Interspeech 2018
2-6 September 2018, Hyderabad

Implementation of Digital Hearing Aid as a Smartphone Application

Saketh Sharma[1], *Nitya Tiwari*[1], *Prem C. Pandey*[1]

[1]Department of Electrical Engineering, Indian Institute of Technology Bombay, Mumbai, India
sakethgsharma@gmail.com, nitya@ee.iitb.ac.in, pcpandey@ee.iitb.ac.in

Abstract

Hearing aids for persons with sensorineural hearing loss aim to compensate for degraded speech perception caused by frequency-dependent elevation of hearing thresholds, reduced dynamic range, abnormal loudness growth, and increased temporal and spectral masking. A digital hearing aid is implemented as a smartphone application as an alternative to ASIC-based hearing aids. The implementation provides user-configurable processing for background noise suppression and dynamic range compression. Speech enhancement technique using spectral subtraction based on geometric approach and noise spectrum estimation based on dynamic quantile tracking is implemented to improve speech perception. To compensate for reduced dynamic range and frequency-dependent elevation of hearing thresholds, a sliding-band dynamic range compression technique is used. Both processing blocks are implemented for real-time processing using single FFT-based analysis-synthesis. Implementation as a smartphone application has been carried out using Nexus 5X with Android 7.1 Nougat OS. A touch-controlled graphical user interface enables the user to fine tune the processing parameters in an interactive and real-time mode. The audio latency is 45 ms, making it suitable for face-to-face communication.

Index Terms: dynamic range compression, hearing aid, noise suppression, smartphone application

1. Introduction

Sensorineural hearing loss is associated with loss of sensory hair cells in the cochlea or degeneration of the auditory nerve. It may be inherited genetically or may be caused by excessive noise exposure, aging, infection, or ototoxic drugs. It is characterized by frequency-dependent elevation of hearing thresholds, abnormal growth of loudness known as loudness recruitment, increased temporal and spectral masking, and widening of auditory filters leading to degraded speech perception [1]-[6]. Several signal processing techniques have been reported for improving the speech perception by patients suffering from sensorineural hearing loss.

Frequency selective amplification and dynamic range compression are the primary processing techniques used in hearing aids [6]-[8]. Single band dynamic range compression leads to reduced high-frequency audibility and multiband dynamic range compression may lead to perceptible distortion due to a transition of speech formants across the band boundaries. These problems can be addressed by using sliding-band dynamic range compression [9]-[10]. The compression parameters can be tuned to fit the frequency dependent thresholds and loudness recruitment of the patient.

Persons with sensorineural loss experience difficulty in understanding speech in a noisy environment. Processing for noise suppression in a hearing aid can improve speech audibility and quality. Spectral subtraction [11], a single-channel speech enhancement technique using an estimate of the noise spectrum, is suitable for such applications as it has low algorithmic delay and computational complexity. Dynamic quantile tracking based noise estimation [12]-[13] is reported to track stationary and non-stationary noise efficiently and can be used for real-time noise suppression.

Hearing aids are generally designed using ASICs due to power and size constraints, leading to prohibitive costs in development and testing of new processing techniques. Use of smartphone-based application (app) as a hearing aid can provide user-configurable settings and a greater flexibility to hearing aid users and developers. A smartphone app with sliding-band dynamic range compression has been developed earlier [14]. Incorporation of processing for suppression of background noise is needed to extend the usefulness of the app under different listening conditions. This paper presents an implementation of signal processing for noise suppression based on computationally efficient dynamic quantile tracking along with the earlier reported sliding-band dynamic range compression as a smartphone app. The signal processing techniques and implementation details for dynamic range compression and noise suppression are presented in the second section. Implementation of the hearing aid app is presented in the third section, followed by test results in the fourth section and conclusions in the last section.

2. Signal processing

Signal processing for dynamic quantile tracking based noise suppression [12] and sliding-band dynamic range compression [9] use a DFT-based analysis-synthesis. The two processing techniques are described briefly in the following subsections.

2.1. Dynamic quantile tracking based noise suppression

Dynamic quantile tracking based noise estimation [12] is used along with spectral subtraction for noise suppression. For each frequency bin of the spectrum, the most frequently occurring value, obtained as the peak of the histogram, is reported to be representative of the noise value [15]. The noise estimation technique dynamically estimates histogram using the dynamic quantile tracking with low memory and computation requirements [16]. The peak of the histogram is used as the adaptive quantile for estimating the noise at each spectral sample. The histogram is estimated by dynamically tracking multiple quantile values for a set of probabilities chosen to be evenly spaced. The desired quantile corresponding to the peak of the histogram is obtained by finding quantile for which the difference between neighboring quantile values is minimum. The estimate of the noise spectrum at the *n*th frame and *k*th spectral sample is obtained as

$$\hat{D}(n,k) = \arg\min_{\hat{q}_i(n,k)}[\hat{q}_i(n,k) - \hat{q}_{i-1}(n,k)];\ i = 2,3,...,J \quad (1)$$

where J is the number of quantiles tracked. The estimate of quantile, $\hat{q}_i(n,k)$, is obtained by applying an increment or a decrement on its previous estimate as

$$\hat{q}_i(n,k) = \hat{q}_i(n-1,k) + d_i(n,k) \tag{2}$$

The change $d_i(n, k)$ is given as

$$d_i(n,k) = \begin{cases} \Delta_i^+(k), & |X(n,k)| \geq \hat{q}_i(n-1,k) \\ -\Delta_i^-(k), & \text{otherwise} \end{cases} \tag{3}$$

where $\Delta_i^+(k)$ and $\Delta_i^-(k)$ are selected to be appropriate fractions of the range $R(n, k)$ as

$$\Delta_i^+(k) = \lambda R(n,k) p_i(k) \tag{4}$$

$$\Delta_i^-(k) = \lambda R(n,k)(1 - p_i(k)) \tag{5}$$

The range is estimated using dynamic peak and valley detectors for updating the peak $P(n, k)$ and the valley $V(n, k)$ using the following equations:

$$P(n,k) = \begin{cases} \tau_p P(n-1,k) + (1-\tau_p)|X(n,k)|, & |X(n,k)| \geq P(n-1,k) \\ \sigma_p P(n-1,k) + (1-\sigma_p) V(n-1,k), & \text{otherwise} \end{cases} \tag{6}$$

$$V(n,k) = \begin{cases} \tau_v V(n-1,k) + (1-\tau_v)|X(n,k)|, & |X(n,k)| \leq V(n-1,k) \\ \sigma_v V(n-1,k) + (1-\sigma_v) P(n-1,k), & \text{otherwise} \end{cases} \tag{7}$$

$$R(n, k) = P(n, k) - V(n, k) \tag{8}$$

The dynamic quantile tracking to estimate quantile $\hat{q}_i(n,k)$ as given by (2), (3), and (8) can be written as the following:

$$\hat{q}_i(n,k) = \begin{cases} \hat{q}_i(n-1,k) + \lambda p_i(k) R(n,k), & |X(n,k)| \geq \hat{q}_i(n-1,k) \\ \hat{q}_i(n-1,k) - \lambda(1-p_i(k)) R(n,k), & \text{otherwise} \end{cases} \tag{9}$$

Spectral subtraction based on geometric approach [17]-[18] is used for suppression of background noise, as it results in smaller residual noise.

2.2. Sliding-band dynamic range compression

The processing for sliding-band compression [9]-[10] comprises the steps of short-time spectral analysis, spectral modification, and signal re-synthesis. To compensate for increased hearing thresholds and reduced dynamic range, a frequency-dependent gain function is calculated in accordance with the desired levels for 'soft', 'comfortable', and 'loud' sounds (referred to as SL, CL, LL, respectively). For each spectral sample k, the spectral modification is carried out using a piecewise linear relationship between the input power and the output power on dB scale. The relationship is specified by the values of $P_{OdBSL}(k)$, $P_{OdBCL}(k)$, and $P_{OdBLL}(k)$ which are the output signal levels corresponding to soft, comfortable, and loud sounds, respectively, for the hearing aid user and by the values of $P_{IdBSL}(k)$ and $P_{IdBLL}(k)$ which are the input signal levels corresponding to soft and loud sounds, respectively, for a normal-hearing listener. The relationship is defined in three regions with the compression ratio as 'CR = 1', 'CR > 1', and 'CR = ∞' in the first, second, and third region respectively. With $G_{LdB}(k) = P_{OdBSL}(k) - P_{IdBSL}(k)$, the target gain for the spectral sample k in the i th frame in the three regions is given as

$$G_{TdB}(i,k) = \begin{cases} G_{LdB}(k), & P_{IdB}(i,k) < P_{OdBCL}(k) - G_{LdB}(k) \\ \dfrac{G_{LdB}(k) - \{P_{IdB}(i,k) - P_{OdBCL}(k)\}\{\mathrm{CR}(k) - 1\}}{\mathrm{CR}(k)}, & P_{OdBCL}(k) - G_{LdB}(k) \leq P_{IdB}(i,k) \leq P_{IdBLL}(k) \\ P_{OdBLL}(k) - P_{IdB}(i,k), & P_{IdB}(i,k) > P_{IdBLL}(k) \end{cases} \tag{10}$$

For the spectral sample k in the ith frame, the input level $P_{IdB}(i,k)$ is calculated as the sum of squared magnitudes of the spectral samples in the band centered at k and with the bandwidth corresponding to the auditory critical bandwidth

$$\mathrm{BW}(k) = 25 + 75\left(1 + 1.4\left(f(k)\right)^2\right)^{0.69} \tag{11}$$

where $f(k)$ is the frequency, in kHz, corresponding to the kth spectral sample. For spectral modification, the target gain is converted to a linear scale. The gain applied to the kth spectral sample in the ith frame is obtained using the desired attack and release rates by updating the gain from the previous value towards the target value, as given in (10). It is given as

$$G(i,k) = \begin{cases} \max\left(G(i-1,k)/\gamma_a, G_T(i,k)\right), & G_T(i,k) < G(i-1,k) \\ \min\left(G(i-1,k)\cdot\gamma_r, G_T(i,k)\right), & G_T(i,k) \geq G(i-1,k) \end{cases} \tag{12}$$

The number of steps during the attack and release phases are controlled using gain ratios $\gamma_a = (G_{max}/G_{min})^{1/s_a}$ and $\gamma_r = (G_{max}/G_{min})^{1/s_r}$, respectively. Here G_{max} and G_{min} are the maximum and minimum possible values of the target gain. The number of steps s_a during attack and the number of steps s_r during release are selected to set the attack time as $T_a = S_a \cdot S/f_s$ and the release time as $T_r = S_r \cdot S/f_s$, where f_s = sampling frequency and S = number of samples for frameshift. A fast attack avoids the output level from exceeding the uncomfortable level during transients, and a slow release avoids amplification of breathing.

3. App implementation

The smartphone app for real-time processing has been developed and tested using 'Nexus 5X' with Android 7.1 Nougat OS due to its relatively small audio I/O delay and high processing capability. Like the earlier app for dynamic range compression [14], it has a touch-controlled graphical interface, enabling the user to adjust the processing parameters in an interactive and real-time mode. In addition to the facility for setting the processing parameters for noise suppression and dynamic range compression, there is a provision for the parameters for additional processing blocks of the future versions.

Figure 1 shows a block diagram of the implementation of the hearing aid app. The setup comprises a handset with its headset. The headset consists of a microphone and a pair of earphones with associated wires and switching. The handset consists of the codec, the processor, and the touch screen for user interface. The input signal acquired from the microphone of the headset is amplified and is converted to digital samples by ADC of the codec. These samples are buffered, processed, and buffered by the processor. The resulting samples are output through DAC of the codec and amplified. The resulting signal is output through the earphones of the headset. The input samples acquired in an S-word buffer and the previous samples stored in a $3S$-word buffer form the L-sample input window for FFT-based analysis-synthesis. The processing for noise suppression and dynamic range compression is carried out using N-point FFT of the input window. The N-point IFFT of the modified complex spectrum is calculated and the output signal is re-synthesized using overlap-add. The analysis-synthesis uses 20-ms frames with 75% frame overlap and 1024-point FFT. The processing is carried out using sampling frequency = 24 kHz, L = 480, S = 120, and N = 1024. The program was written using a combination of C++ and Java, with Android Studio 2.3.0 as the development environment.

The screenshot of the home screen of the app is shown in Figure 2. The play/stop button is for control of the output. All processing modules have individual on/off and 'settings' buttons. The on/off button can be used for toggling the processing and the settings button can be used for setting the processing parameters graphically. Figure 3a shows screenshot of the 'settings' screen for noise suppression and dynamic range compression modules. Settings for noise suppression module provides user interface (UI) with touch control of points, called 'thumbs', for selecting the values of over-subtraction factor α as function of frequency. The values of α can be set as 1−5 for up to 10 frequencies and the values for all the intermediate frequencies are obtained by smooth curve fitting. The screenshot of the 'settings' screen for dynamic range compression showing graphical controls for the SL, CL, and LL values, is shown in Figure 3b. The UI consists of three touch-controlled curves to set the values of SL, CL, and LL across frequencies. Control points called as thumbs are provided to adjust the curves. Each curve consists of 10 thumbs. Provision is provided to store and retrieve up to 4 parameter settings. The UI also consists of undo and redo button to access recent thumb movements. The implementation enables the user to adjust the processing parameters in an interactive and real-time mode, to save them as one of the profiles, or to select the most appropriate profile from the saved ones.

4. Test results

The processing modules were tested on the handset model 'Nexus 5X' with Android 7.1 OS. Qualitative evaluation was carried out using the headset of the handset for speech input through its microphone and audio output through its earphone. For objective evaluation, a PC sound card was used for applying the audio input and acquiring the processed output using a 4-pin TRRS connector to the headset port of the handset. The time taken for computation per frame was measured using 'Android device monitor', a profiling tool for Android OS. The average CPU time per frame for audio loopback (inclusive of FFT and IFFT operations) without any processing modules was measured as 0.45 ms. Total audio latency comprises the processing delay (sum of algorithmic delay and computational delay) and the I/O latency. The algorithmic delay is 25 ms (frame length of 20 ms and frameshift of 5 ms). The total audio latency was measured by applying a 1 kHz tone burst of 200 ms from a function generator as the input and observing the delay from onset of the input tone burst to the corresponding onset in the output, using a digital storage oscilloscope. It was found to be 20 ms for audio loopback and 45 ms for the app. The test results for two processing modules are given in following subsections.

4.1. Results for noise suppression

Informal listening and objective evaluation using perceptual evaluation of speech quality (PESQ) measure [19] was used for evaluation of the noise suppression module. The processing was also tested using the 30 sentences from

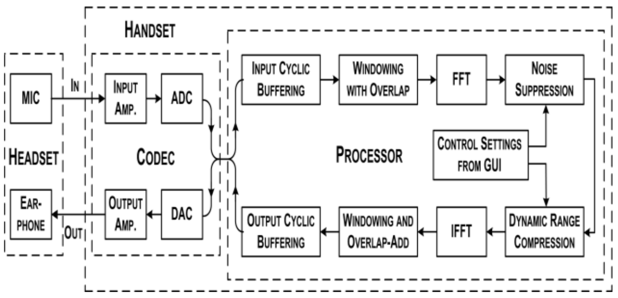

Figure 1: *Implementation of hearing aid app with noise suppression and dynamic range compensation.*

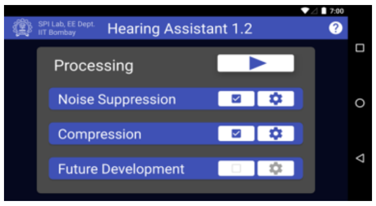

Figure 2: *Screenshot of the home screen of the app.*

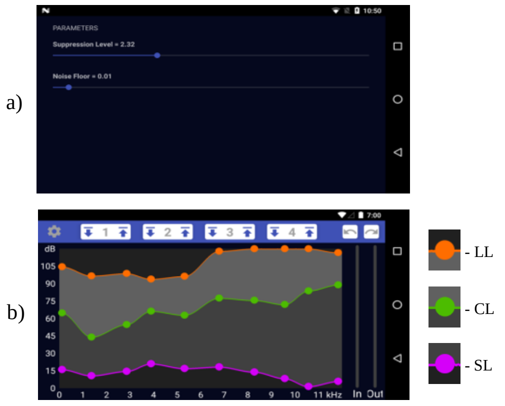

Figure 3: *Screenshot of the settings screen for a) noise suppression and b) dynamic range compression.*

NOIZEUS database [20] added with noises from AURORA database [21]. Airport, babble, car, street, and train station noises from AURORA database and white noise are added at SNR of 15, 12, 9, 6, 3, 0, and –3 dB to form noisy speech. The noisy speech was output from the PC soundcard and input to the handset using TRRS connector. The processed output was captured using 'Focusrite Scarlett 2i2' soundcard to avoid excessive noise observed while capturing audio using PC soundcard. The degradation in the PESQ score because of loopback through the device is less than 0.01. The processing parameters for noise suppression were set as $\lambda = 1/256$, $\tau_p = \tau_v = 0.1$ and $\sigma_p = \sigma_v = (0.9)^{1/1024}$. The histogram was estimated by tracking eight quantiles corresponding to p = 0.25, 0.30, 0.35,...., 0.60. Table 1 shows the PESQ improvement for different noises for 0, 3, and 6 dB SNR. It can be seen that the improvement in PESQ scores is in the range 0.17−0.32. The average time taken for computation per frame is measured as the difference of the average CPU time taken per block with and without noise suppression module. The average CPU time per block for noise suppression with spectral subtraction based on geometric approach was measured as 1.1 ms.

Table 1: *PESQ scores for unprocessed speech and improvement in scores by noise suppression for different types of noises and SNRs (test material: 30 sentences from NOIZEUS).*

Airport noise					**Babble noise**				
	Unproc. Score		Proc. Impr.			Unproc. Score		Proc. Impr.	
SNR	Mean	S. D.	Mean	S. D.	SNR	Mean	S. D.	Mean	S. D.
6 dB	2.21	0.15	0.29	0.16	6 dB	1.96	0.13	0.26	0.14
3 dB	2.01	0.17	0.33	0.16	3 dB	1.78	0.15	0.23	0.20
0 dB	1.81	0.18	0.35	0.18	0 dB	1.61	0.19	0.17	0.24
Car noise					**Street noise**				
	Unproc. Score		Proc. Impr.			Unproc. Score		Proc. Impr.	
SNR	Mean	S. D.	Mean	S. D.	SNR	Mean	S. D.	Mean	S. D.
6 dB	2.28	0.15	0.20	0.13	6 dB	2.28	0.15	0.27	0.15
3 dB	2.09	0.16	0.22	0.15	3 dB	2.08	0.17	0.30	0.16
0 dB	1.90	0.17	0.21	0.18	0 dB	1.86	0.19	0.35	0.17
Train station noise					**White noise**				
	Unproc. Score		Proc. Impr.			Unproc. Score		Proc. Impr.	
SNR	Mean	S. D.	Mean	S. D.	SNR	Mean	S. D.	Mean	S. D.
6 dB	2.52	0.14	0.22	0.14	6 dB	1.89	0.15	0.32	0.26
3 dB	2.33	0.15	0.27	0.14	3 dB	1.73	0.18	0.26	0.24
0 dB	1.92	0.19	0.34	0.17	0 dB	1.57	0.19	0.18	0.26

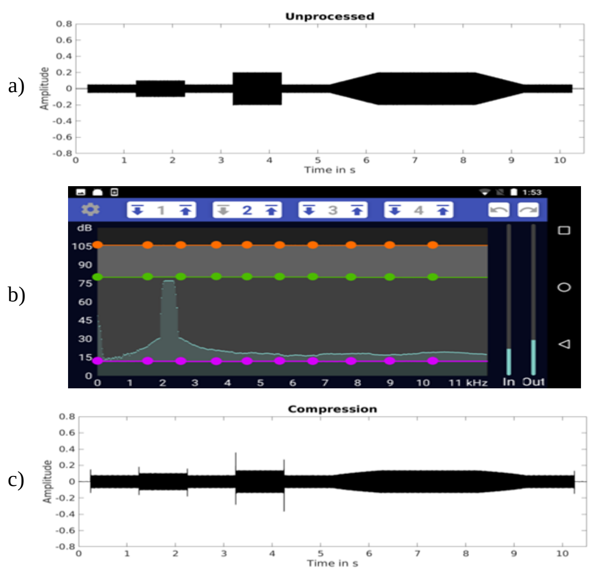

Figure 4: *Example of processing for dynamic range compression: (a) input signal of amplitude modulated tone of 1 kHz, (b) GUI parameters set for constant gain of 12 dB and compression ratio of 2, (c) processed output.*

4.2. Results for dynamic range compression

An example for dynamic range compression with amplitude modulated input is shown in Figure 4. Input is an amplitude-modulated tone of 1 kHz and processing parameters are set as shown in Figure 4 (b) with compression ratio of 2. The processing gives higher gains at lower values of the input level. Spikes in the amplitude envelope of the output signal in response to step changes in the amplitude envelope of the input signal, as seen in the figure, are typical of the dynamic range compression with a finite frame shift and can be eliminated by using one-sample frameshift but with a significantly increased computation load.

The app was further tested for speech modulated with different types of amplitude envelopes. An example of the processing is shown in Figure 5, for an amplitude modulated concatenation of speech signals. The input consists of three isolated vowels, a Hindi sentence, and an English sentence, (-/a/-/i/-/u/-“aaiye aap ka naam kya hai?” – “where were you a

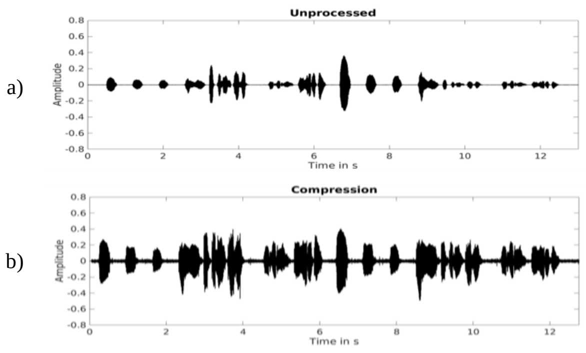

Figure 5: *Example of processing for dynamic range compression: (a) input; amplitude modulated VHSES speech, and (b) processed speech with parameters as shown in 3(b).*

year ago?”) referred to as VHSES material. An informal listening test was carried out with different speech materials, music, and environmental sounds with large variation in the sound level as inputs. The outputs exhibited the desired amplification and compression without introducing perceptible distortions. The average CPU time taken per block with compression module is measured to be 1.3 ms.

5. Conclusion

To enable the use of smartphone as a hearing aid, integration of the signal processing for dynamic quantile tracking based noise suppression and sliding-band dynamic range compression has been implemented using ‘LG Nexus 5X’ running ‘Android 7.1’. The processing parameters can be set by the user in an interactive and real-time mode using a graphical touch interface. The audio latency of the app is 45 ms, which is much less than the detectability threshold of 125 ms for audio-visual delay [22] and hence may be considered as acceptable for a hearing aid during face-to-face conversation. Implementation of the app on other smartphones and its use by a large number of hearing-impaired listeners is needed for real-life evaluation and further enhancement.

Acknowledgments

The research is supported by 'National Program on Perception Engineering Phase-II' sponsored by the Department of Electronics & Information Technology, Government of India.

References

[1] H. Levitt, J. M. Pickett, and R. A. Houde (eds.), *Senosry Aids for the Hearing Impaired.* New York: IEEE Press, 1980.

[2] J. M. Pickett, *The Acoustics of Speech Communication: Fundamentals, Speech Perception Theory, and Technology*. Boston, Mass.: Allyn Bacon, 1999, pp. 289–323.

[3] B. C. J. Moore, *An Introduction to the Psychology of Hearing*, London, UK: Academic, 1997, pp. 66–107.

[4] S. A. Gelfand, *Hearing: An Introduction to Psychological and Physiological Acoustics*. New York: Marcel Dekker, 1998, pp. 314–318.

[5] D. O' Shaughnaessy, *Speech Communications: Human and Machine*. Hyderabad, India: Univ. Press, 2001, pp. 127–128.

[6] H. Dillon, *Hearing Aids*. New York: Thieme Medical, 2001.

[7] R. E. Sandlin, *Textbook of Hearing Aid Amplification*. San Diego, Cal.: Singular, 2000, pp. 210–220.

[8] D.Byrne, W. Tonnison, "Selecting the gain of hearing aids for persons with sensorineural hearing impairments," *Scandinavian Audiology*, vol. 5, pp. 51–59, 1976.

[9] N. Tiwari and P. C. Pandey, "A sliding-band dynamic range compression for use in hearing aids," in *20th National Conference on Communication (NCC 2014)*, Kanpur, India, doi: 10.1109/NCC.2014. 6811300.

[10] P. C. Pandey and N. Tiwari, "Dynamic range compression with low distortion for use in hearing aids and audio systems," US Patent No. 9672834, 2017.

[11] S. F. Boll, "Suppression of acoustic noise in speech using spectral subtraction," *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. 27, no. 2, pp. 113-120, 1979.

[12] N. Tiwari and P. C. Pandey, "Speech enhancement using noise estimation based on dynamic quantile tracking for hearing impaired listeners," in *21st National Conference on Communication (NCC 2015)*, Mumbai, India, 2015, paper no. 1570056299.

[13] P. C. Pandey and N. Tiwari, "Method and system for suppressing noise in speech signals in hearing aids and speech communication devices, US Patent application publication no. US 2017/0032803 A1, 2017.

[14] S. Sharma, N. Tiwari, and P. C. Pandey, "Implementation of a digital hearing aid with user-settable frequency response and sliding-band dynamic range compression as a smartphone app," in *8th International Conference on Intelligent Human Computer Interaction (IHCI 2016)*, Pilani, India, Dec. 12 - 13, 2016, paper no. 81.

[15] H. Hirsch, C. Ehrlicher, "Noise estimation techniques for robust speech recognition," in *International Conference on Acoustics, Speech, and Signal Processing (ICASSP 1995)*, Detroit, Michigan, 1995, pp. 153-156.

[16] N. Tiwari and P. C. Pandey, "A technique with low memory and computational requirements for dynamic tracking of quantiles," *Journal of Signal Processing Systems*, Epub. 2018 Jan 08, DOI: 10.1007/s11265-017-1327-6

[17] Y. Lu and P. C. Loizou, "A geometric approach to spectral subtraction," *Speech Communication*, vol. 50, pp. 453–466, 2008.

[18] Y. Lu and P. C. Loizou, Speech enhancement code. [online]. Available: http://ecs.utdallas.edu/loizou/speech/GA_code.zip

[19] ITU, "Perceptual evaluation of speech quality (PESQ): an objective method for end-to-end speech quality assessment of narrow-band telephone networks and speech codecs," ITU-T Rec., P.862, 2001.

[20] A. Varga and H. J. M. Steeneken, "Assessment for automatic speech recognition: II. NOISEX-92: A database and an experiment to study the effect of additive noise on speech recognition systems," *Speech Commun*ication, vol. 13, no. 3, pp. 247–251, 1993.

[21] H. Hirsch and D. Pearce, "The AURORA experimental framework for the performance evaluation of speech recognition systems under noisy conditions," in *ISCA Tutorial and Research Workshop on Automatic Speech Recognition: Challenges for the New Millenium 2000, (ASR 2000)*, Paris, France, pp. 181–188.

[22] International Telecommunication Union: Relative timing of sound and vision for broadcasting, ITU Rec. ITU-R BT.1359 1998.