Ambient Noise Reduction

For Portable PCs and Tablets

White paper, June 2013

Itai Neoran, Director of R&D, Waves Audio

Aviv Shauby, Senior R&D Research Associate, Waves Audio

The recent adoption of portable computers and tablets as primary devices used for communication has brought new challenges in regards to the reduction of typical ambient sounds. At the same time, it presents new opportunities for the development of advanced algorithms in this field. This paper deals with some of the most common types of noises, and proposed solutions for suppressing them.

Introduction

Until recently, noise suppression for telecommunications has been associated, almost exclusively, with corded and mobile phones. These small hand-held devices assumed particular use-cases; where voice generally entered from a microphone near the mouth, while an output stream exiting from the other end of the device reached only one ear. Over the years, the need for ambient noise suppression for these devices has yielded an entire industry of technologies developing a wide range of solutions, mostly DSP processor-based.

However, recently many teleconferencing applications have begun shifting in favor of IP communications, running on CPU-based desktop computers, All-in one, notebooks, ultrabooks, convertibles and tablets. While desktop PCs are often used in moderate noise areas such as homes or quiet offices, all other devices are portable, thus exposed to countless types of ambience and new sounds. Unlike mobile phones, most people expect to use their portable PCs in a hands-free manner, while the user is positioned far away from the microphone(s) and loudspeaker(s), and with several people possibly participating on both ends of conference calls.

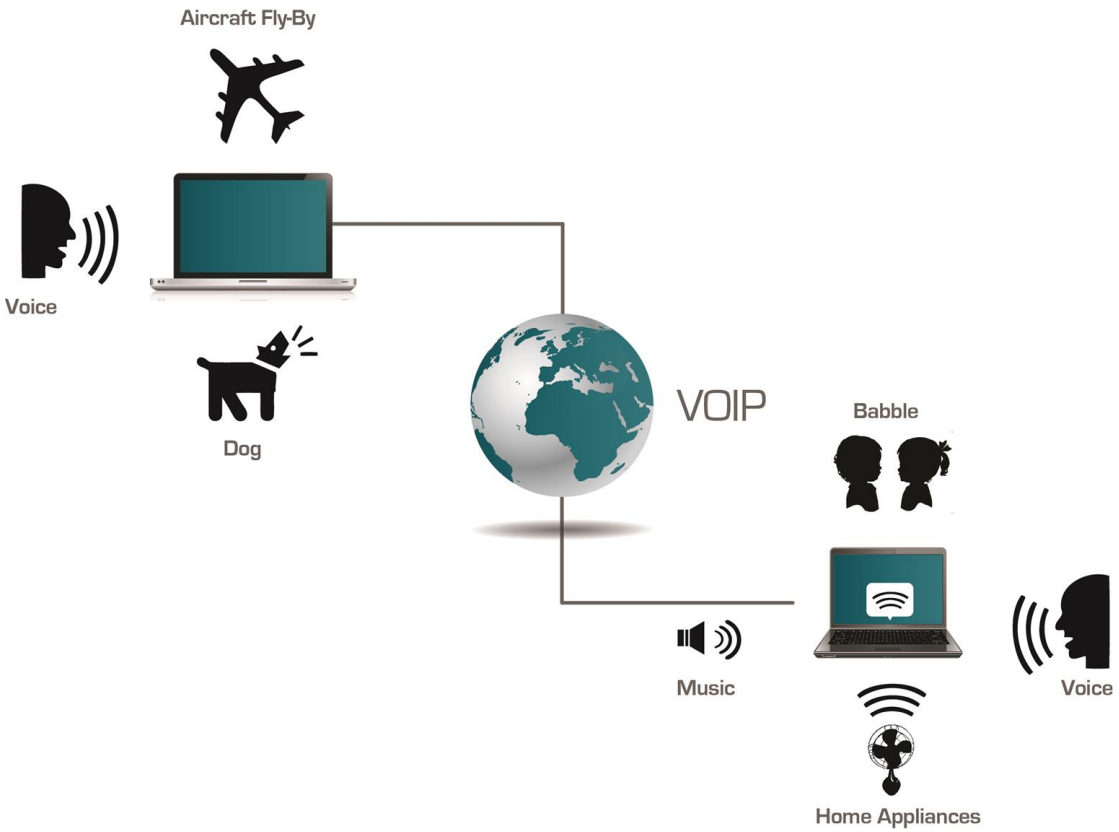
In addition, emerging automatic speech recognition (ASR) applications require solutions for noise-suppression, as pre-processing prior to the speech classification stage. ASR applications impose new

approaches in noise reduction, since classic noise-suppression strategies can sometimes degrade the performance of ASR classification, usually pre-trained on data with certain statistical assumptions.

As a result of the new use-cases, new types of noise entered the game. A good example is keyboard typing sounds and other noises conducted directly through a device casing. In addition to pure voice conversations, people expect to use VOIP applications in conjunction with gaming, music listening, and simultaneous commanding to an ASR engine. These multi-applied use cases result in complex situations since they combine many types of content as well many types of noise sources.

At the same time, portable devices are much larger than typical handheld corded-phones or mobile-phones, and their position toward the speaker/listener is relatively well defined, thanks to their wide screens. This has driven the development of new technologies, such as methods based on combining inputs from several microphones, e.g., ambience subtraction and beam-forming.

In teleconferencing, the voice transmission process always has two ends, even when several people take part in one conference call: the talking/transmitting end (denoted Tx) and the listening/receiving end (denoted Rx). Users expect noise to be suppressed at both ends, because the type of device and its audio quality used by the party at the other end is of no concern to them.



What is Noise?

When a single person stands in the middle of a noisy train station, with a mobile phone attached to one ear, the definition of noise is clear: for a near-end microphone, everything but the person's voice is considered noise.

Now, let's consider a more complex situation. Imagine three technicians in a noisy laboratory joining a hands-free conference call through a single ultrabook, one of them sitting and typing formulas through a chat window, while the other two are hovering behind him and making occasional comments. Away from the conference desk, other technicians in the same lab are busy discussing other projects between themselves. In this case, three of the voices are wanted sound sources, while other voices are considered noise. Certainly, laboratory machinery sounds are considered noise, as is the chat window typing noise. Can we expect an automatic noise reduction system to tell one voice from another?

In another recent use-case, a gamer is talking to his network opponents via a VOIP application running parallel to the game. Loud music and gunshots are heard from the game, while the gamer's dog is barking from his couch side, and passing cars on the street are heard through an open window across the room. In this scenario, can we even determine which sound source is wanted and which is

considered noise? Why would sound from cars on the street be considered noise, while game gunshots are wanted signals? What about the game's background music, not to mention the dog?

Clearly, in the above complex situations, the definitions of wanted signals and unwanted noise are dependent upon human understanding of the context which, as of this writing, computer software cannot analyze and understand. The famous idiom "one man's noise is another man's music" applies here, and suggests that we should give the end-user some real-time control, in order to help decide what to remove from the audio signal. This real-time control could include certain algorithmic parameters as well as the option to switch between noise-reduction algorithms.

We acknowledge that noise can be generated from various sources:

(1) External ambience, such as room tone (combination of domestic machinery), street noise (combination of motors), or babble noise (combination of speech like sources).

(2) From the PC itself, such as internal fan noise.

(3) From human interaction with the PC, such as keyboard typing or handling of the laptops's casing.

Noise may consist of random white hiss, machine-humming, clicks, music sounds, or any disturbance caused by human voices. Noise may be stationary, short, or transient.

An interesting question to ask is why, at all, users expect electronic conversation noise to be suppressed, while they do not expect the same when the conversation takes place face to face, in the real-world?

The answer, according to recent research, seems to be related to the way our brains process noise. The brain uses complex procedures to isolate and analyze surrounding noise sources, allowing humans to ignore noise sources once they can be predicted by the brain. In order for this to work well, the brain gathers three-dimensional sound information collected from two ears (binaural hearing), combined with head movements, followed by spatial separation of each of the wanted and unwanted sound sources.

The above three-dimensional information is missing from the transmitted audio signal of today's voice telecommunications. Since binaural hearing and head movement information is missing, and head movements take place on the listener’s end rather than the talker's end, this can lead to a wrong conclusion: that the noise arrives directly from the far-end loudspeakers. As a consequence, listeners feel much more uncomfortable with noise added to the voice they are listening to.

To allow the brain to do its job properly, we would need to record and transmit true binaural audio (e.g. artificial binaural head sensors), and ensure that the listeners head movements result in movements of the “artificial head” in the near-end listening space. While some academic work exists in this field, this kind of complete virtual-reality set-up is currently impractical for common VOIP use-case, making the straight-forward noise reduction methods attractive to most users.

For speech recognition applications (ASR), the definition of noise is simple: everything aside from the speech being detected, which interferes with the process, can be considered noise. While the definition of noise for ASR engines is simple, it is complicated to predict the effect of each such noise source on overall ASR performance, i.e. the rate of success in automatic interpretation of human commands, and the rate of word errors with commands and free dictations. ASR engines use combinations of many features extracted from the audio signal, where each feature is sensitive to a different type of noise. In general it may be said that ASR applications cannot function under low signal-to-noise ratios (when noise is closely intense or more intense than speech), and that the more the noise source sounds like speech (e.g. babble noise), the more sensitive the ASR performance is to it.

On the other hand, total absence of noise is not always beneficial with ASR performance, especially if the ASR algorithms are pre-trained on noisy sources. Moreover, certain ASR algorithms are insensitive to low level stationary random noise, but its removal may result in degradation of performance. We thus need a more careful determination of what exactly is the noise that should be reduced.

Characterizing Signals and Noises

When approaching the general problem of noise reduction, the first step is to form a mathematical model of the digital audio signal to be processed, describing all the wanted sources and noise sources involved, and how they are mixed, acoustically and electronically.

For example, when targeting an input signal, arriving from the device's microphone(s), we need to write a mathematical model which comprises:

1. The sound sources mixed acoustically in the air
2. Sound sources originating in the device casing
3. The microphone transfer function
4. Electronic and thermal noise in the microphone and its pre-amplifier
5. The analogue to digital signal conversion.

This model may also contain the estimated acoustic leak from the loudspeaker into the microphone, to be addressed by echo-cancellation algorithms.

Once this model is created, we analyze the statistical properties of each of the modeled sources, searching for properties that would allow us to discriminate between wanted and unwanted sources, by means of signal processing. There are three possible strategies:

1. Detect and separate the wanted sound sources, e.g. voice; everything else is noise.
2. Detect and separate all unwanted sound sources, e.g. machine noise and clicks; everything else is considered wanted signal.
3. Detect and separate both wanted and unwanted sound sources, and try to categorize them into wanted and unwanted through context-driven heuristics.

The respective algorithms differ for the three different strategies above, so if we build a system that switches from one approach to another, in real-time, user intervention is usually unavoidable, through an end-user control.

Approach (1) is very common in traditional digital telecommunications. Since the lines are designed to transmit voice only, LPC-like predictors and estimators are designed to model the human voice and encode it to a vocabulary of excitation signals, formant filters, and noise. Since random noise sources are unpredictable by these algorithms, the encoders transmit mainly the modeled voice, leaving behind a lot of the noise, un-transmitted to the far-end.

When the wanted signal is a single voice, close to the microphone, it is possible to model it correctly. However, when the person speaking is far from the microphone, room reflections are picked up by the microphone, and other voices may also join in. Some users may even play musical instruments or their MP3 players into the microphone. How can such a complex set of wanted signals be modeled by simple algorithmic predictors?

On the other hand, a simple situation of a microphone receiving clear voice, mixed with pre-amplifier stationary pink-noise, is easily dealt with by modeling the noise statistics, and suppressing it via 'Weiner' filter and spectral –subtraction techniques. The latter approaches measure only the statistics of the noise (2), and are thus common in restoration of old tape and vinyl recordings, where the wanted signal is complex and usually contains full range music, and where the noise statistics are rather simple and stationary.

Suppose such a noise-only approach (2) is used for noise reduction in our simple voice + pink noise example, and that at a certain moment, the user starts typing on the device's keyboard. Since the noise-based algorithm relates to signals beyond the stationary noise profile as 'wanted signal', it would let the keyboard clicks pass through. Indeed, in order to deal with the new type of noise, a new algorithm needs to join our processing chain.

Now, suppose that a dog starts barking in the background. We would need to add yet another algorithm for this extra type of noise. Very quickly we would find ourselves with dozens of algorithms running in a chain, each matching a particular type of noises, each appending its false-positive rate and artifacts to the audio, eventually yielding an unpleasant overall result.

Consequently, a third strategy (3) is preferred for modern teleconferencing in general and for VOIP communications and portable devices in particular. In this third approach we model the partials of the mixed signal via general statistical properties, and make decision about wanted vs. unwanted partials through context-driven heuristics.

For example in (3), we may propose a criterion that wanted signal is the loudest partial sound source that has voice-like statistics. Alternately, we may decide that the wanted sound source is always located directly in front of the device, while anything arriving from the sides is noise, and combine this with another assumption that impulsive noises (clicks) are always unwanted, no matter what they look like or where they originate.

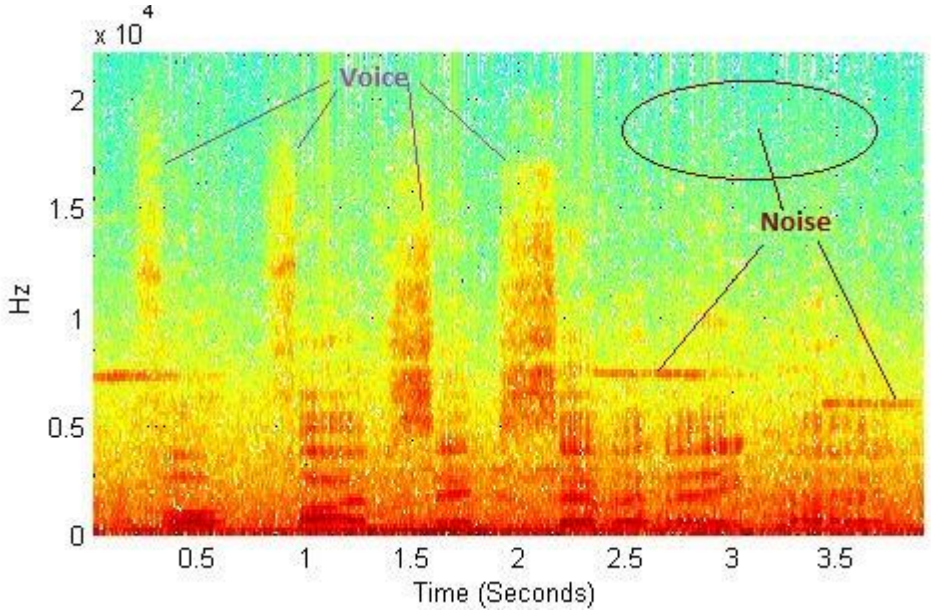
The better our heuristics, the less we need end-user intervention and real-time control.

Time Resolution vs. Frequency Resolution

If we were examining a pure 1kHz tone, and if we were told that it was composed as a mixture of signal and noise partials, then unless we had precise a-priori information about the composition, there would be no way to detect the noise or to remove it. Hence, to separate two unknown signals, they must not overlap in either time or frequency.

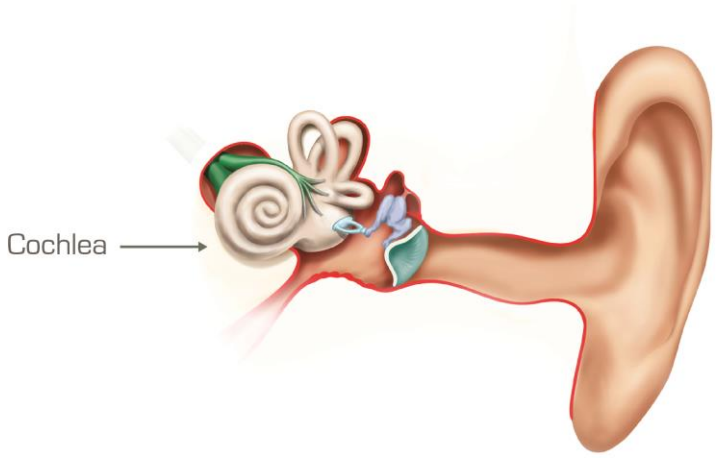
Since 'frequency' is defined as 'inverse time,' it may be demonstrated that the multiplication of the frequency-resolution by the time-resolution is a constant. This is called the ‘uncertainty principle’, which is a close cousin of Heisenberg’s principle, and at the very heart of quantum mechanics. We conclude that since a signal cannot be analyzed with infinite frequency resolution or with infinite time resolution, as our observation of frequency becomes more accurate, our analysis of time becomes less accurate.

The compromise between frequency-resolution and time-resolution is found within the basis of all noise reduction algorithms, and is a major factor that differentiates them. The sharper the frequency-resolution, the better separation we have between harmonics of the speech signal and the in-band noise. On the other hand, the better the temporal resolution, the less we affect consecutive speech phonemes when we remove the noise between them. It is unfortunate that these two resolutions are contradictory.



As can be seen in the spectrogram, plotting speech energy vs. time and frequency, it is possible to identify areas of speech and areas of noise, and possibly separate them with the right compromise of time and frequency resolutions.

Psycho-Acoustics investigates the way our ears and brain analyze temporal events and frequency elements. It is widely accepted that our inner-ear cochlea separates frequencies with a logarithmic resolution of about one third of an octave, ranging from fine frequency resolution and poor time resolution at low-frequencies, to coarse frequency resolution with excellent time resolution at high frequencies. Indeed, for wide-band signals and for wide-band noises, a good strategy is to build an algorithm with a similar logarithmic definition.



On the other hand, human hearing system is more complex than just logarithmic resolution. It features adaptive resolution to particular events in time, such as transients (sudden signal envelope or frequency changes), precise analysis of periodic signals (e.g. pitch in voice), and impressive averaging mechanisms over time and through head movements.

In many cases, the ability to suppress noise in particular frequency bands relies on its local energy being weaker than the energy of the signal at that band. In cases where high frequency harmonics of the signal are buried deep in high frequency noise, noise reduction is only possible if we construct very narrow frequency bands tightly around the narrow harmonic lines of the signal. In such cases, fine frequency resolution may be useful even in high frequencies, at the expense of some degradation in temporal definition.

The brain uses binaural information from both ears to estimate the direction of arrival of each sound partial, and a lot of the separation of signal from noise can happen this way. When two or more microphones with correct placement and spacing are used, the multi-channel captured signal may be used in a similar manner. Direction of sound partials may be analyzed and separated in two or even three dimensional spaces. Since sound waves travel in the air at an approximately constant speed, we may interchange time scale measurements of a signal with X-Y-Z coordinates in space. Three-dimensional spatial resolution can thus replace time resolution, competing against frequency-resolution through the same uncertainty principal: The finer the spatial resolution, the poorer the frequency-resolution, and vice versa.

A more useful way to relate to sound in space and time is to use polar coordinates. We can replace the X-Y-Z axis representation with the angular direction of each sound source (azimuth and elevation), and our complete measurement system then relates to direction, time, and frequency, which are all very useful in separating wanted signal partials from unwanted ones.

We conclude that noise reduction algorithms should relate to the type of signal (or noise) under inspection, and adapt the time resolution, directional resolution, and frequency resolution to each particular case. In some cases, smart heuristics may be used for the selection of the best axis, yielding optimal separation of a particular type of signal (or noise), while in other cases, this selection must even be left to a context-aware user decision.

Available Noise Reduction Tools

We examine here a variety of noise reduction algorithms, taken from a list of commercial products by Waves.

We define a 'beam' as a two-dimensional or three-dimensional angular range around a central axis, pointing outward from the center of all a device's microphones, to the expected (or preferred) position of the speaker's mouth. The direction of the beam's central axis may be steered in real-time, depending on use-case, type of device, orientation of the device (e.g. hand-held tablet), or even audio content. The beam width is defined by the width of its angular range. A wide beam indicates a wide angular range around the central axis. In many beam-forming algorithms the beam width relates to the direction where sound sources are attenuated by -3dB.

MaxxBeam™

MaxxBeam is an algorithm for directional noise reduction. It uses a beam-forming algorithm to distinguish noise from signal frequency partials based on both direction of each partial, the energy of each partial, and directness/diffuseness of each partial.

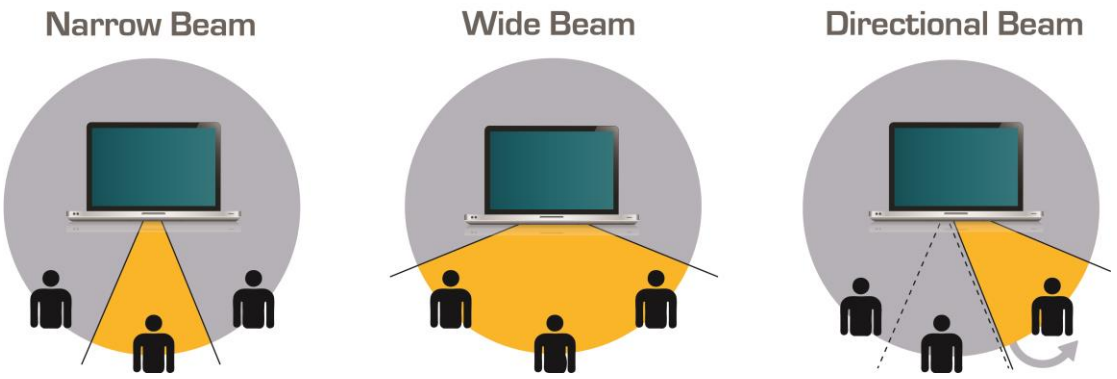
MaxxBeam applies three parallel directional sub-algorithms, where each of the three can be controlled independently:

1. A narrow beam, with a controlled width and central axis, selecting only one speech source or one direction from which speech should arrive. Inside the beam, speech is fully preserved. Outside the beam, everything is attenuated, where noise-like sources are attenuated even more.

2. A diffuseness detector and diffuse-sound reduction algorithm. This can operate well even with a very wide beam, where anything that is very directional is maintained, while diffuse sound with no discrete direction (mostly room reflections and noise) is attenuated.

3. Out-of-beam-silence algorithm, with a heuristic approach to handle speech starting and stopping, where whenever the main speaker is inactive, noises outside the beam are further attenuated until complete silence is achieved.

MAXXBEAM Microphone-Array Noise Suppression



MaxxBeam generates some latency between its input audio and its processed output audio, depending on the frequency resolution. This latency can be in parallel to other noise algorithms below so that total delay does not add up.

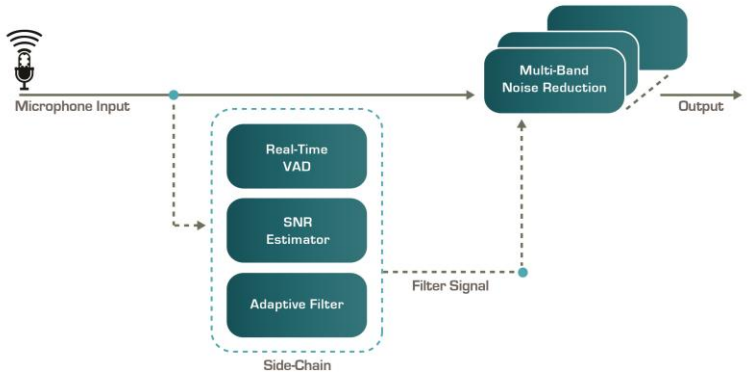
MaxxNR™

MaxxNR is an adaptive noise reduction algorithm. It is insensitive to direction, but highly sensitive to input content and performs simultaneous adaptation to speech and noise envelopes, as well as to noise statistics, all in logarithmic frequency resolution. The MaxxNR algorithm operates in multiple frequency bands, with good time resolution and very low phase shift. MaxxNR applies a voice activity detector (VAD) to classify signal as speech or noise for better adaptation, and automatically determines the amount of noise-reduction using a signal-to-noise (SNR) estimator.

Thanks to a unique filter-bank design, MaxxNR is capable of reducing noise in one frequency band while other frequency bands pass through cleanly.

MaxxNR algorithm features zero latency between its input and its processed output.

MAXXNR Noise Reduction

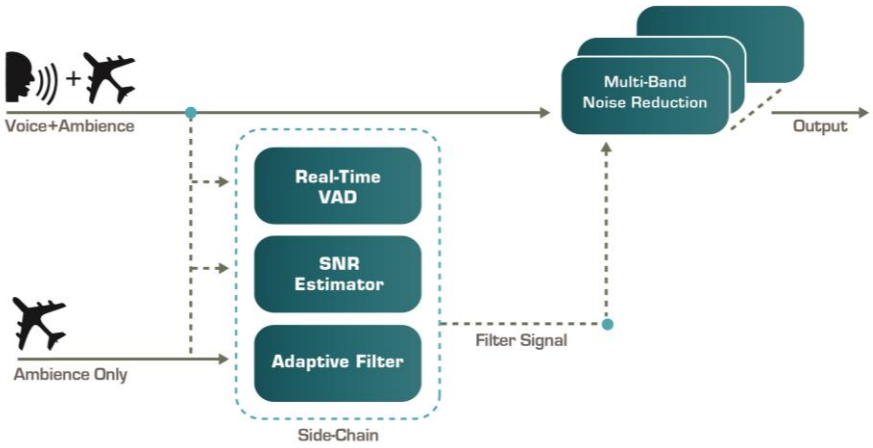


NS1™ and NS1 Pro™

The Noise Suppressor algorithm is similar to MaxxNR, with the addition of an end-user control (with a 0-100 scale) that determines the amount of noise reduction. This can be useful for providing some user control on the amount of noise-reduction (depth of effect).

NS1 Pro is also similar to MaxxNR, and adds a side-chain input, intended for devices having a dedicated ambience microphone (e.g. located at the back of the device). When this microphone signal is fed to NS1 Pro's side-chain input, more effective noise cancellation is achieved.

NS1 and NS1 Pro have zero latency between the input and the processed output.



WNS™

The Waves Noise Suppressor (WNS) algorithm is similar to MaxxNR, but provides more control over internal parameters while at the same time less tight adaptive behavior with respect to speech and noise. This implies that in situations where noise-statistics are well defined and do not change over time (such as a movie file with a particular noise), better noise suppression may be achieved. WNS has zero latency between its input and its processed output.

X-Noise™

When the signal is buried in strong noise (also known as 'negative signal-to-noise'), higher frequency resolution is required, at the expense of lower time resolution, resulting from the uncertainty principal. This can be attained by using high frequency-resolution FFTs. This is recommended for devices intended to be used in high-noise environments such as cars, buses, train stations, airports, and production lines.

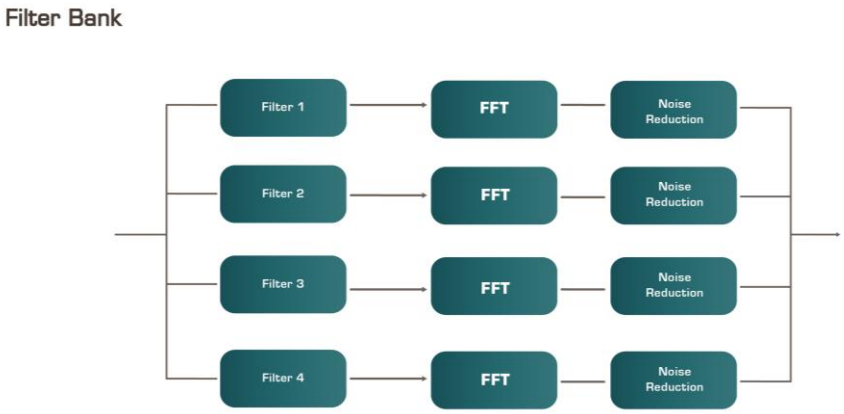
Due to the FFT operation, X-Noise generates some latency between its input and its processed output, depending on frequency resolution. This latency can be in parallel to other noise algorithms such as MaxxBeam.

Z-Noise™

To obtain even finer frequency resolution than X-Noise, while still maintaining transients and avoiding unnecessary phase smearing, Z-Noise uses a high frequency-resolution combination of filter-bank and FFTs, where the filter-bank is arranged in a logarithmic frequency resolution. This is recommended for situations of devices intended to be used in high-noise environments such as cars, buses, train stations, airports, and production lines.

Z-Noise is heavier in CPU consumption (MIPS) than X-Noise, but it can deal with more complex noise profiles and with more intense noise.

Due to the FFT operation, Z-Noise generates some latency between its input and its processed output, depending on filter bank architecture and frequency resolution. This latency can be in parallel to other noise algorithms such as MaxxBeam.



X-Click™ and X-Crackle™

X-Click and X-Crackle remove small clicks from an audio stream based on detection of the click, its removal, and interpolation of the missing audio.

While X-Click detects and removes clicks of typically 1-100 samples, (e.g., vinyl clicks or discontinuities in audio due to digital drops), X-Crackle detects and removes tiny high density clicks usually generated by electro-static noise (such as vinyl turntables).

MaxxVolume™ Gate

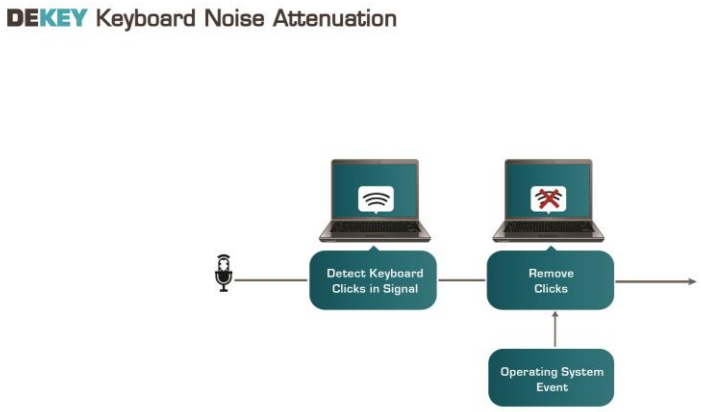
MaxxVolume™ is a complete solution for processing of dynamics, including peak limiting, dynamic range compression, loudness leveling (AGC), and noise-gating. The built-in noise-gate of MaxxVolume is capable of performing clean noise reduction in time-domain.

Since the MaxxVolume noise-gate operates in time domain, it is perfect for preserving sharp transients also on music and film contents, but it is limited to operate only in low-noise and good SNR situations. This is useful for reduction of noise originating from pre-amplifier hiss, CD player hum, or noisy analogue-to-digital converters.

DeKey™

The DeKey algorithm suppresses keyboard typing noise by detecting the particular type of clicks on the audio stream and by suppressing them in the audio output. To reduce 'false alarms,' detected clicks are correlated with operating-system keyboard events.

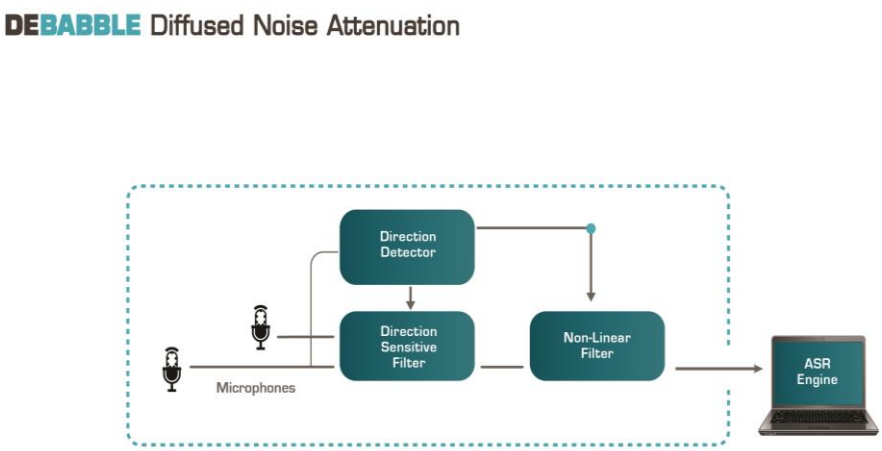
To allow removal of clicks right from their onsets, while avoiding confusion with speech onsets, DeKey generates some latency between its input and its processed output. This delay can be in parallel to the latency of other algorithms such as MaxxBeam or X-Noise.



DeBabble™

DeBabble is a direction-sensitive filter optimized to enhance automatic speech recognition (ASR) performance. Unlike VOIP conferencing, where several people may speak simultaneously, in ASR only one person may speak at a time. DeBabble detects and tracks the main speech source and then assumes all other speech-like sources are unwanted environmental babble.

DeBabble operates in real-time by applying a combination of linear direction filter and non-linear directional filter, both controlled by the detected direction of speech, suppressing some of the frequencies in the unwanted noise and speech sources, so as to minimize disturbance to ASR feature extraction. DeBabble's directional filter has been trained on ASR engines for highest CAR (Command Acceptance Rate) and lowest WER (Word Error Rate).



Use-Cases and Signal Models

We can now design a complete solution for portable PCs and tablets, based on well-defined use-cases and heuristic assumptions.

Four major use-cases are considered for the near-end microphone: single speaker, multiple speakers, music/gaming, and speech recognition. It is unrealistic to expect an algorithm to predict the desired use-case, so we intend for this 'mode' selection to be a user control.

In single-speaker mode, one person talks at the near-end and everything else is considered noise. The speaker usually is located directly in front of the device, 40-100 c”m away from the microphones. A small amount of room reflections may be received by the microphones along with the dry speech. Since potential broad-band noise sources do overlap with speech in frequency and in time, we also need to assume that:

- Directly in front of the device, within a pre-defined beam, the speaker’s voice is always stronger than noise arriving from within the beam.
- Outside the beam, everything is noise.
- Noise statistics may vary with time.
- All clicks or other impulsive events are noise.

The single speaker mode can yet be split into two sub-modes: normal noise vs. very high noise situations (dealt with in the next chapter).

In multiple-speaker mode, several speakers may be talking from the near-end point, consecutively or simultaneously. All are located within a half sphere in front of the device, 40-100 cm away from the microphone(s). For some of the farthest speakers, a fair amount of room reflections may be received by the microphones along with their respective dry voices, though the reflection energy is assumed not to exceed the energy of the dry voices. Since potential broad-band noise sources overlap with speech in frequency, in time, and partially in direction, we also need to assume that:

- The wanted signal consists of sound sources that are all 'speech'.
- Sound sources consistent in their spatial direction, and arriving from a discrete well-defined direction, are 'wanted' signal partials.
- Sound sources inconsistent in their spatial direction, or arriving from a well defused direction, are noise.
- All clicks or other impulsive events are unwanted noise sources.

In music/gaming mode, a single speaker talks at the near-end, while music plays in the background (e.g. live concert or game), as well as possible special effects (e.g. gunshots, explosions, bumping noises). We also maintain that the speaker is not necessarily positioned in front of the device with -100 c”m distance from the microphones. A small amount of room reflections may be received by the microphones along with the dry speech. All noise sources overlap with the mixture of speech and music in frequency, time, and direction. We thus assume that:

- The user expects almost all types of content to be transmitted to the far-end.
- The user does not mind if some low-level room reflections (reverberation) is not transmitted to the far-end.

- Ambience noise envelope and frequency content change only very slowly over time (forced assumption to distinguish it from all other content)
- Noise is considerably weaker than the signal. Stationary low-level wide-band signal is considered to be noise.
- Only clicks coinciding with operating-system keyboard events are considered noise.

In speech recognition (ASR) mode, as in single-speaker mode, one person talks at the near-end and everything else is considered noise. The speaker is located near the device, 40-100 c”m away from the microphone(s), but not necessarily directly in front of the device. A small amount of room reflections may be received by the microphones along with the dry speech. Since potential broad-band noise sources do overlap with speech in frequency and in time, we also need to assume that:

- The speaker’s voice is always stronger than the noise.
- The noise is random and diffused (e.g. room tone, wind, babble), meaning that no other discrete sound sources interfere, except for leak from the loudspeakers into the microphone, which is dealt-with separately via echo-cancellation.
- Noise statistics may vary with time.
- Any clicks or other impulsive events are noise, but should not be removed, as their removal would degrade ASR algorithmic performance

In all use-cases above, we also assume that ambient noise does not vary quickly over time. This assumption is more acceptable for notebooks and tablets than it is for mobile phones.

For the discussion in this paper, we assume at least two microphones in all devices. In cases where only one microphone is available, all discussions on directional data and beam become irrelevant.

Noise Reduction Solutions

We now present possible solutions for each of the defined use-cases.

1. Single-speaker mode

In this mode, a directional noise reduction algorithm is required, combined with an adaptive noise reduction algorithm. The adaptive noise reduction algorithm needs to detect speech, to follow both speech envelope and noise envelope, and good time resolution to avoid artifacts on the voice. In addition, clicks and other transients need to be identified and removed.

We propose the MaxxBeam algorithm for directional noise reduction. MaxxBeam uses a beam-forming algorithm to distinguish noise from signal frequency partials based on the direction of each partial, the

energy of each partial, and the directness/diffuseness of each partial. We use rather narrow beam settings in MaxxBeam to obtain better separation of the single speaker.

MaxxBeam is then followed by MaxxNR , an adaptive noise reduction algorithm. The latter features close adaptation to the speech-envelope and noise-envelope at logarithmic frequency resolution, with rather good time resolution and very low phase shift. MaxxNR applies a voice activity detector (VAD) to classify the signal as speech or noise for better adaptation.

MaxxNR may be interchanged with other noise reduction algorithms such as WNS, NS1, and NS1 Pro. These four algorithms are similar in their time and frequency resolutions, but there are some relevant differences as explained in detail in the previous chapter. In this manner, WNS would obtain better noise separation in situations where noise-statistics are well defined, NS1 would allow a most effective end-user control, and NS1 Pro would add functionality of a side-chain input, intended for devices having a dedicated ambience microphone, for an effective noise cancellation.

Yet another alternative to MaxxNR may be proposed in particular use-cases of very intense noise. As mentioned in previous chapters, when the signal is buried in strong noise (a.k.a negative signal-to-noise), higher frequency resolution is required, at the expense of somewhat more “musical noise” artifacts. This can be attained by using an alternative noise-reduction algorithm with Z-Noise algorithm. This is recommended for situations of devices intended to be used in high-noise environments such as cars, busses, train-stations, airports, and production lines.

MaxxNR (or its alternative) is then followed by DeKey, a click detection and removal algorithm focused on keyboard typing noise.

2. Multiple-speaker mode

In this mode, a multi-directional noise reduction algorithm is required, combined with an adaptive noise reduction algorithm. The adaptive noise reduction algorithm needs to detect speech, follow both speech envelope and noise envelope, and needs a good time resolution to avoid any artifacts on the voice. In addition, clicks and other transients need to be identified and removed.

We propose MaxxBeam algorithm for separating discrete voices from diffuse noise and diffuse room reflections. We set MaxxBeam to very wide beam settings (half sphere) where MaxxBeam uses a frequency correlation algorithm to distinguish noise from signal frequency partials based on the directness/diffuseness of each partial.

MaxxBeam is then followed by MaxxNR, an adaptive noise reduction. The latter features close adaptation to the speech-envelope and noise-envelope at logarithmic frequency resolution, with rather good time resolution and very low phase shift. MaxxNR applies a voice activity detector (VAD) to classify the signal as speech or noise for better adaptation.

In this mode, as in the previous mode, MaxxNR may be interchanged with other noise reduction algorithms such as WNS, NS1, NS1 Pro, and Z-Noise. Nevertheless the side-chain input feature of NS1-pro™ would be less effective in multiple-speaker mode as some of the speakers may be received by the dedicated ambience microphone.

MaxxNR (or its alternative) is then followed by DeKey, a click detection and removal algorithm focused on keyboard typing noise.

3. Music/gaming mode

In this mode, noise-reduction must not rely on directivity or on diffuseness. Instead, an algorithm with high-frequency resolution is required, where the threshold must be set to a low level, in order to deal only with low-level stationary noise.

Such algorithms are found in X-Noise and Z-Noise, where the latter has even higher frequency resolution via a combination of an FFT with logarithmic filter-bank, with extra real-time complexity. Definition of the exact frequency resolution needed for this use-case is per device and would match the quality of the microphones used in it.

MaxxNR (or its alternative) may then followed by DeKey, a click detection and removal algorithm focused on keyboard typing noise, where the algorithm is set to remove only clicks attached to operating system typing events.

4. Speech Recognition (ASR) mode

In this mode noise reduction relies mainly on three steps:

- Echo-cancellation removes all leaks from loudspeakers into the microphones. This requires a stereo, full-duplex algorithm, also available commercially by Waves but is outside the scientific scope of this paper.
- MaxxBeam set to wide-beam, with diffused sound removal on and out-of-beam-silence off.
- DeBabble is used for removing babble noise and other diffuse noise in a way preserving the quality of ASR feature extraction.
- MaxxNR or X-Noise set to subtle noise suppression of broad-band noise, intended for whitening of the residual noise rather than completely removing it.

Conclusions

We presented four use-cases for voice telecommunications using notebook, ultra-books, and tablets, where some of the cases also to desktop PCs. For each or the use-cases, we analyzed the strategies for low, mild, and intense noise situations.

We presented ten different noise reduction algorithms, and how they are combined in the three use-cases above to handle complex wanted sources vs. unwanted noise sources.

Other use-cases and modes of operation may be defined by particular device manufacturers and particular VOIP applications and ASR applications, and appropriate algorithm alternative combinations may be defined.

Future systems can also be tailored to operate in a more context-aware heuristic manner, and switch automatically between particular algorithms.

In the future, once stereo is supported in VOIP applications, more advanced algorithms can be applied by using binaural information.

WAVES

CONTACT INFORMATION

Waves North America Office
Waves Inc.
2800 Merchants Drive
Knoxville, TN 37912
Phone: 1-865-909-9200
Fax: 1-865-909-9245

Waves Corporate Headquarters Israel
Waves Audio Ltd.
Azrieli Center 3
The Triangle Tower, 32nd Floor
Tel-Aviv 67023, Israel
Phone: +972-3-608-4000
Fax: +972-3-608-4056