-reverberation is necessary. Typical applications are far-field sound capture for communications, and voice control of multimedia equipment.

In the first case the target is human ears. The evaluation criteria should be the perceptual sound quality achieved by the end-to-end system. The PESQ (perceptual evaluation of sound quality) algorithm or the criterion from [2] are good for evaluation, separately or in combination.

If the target is speech recognition, there is no better criterion than the recognition rate. Build a speech corpus, recorded in the target conditions, and tune the de-reverberation algorithm to minimize the word error rate.

One of the most robust and efficient solutions for decreasing reverberation remains the combination of a microphone array with a beamformer and reverberation suppressor after that. As was mentioned in the chapter on echo reduction (and in many other places in this book), chaining suppressors is not a good idea. The suppression blocks for noise, echo residual, and reverberation should be combined for joint suppression. Each type of suppression should have what was called in Chapter 4 "minimal gain." This is a number that can adjust the amount of suppression from each algorithm: a minimal gain of 0 means full suppression, while going up to 1 turns off this type of suppression. The system should be tuned end-to-end to achieve the best results according to the design goals.

EXERCISE

Where should the reverberation suppression be placed on Figure 7.8? Add a reverberation suppressor to the microphone array processor and compare the results with those from the previous exercise.