Towards GCC-NMF Speech Enhancement for Hearing Assistive Devices: Reducing Latency with Asymmetric Windows

Sean UN Wood, Jean Rouat

NECOTIS, Department of Electrical and Computer Engineering
Université de Sherbrooke, Québec, Canada

sean.wood@usherbrooke.ca, jean.rouat@usherbrooke.ca

Abstract

We present a modified version of the real-time GCC-NMF stereo speech enhancement algorithm that drastically reduces the inherent system latency by incorporating an asymmetric STFT windowing strategy. Long analysis windows retain the high spectral resolution required by GCC-NMF, while short synthesis windows significantly reduce the overall system latency. We show that GCC-NMF speech enhancement quality is relatively unaffected by this windowing strategy, with the overall objective PEASS score remaining stable for varying system latencies. The asymmetric windowing technique comes at a cost of increased computational load, with shorter synthesis windows requiring a shorter frame advance, thus increasing the number of windows to be processed. We present an analysis of the computational requirements of GCC-NMF to run in real-time on a variety of hardware platforms including the Raspberry Pi and the NVIDIA Jetson TX1. All tested systems are fast enough to achieve latencies at least as low as 24 ms with small NMF dictionaries of 64 atoms, while the fastest NVIDIA K40 GPU system is capable of achieving 6 ms latency with a large dictionary of 1024 atoms.

Index Terms: real-time, latency, speech enhancement, source separation, GCC-NMF, GCC, NMF, GCC-PHAT, CASA

1. Introduction

A wealth of speech enhancement algorithms designed to suppress noise and reverberation have been developed in fields such as speech coding, automatic speech recognition, and source separation. Many such algorithms, however, remain inapplicable in the context of hearing assistive devices due to both inherent algorithmic latency and computational performance on low-power hardware. We address these hurdles here with respect to the real-time GCC-NMF speech enhancement algorithm we introduced recently [1, 2].

Many speech enhancement algorithms including GCC-NMF are built around the short-time Fourier transform (STFT) with which sound is processed in short, overlapping segments of time [3]. A consequence of the traditional STFT is an inherent algorithmic latency where, independent of processing speed, there exists a trade-off between spectral resolution and the delay between the system's input and output. With many algorithms relying on high spectral resolution, latencies greater than 64 ms are common. In the context of assistive listening devices, however, such high latencies are perceived as objectionable echoes as a superposition of both the aided and unaided sounds are heard by the listener [4]. Depending on the type and severity of hearing loss, delays below 15 to 32 ms are likely required to be tolerable [5, 6], with delays less than 10 ms being a reasonable objective in the general case [7, 8].

In this work, we integrate the asymmetric STFT windowing approach proposed by Mauler and Martin [9] into the GCC-NMF speech enhancement system, simultaneously providing high spectral resolution and latencies well below 10 ms, depending on available computational power. An alternative approach to low delay speech enhancement was developed by Löllmann and Vary using low delay filter banks [10, 11]. We begin with a review of the real-time GCC-NMF speech enhancement algorithm in Section 2, followed by a description of the asymmetric STFT windowing method in Section 3. We then demonstrate the robustness of GCC-NMF speech enhancement quality to latency reduction with asymmetric windowing, as well as an analysis of the computational requirements of GCC-NMF on a variety of hardware platforms in Section 4, followed by the conclusion in Section 5.

2. Real-time GCC-NMF

The GCC-NMF stereo speech enhancement algorithm combines the non-negative matrix factorization (NMF) unsupervised dictionary learning algorithm [12] with the generalized cross-correlation (GCC) spatial localization method [13]. GCC-NMF is flexible in terms of microphone separation, where separations ranging from 5 cm to 1 m have been tested previously [1]. NMF provides a parts-based representation of the input mixture signal in terms of *dictionary atoms* in the magnitude frequency domain, while GCC provides an estimate of the time delay of arrival (TDOA) of each dictionary atom, at each point in time. The NMF dictionary atoms estimated to originate from the direction of interest are recombined and used to construct a Wiener-like filter, as is typical for NMF-based speech enhancement [14]. The resulting filter is applied to the mixture signal to yield the system output. For offline speech enhancement, the NMF dictionary may be learned directly from the mixture signal, while in the online case, it is pre-learned from isolated speech and noise signals using a different dataset than used at test time, generalizing to new speakers, acoustic and noise conditions, and recording setups [2].

Online GCC-NMF speech enhancement is performed on a frame-by-frame basis given the pre-learned NMF dictionary W_{fd} (with f indexing frequency and d indexing the dictionary atoms), the complex-valued left and right Fourier-transformed frames V_{lf} and V_{rf}, a set of possible TDOAs indexed by τ, and the target direction τ_s estimated using an accumulated GCC-PHAT localization process [2]. First, the GCC-NMF angular spectrum $G^{\mathrm{NMF}}_{d\tau}$, is constructed for each dictionary atom,

$$G^{\mathrm{NMF}}_{d\tau} = \sum_f W_{fd} \,\mathrm{Re}\left(\frac{V_{lf} V^*_{rf}}{|V_{lf}|\,|V_{rf}|} e^{j2\pi f \tau}\right) \qquad (1)$$

where for a given atom d, $G^{\mathrm{NMF}}_{d\tau}$ is a function of τ that will be

high for values of τ near the estimated direction of arrival. A binary atom mask M_d is then constructed given the estimated target direction τ_s,

$$M_d = \begin{cases} 1 & \text{if } \left|\tau_s - \text{argmax}_\tau G_{d\tau}^{\text{NMF}}\right| < \epsilon/2 \\ 0 & \text{otherwise} \end{cases} \quad (2)$$

such that only atoms whose GCC-NMF angular spectrum reaches its peak within a window of size ϵ of the target TDOA are accepted. Finally, a Wiener-like filter is constructed as the ratio of the sum of unmasked atoms $\sum_d W_{fd}M_d$ to the sum of all atoms $\sum_d W_{fd}$, and is used to filter input signals resulting in the enhanced speech signal $\widehat{X}_{cf}$, where c indexes the left and right channels,

$$\widehat{X}_{cf} = \frac{\sum_d W_{fd}M_d}{\sum_d W_{fd}} V_{cf} \quad (3)$$

With the online GCC-NMF speech enhancement algorithm now defined, we proceed to show how the underlying STFT imposes an lower limit on its real-time latency, and how the asymmetric STFT windowing method mentioned previously may be used to drastically reduce this latency.

3. Asymmetric STFT Windowing

3.1. STFT and Latency

The STFT processes sound in short, overlapping segments of time called *frames*. Each frame is multiplied by an *analysis window* prior to computing its Fourier transform. Resynthesis is achieved by taking the inverse Fourier transform of the transformed frame, multiplying the resulting samples by a *synthesis window*, and combining neighbouring frames via the overlap-add (OLA) method. Perfect reconstruction can be achieved if the transform has the constant overlap-add (COLA) property, i.e. if the overlapped sum of the product of the analysis and synthesis windows is constant over time [15]. A commonly used window for analysis and synthesis is the square root of the periodic Hann window, where the periodic Hann function is defined for frame size N as,

$$\text{H}_N[n] = \begin{cases} \frac{1}{2}\left(1 - \cos\left(2\pi\frac{n}{N}\right)\right) & 0 \le n < N \\ 0 & \text{otherwise} \end{cases} \quad (4)$$

The above process of overlapped signal windowing with OLA resynthesis induces a latency L_{OLA} equal to the window size N. In order to run in real-time, all processing including the Fourier transform and its inverse, should occur within a single frame advance R, resulting in a total system latency of $N + R$ [15]. We previously presented the real-time GCC-NMF separation system on input signals sampled at 16 kHz, with a window size of 1024 samples with varying frame advance, resulting in a total latency of 64 ms plus 8 ms with a frame advance of 128 samples, for example. As described in Section 1, latencies this large are unsuitable for real-world use in hearing assistive devices.

A first approach to reduce the GCC-NMF system latency is to simply reduce the window size N. This comes at the expense of decreasing the spectral resolution, however, and as we will show in Section 4.2, GCC-NMF speech enhancement quality decreases significantly for small window sizes with this approach. We therefore present another approach to latency reduction based on an asymmetric STFT windowing method that combines long analysis windows with short synthesis windows.

3.2. Asymmetric STFT windowing

Departing from the tradition of symmetric analysis and synthesis windows that have the same duration, asymmetric windowing allows us to simultaneously achieve high spectral resolution and low latency by combining long analysis windows with relatively short synthesis windows. The asymmetric windows we use in this work have been adapted from the more general case proposed by Mauler and Martin [9], though other asymmetric windowing approaches can be found in the literature [16, 17, 18, 19].

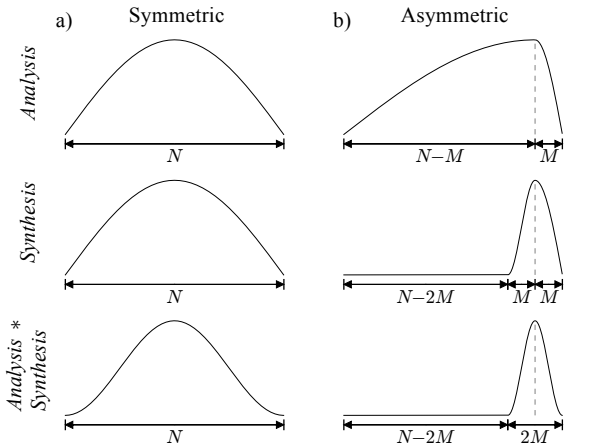

Figure 1: *Comparison of the symmetric and asymmetric STFT window functions for frame size N. a) Traditional symmetric square root Hann analysis and synthesis window functions and their product Hann window, all having duration N. b) Asymmetric window functions, where the analysis window has duration N and is weighted towards the right, while the synthesis window has duration $2M<N$, and shares its right edge with the underlying frame. The resulting product of the analysis and synthesis windows is a Hann window of size $2M$ that also shares its right edge with the underlying frame.*

For a given frame size N, the asymmetric analysis and synthesis windows are designed such that their point-wise product is a Hann window of size $2M < N$. This Hann window shares its right edge with the underlying frame, and can be made to be much shorter than the frame itself by choosing $2M \ll N$, as depicted in Figure 1. The analysis window h_A and the synthesis window h_S are defined mathematically as,

$$h_A[n] = \begin{cases} \sqrt{\text{H}_{2(N-M)}}[n] & 0 \le n < N-M \\ \sqrt{\text{H}_{2M}}[n-(N-2M)] & N-M \le n < N \\ 0 & \text{otherwise} \end{cases} \quad (5)$$

$$h_S[n] = \begin{cases} \frac{\sqrt{\text{H}_{2M}}[n-(N-2M)]}{\sqrt{\text{H}_{2(N-M)}[n]}} & N-2M \le n < N-M \\ \sqrt{\text{H}_{2M}}[n-(N-2M)] & N-M \le n < N \\ 0 & \text{otherwise} \end{cases} \quad (6)$$

These window functions are constructed in two parts with respect to the center of the analysis-synthesis product Hann window, i.e. $N-M$. To the right of $N-M$, both analysis and synthesis windows consist of the right half of a square root Hann window of size $2M$. To the left, the analysis window consists of the left half of a Hann window of size $N-M$, while

the synthesis window is defined as the ratio of the analysis window and the product Hann window, limited to the range $N-2M \leq n < N-M$.

In Figure 2, we compare the traditional STFT windowing method using square root Hann windows with the asymmetric STFT windowing presented above. The analysis window size is N in both cases, and the asymmetric synthesis window size is set to $N/4$, i.e. with $M=N/8$. In both cases, the window overlap is 50% of the synthesis window, such that perfect reconstruction (PR) is achieved. We note that retaining the relative synthesis window overlap while decreasing the synthesis window size results in a significant increase in the number of windows required for the overlap-add windowing process, thus increasing the computational load. We also note that this approach increases the start-up latency of the STFT, though this may mitigated by simply pre-padding the signal with zeros.

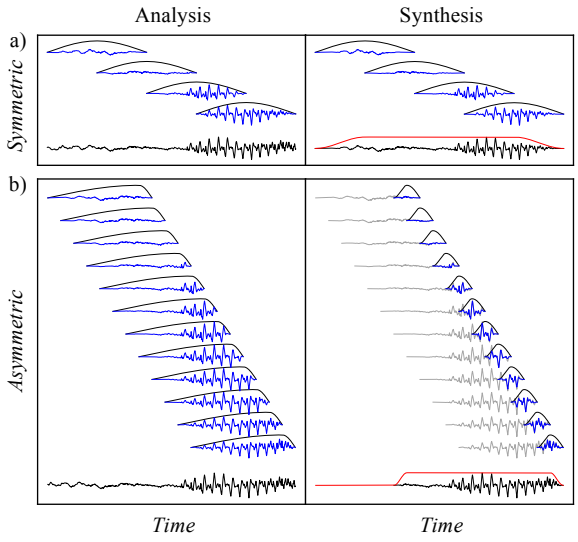

Figure 2: *Comparison of analysis and synthesis processes for a) symmetric and b) asymmetric windowing functions, given a synthesis window overlap of* 50%*. For the analysis stage (left), the input signal and time-shifted analysis window functions are shown in black, with the resulting windowed frames shown in blue. For synthesis stage (right), the time-shifted synthesis window functions shown in black, the reconstructed frames are shown in gray, and the resulting windowed frames used for overlap-add resynthesis are overlaid in blue. The overlap-add result is then shown below in black, with the normalized overlapped sum of the anslysis-synthesis window products in red.*

4. Experiments

In this section, we first compare the effect of latency reduction using the symmetric and asymmetric windowing methods on the learned NMF dictionary atoms, followed by the effect on GCC-NMF speech enhancement quality. We then study the empirical processing time requirements of GCC-NMF for a variety of hardware platforms, to determine the conditions under which the proposed low-latency system may be run in real-time on currently available hardware.

We reuse here the data and evaluation metrics as presented in the development of the real-time GCC-NMF speech enhancement algorithm [2]. Unsupervised training data consists of a small subset of the speech and noise signals from the CHiME challenge [20], taken as 4096 randomly chosen frames divided equally between speech and noise signals from a single microphone. Evaluation data consists of the two-channel mixtures of speech and real-world noise from the SiSEC speech enhancement challenge [21], where the microphones are separated by 8.6 cm, though as mentioned previously, GCC-NMF has been tested for microphone separations ranging from 5 cm to 1 m [1]. Both datasets are sampled at 16 kHz. Speech enhancement quality is quantified with the Perceptual Evaluation methods for Audio Source Separation (PEASS) toolkit [22], designed to better correlate with subjective assessments than the traditional SNR-based metrics. PEASS metrics consist of four scores quantifying the overall enhancement quality, target fidelity, interference suppression, and lack of perceptual artifacts, where higher scores are better for all measures. Future work will include a wider range of evaluations metrics including measures of speech intelligibility, STOI [23] and ESTOI [24]. The default NMF dictionary size for the experiments that follow is 1024, while the default STFT synthesis window overlap is 75%.

4.1. Effect on NMF dictionary atoms

As described in Section 3.1, the inherent latency of the real-time GCC-NMF speech enhancement algorithm using traditional symmetric STFT windowing may be reduced by simply reducing the STFT frame size N. An undesired consequence of this approach, however, is a reduction spectral resolution, as decreasing the STFT frame size results in increasingly wideband spectrograms. In Figure 3a), we depict example NMF dictionary atoms learned for varying symmetric STFT window size, noting that as the window size is decreased, dictionary atoms become increasing wideband, and the spectral details captured with longer duration windows are lost.

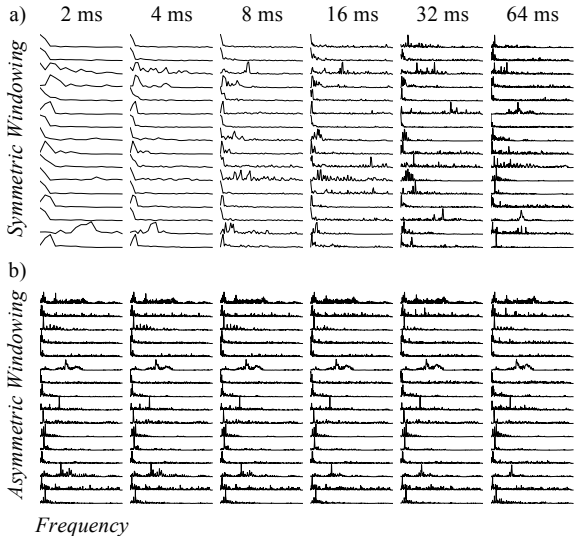

Figure 3: *Example NMF dictionary atoms learned for varying STFT synthesis window size for a) symmetric windowing and b) asymmetric windowing. For each window size, a subset of 16 randomly chosen dictionary atoms are shown from a total of 1024. For symmetric windowing, the analysis window length decreases with the synthesis window, while for asymmetric windowing, the analysis window size remains fixed at 64 ms, with only its shape changing as a function of synthesis window size.*

Contrary to the traditional windowing approach, asymmetric windowing allows us to retain the long-duration analysis

windows while decreasing the synthesis window size. As the synthesis window size $2M$ is reduced, the analysis window size remains fixed at the frame size N with its shape increasingly weighted towards the future, as we showed in Figure 1b). In Figure 3b), we present example NMF atoms learned using the asymmetric window approach for varying synthesis window size, where the learned NMF atoms are shown to retain spectral detail, regardless of synthesis window size. As identical training data and random seed is used in all cases, the resulting atoms remain very similar across synthesis window sizes, with only subtle differences in the learned dictionary atoms resulting from the different analysis window shapes.

4.2. Effect on speech enhancement quality

In Figure 4a), we present the PEASS scores on the SiSEC speech enhancement dataset as a function of STFT window size for the symmetric windowing case. We first note that the overall enhancement performance decreases with decreasing window size, with a significant drop in performance for window sizes less than 8 ms. This is likely due to a decreased separability of speech and noise sources with the wideband NMF atoms shown above, resulting in decreased quality of the resulting GCC-NMF speech enhancement. We also note a drastic trade-off between interference suppression and lack of artifacts, where smaller window sizes result in increased interference suppression at the cost of significant artifacts.

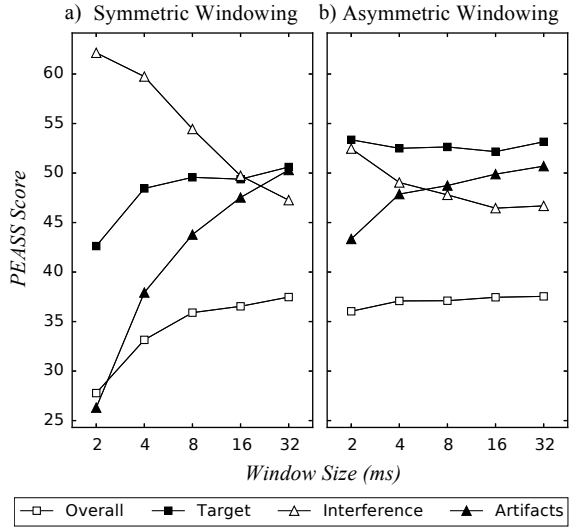

Figure 4: *Effect of STFT synthesis window size on GCC-NMF speech enhancement performance for a) symmetric windowing and b) asymmetric windowing with a fixed analysis window of 64 ms. The PEASS scores correspond to objective measures of overall enhancement quality, target fidelity, interference suppression, and lack of artifacts, where higher scores are better in all cases.*

In Figure 4b), we present the effect of latency on PEASS scores for the SiSEC dataset for the asymmetric windowing approach. The analysis window here is kept fixed at 1024 samples at 16 kHz (64 ms), while the synthesis window size is varied from 512 to 32 samples (32 to 2 ms), with an overlap of 75% of the synthesis window used in each case. We note that the overall PEASS score remains relatively constant for varying synthesis window size, with only a slight reduction for synthesis windows as short as 2 ms. We also note the same trade-off between interference suppression and lack of artifacts as with symmetric windowing, though it is much more tempered for the asymmetric windowing approach. Finally, we note that the target fidelity is consistently higher for the asymmetric windowing case, and remains relatively constant for varying synthesis window size. These results demonstrate that the proposed asymmetric windowing approach is a viable solution to reduce the latency of real-time GCC-NMF to values well below the threshold required for hearing devices while maintaining the enhancement quality of the higher latency symmetric windowing approach.

4.3. Latency and GCC-NMF processing time

We now proceed to study the computational requirements of the GCC-NMF speech enhancement algorithm with asymmetric windowing to determine the conditions under which it may be executed in real-time. As we saw in Section 3.2, the inherent latency of the asymmetric STFT process is equal to the duration of the synthesis window plus the frame advance. For speech enhancement to be performed in real-time, the system must then process a single frame within the time of a single frame advance. This processing time includes the windowing processes, the forward FFT, the GCC-NMF speech enhancement processing itself, the inverse FFT, and the OLA summation.

In Figure 5a), we present the average measured processing time of the online GCC-NMF enhancement algorithm for a *single frame* as a function of the NMF dictionary size, for a variety of hardware platforms. We note that the processing time increases approximately linearly with dictionary size, with the slope varying between hardware platforms. On all systems presented, processing times less than 8 ms are possible, provided a small enough dictionary is used, where enhancement performance decreases smoothly with decreasing dictionary size as we have shown previously [2].

In Figure 5b), we depict the relationship between system latency and available processing time for a single frame, as a function of synthesis window size and overlap. Decreasing either the synthesis window size or the frame advance decreases the system latency at a cost of decreased available processing time. We may combine this information with Figure 5a) to determine, for a given hardware system and dictionary size, the available synthesis window size and overlap values (and resulting latencies), in order for the system to run in real-time. All systems prove fast enough for a synthesis window size of 16 ms with 50% overlap and a dictionary size of 64, resulting in a latency of 24 ms. All systems except the Raspberry Pi may achieve 12 ms latency for small to moderate dictionary sizes, with a window size of 8 ms and 50% overlap. The fastest system (Tesla K40 GPU) can achieve 6 ms latency for dictionaries at least as large as 1024 atoms. These results demonstrate that it is possible to achieve latencies suitable for hearing assistive devices with real-time speech enhancement with GCC-NMF using the asymmetric windowing technique. While these results are promising, the hardware platforms tested remain significantly more powerful than those found in currently available hearing aids. Future work will therefore involve additional implementation optimizations in order to run the system with larger dictionaries on even lower-power devices.

5. Conclusion

We have presented an approach to reducing latency in the real-time GCC-NMF speech enhancement system by incorporating

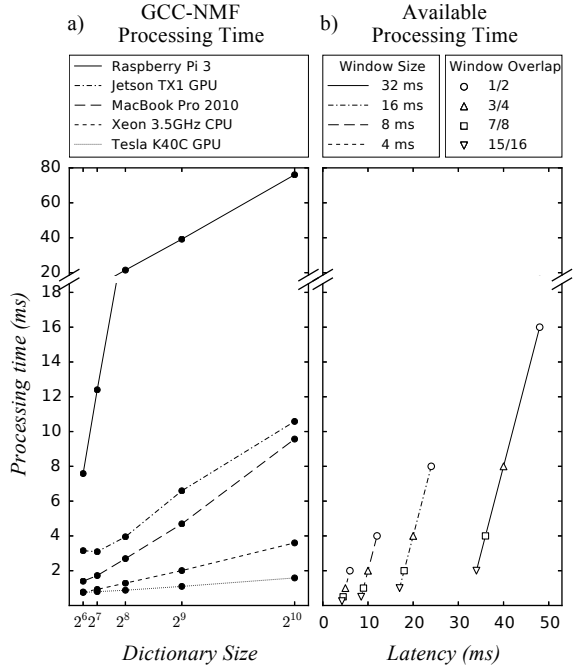

Figure 5: *Real-time GCC-NMF computational requirements with the asymmetric STFT windowing technique, with a) Effect of dictionary size on GCC-NMF mean empirical processing time for a single frame on various hardware platforms given an analysis window size of 64 ms, and b) available processing time for a single frame, given the asymmetric STFT windowing approach, presented for varying synthesis window size and overlap, with the resulting latency as the horizontal axis.*

an asymmetric STFT windowing technique. This asymmetric windowing method provides long duration analysis windows as with the traditional symmetric window approach, maintaining the high spectral resolution required by GCC-NMF, but uses short synthesis windows in order to drastically reduce system latency. We have shown that while speech enhancement performance suffers for the traditional symmetric windowing method when decreasing system latency using shorter windows, the asymmetric windowing approach results in relatively constant performance across a wide range of synthesis window sizes as short as 2 ms given an analysis window size of 64 ms. The computational requirements of the online GCC-NMF algorithm were presented for a variety of hardware platforms including the Raspberry Pi and NVIDIA Jetson TX1, and it was shown that the system may run in real-time with latencies below 24 ms on all platforms, provided the NMF dictionary size is adapted to the computational capabilities of the hardware. Moderately powerful hardware may achieve 12 ms latency with moderate dictionary sizes, while for the most powerful hardware we tested, a Tesla K40 GPU, latencies as low as 6 ms are possible with a large NMF dictionary of 1024 atoms. Source code for this work will be made available at `https://www.github.com/seanwood/gcc-nmf`.

6. Acknowledgements

The authors would like to thank the NSERC discovery grant, and FQRNT (CHIST-ERA, IGLU) for funding our research, and NVIDIA for contributing the Tesla K40 GPU. We would also like to thank William E Audette for inspiring discussions that lead directly to this work, and the anonymous reviewers for their thoughtful feedback.

7. References

[1] S. U. N. Wood, J. Rouat, S. Dupont, and G. Pironkov, "Blind speech separation and enhancement with GCC-NMF," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 25, no. 4, pp. 745–755, 2017.

[2] S. U. N. Wood and J. Rouat, "Real-time speech enhancement with GCC-NMF," in *Interspeech 2017*, 2017.

[3] P. C. Loizou, *Speech enhancement: theory and practice*. CRC press, 2013.

[4] M. A. Stone and B. C. Moore, "Tolerable hearing aid delays. I. estimation of limits imposed by the auditory path alone using simulated hearing losses," *Ear and Hearing*, vol. 20, no. 3, p. 182, 1999.

[5] M. A. Stone and B. C. J. Moore, "Tolerable hearing-aid delays: IV. effects on subjective disturbance during speech production by hearing-impaired subjects," *Ear and Hearing*, vol. 26, no. 2, pp. 225–235, 2005.

[6] R. Herbig and J. Chalupper, "Acceptable processing delay in digital hearing aids," *Hearing Review*, vol. 17, no. 1, pp. 28–31, 2010.

[7] J. Agnew and J. M. Thornton, "Just noticeable and objectionable group delays in digital hearing aids," *JOURNAL-AMERICAN ACADEMY OF AUDIOLOGY*, vol. 11, no. 6, pp. 330–336, 2000.

[8] H. Dillon, G. Keidser, A. O'brien, and H. Silberstein, "Sound quality comparisons of advanced hearing aids," *The hearing journal*, vol. 56, no. 4, pp. 30–32, 2003.

[9] D. Mauler and R. Martin, "A low delay, variable resolution, perfect reconstruction spectral analysis-synthesis system for speech enhancement," in *Signal Processing Conference, 2007 15th European*, 2007, pp. 222–226.

[10] H. W. Löllmann and P. Vary, "Low delay filter-banks for speech and audio processing," *Speech and audio processing in adverse environments*, pp. 13–61, 2008.

[11] ——, "Low delay noise reduction and dereverberation for hearing aids," *EURASIP Journal on advances in signal processing*, vol. 2009, no. 1, p. 437807, 2009.

[12] D. D. Lee and H. S. Seung, "Learning the parts of objects by non-negative matrix factorization," *Nature*, vol. 401, no. 6755, pp. 788–791, 1999.

[13] C. H. Knapp and G. C. Carter, "The generalized correlation method for estimation of time delay," *Acoustics, Speech and Signal Processing, IEEE Transactions on*, vol. 24, no. 4, pp. 320–327, 1976.

[14] N. Mohammadiha, P. Smaragdis, and A. Leijon, "Supervised and unsupervised speech enhancement using nonnegative matrix factorization," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 21, no. 10, pp. 2140–2151, 2013.

[15] J. O. Smith III, *Spectral audio signal processing*. W3K publishing, 2011.

[16] E. Allamanche, R. Geiger, J. Herre, and T. Sporer, "MPEG-4 low delay audio coding based on the AAC codec," in *Audio Engineering Society Convention 106*. Audio Engineering Society, 1999.

[17] M. Schnell, M. Schmidt, M. Jander, T. Albert, R. Geiger, V. Ruoppila, P. Ekstrand, and G. Bernhard, "MPEG-4 enhanced low delay AAC-a new standard for high quality communication," in *Audio Engineering Society Convention 125*. Audio Engineering Society, 2008.

[18] L. Su and H.-t. Wu, "Minimum-latency time-frequency analysis using asymmetric window functions," *arXiv preprint arXiv:1606.09047*, 2016.

[19] K. T. Andersen and M. Moonen, "Adaptive time-frequency analysis for noise reduction in an audio filter bank with low delay," *IEEE/ACM Transactions on Audio, Speech and Language Processing (TASLP)*, vol. 24, no. 4, pp. 784–795, 2016.

[20] E. Vincent, S. Watanabe, A. A. Nugraha, J. Barker, and R. Marxer, "An analysis of environment, microphone and data simulation mismatches in robust speech recognition," *Computer Speech & Language*, 2016.

[21] A. Liutkus, F.-R. Stöter, Z. Rafii, D. Kitamura, B. Rivet, N. Ito, N. Ono, and J. Fontecave, "The 2016 signal separation evaluation campaign," in *International Conference on Latent Variable Analysis and Signal Separation*. Springer, 2017, pp. 323–332.

[22] V. Emiya, E. Vincent, N. Harlander, and V. Hohmann, "Subjective and objective quality assessment of audio source separation," *Audio, Speech, and Language Processing, IEEE Transactions on*, vol. 19, no. 7, pp. 2046–2057, 2011.

[23] C. H. Taal, R. C. Hendriks, R. Heusdens, and J. Jensen, "An algorithm for intelligibility prediction of time–frequency weighted noisy speech," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 19, no. 7, pp. 2125–2136, 2011.

[24] J. Jensen and C. H. Taal, "An algorithm for predicting the intelligibility of speech masked by modulated noise maskers," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 24, no. 11, pp. 2009–2022, 2016.