Source Separation, Removal, and Resynthesis Using

Azimuth-based Source Separation

Jonathan P. Forsyth

Submitted in partial fulfillment of the requirements for the

Master of Music in Music Technology

in the Department of Music and Performing Arts Professions

in The Steinhardt School

New York University

Advisor: Juan Pablo Bello

08/13/2008
1. Introduction

Commercially available music recordings typically consist of individual tracks mixed down to a two-track (stereo) mix. Often, each instrument is recorded on a separate track. A multi-track recording thus allows, among other things, for the removal of a particular instrument from the mix. However, in a typical stereo mix, it is not possible to isolate individual instruments. A source removal system, that is, a system capable of isolating and removing individual instruments from a stereo mix, could find a number of uses. For example, such a system would allow users to remove unwanted instruments from one stereo recording and mix in separate instruments from other stereo recordings. This functionality would be of use to those interested in creating re-mixes and “mash-ups” of existing stereo recordings. The system described could also be used to remove a soloing instrument from an existing recording, thereby creating a recording a soloist could use for practice purposes. Similarly, one could remove a vocal line from an existing recording in order to create recordings for use in a karaoke system. A source removal system could also be applied to problems in automatic music transcription and music information retrieval.

A major component to any source removal system would necessarily be some method by which a particular instrument is separated from the original recording. This general task is referred to as sound source separation or audio source separation, and can be defined as “the process of extracting the underlying [audio] sources” (O’Grady et al. 2005, p. 18) from a set of observations (i.e., recordings). Blind source separation attempts to estimate the sources “without strong additional information about the
individual sources or constraints on the mixing process” (O’Grady et al. 2005, p. 18). The concept of what constitutes a source is not necessarily well defined, however. A source may be defined as an individual instrument or a group of instruments playing in unison (Virtanen 2006, p. 268). The former is an example of a physical source, a source corresponding to an individual “vibrating physical entity” (Virtanen 2006, p. 268); the latter is an example of a perceptual source, a source corresponding to “what humans tend to perceive as a single source” (Virtanen 2006, p. 268). The appropriate definition of a source depends on the algorithm employed to produce it and on the specific application (Virtanen 2006, p. 268).

The source separation problem is a difficult one, and thus “has been investigated in audio signal processing for many years” (Wang and Brown 2006, p. 28). A variety of solutions have been proposed (Wang and Brown 2006; Virtanen 2006; O’Grady et al. 2005). However, “the performance of existing [sound separation] algorithms is still quite limited compared to the human auditory system” (Virtanen 2006, p. 267). The nature of music in general contributes greatly to the difficulty of the problem, due to the fact that in a polyphonic texture, there is a great deal of overlapping, in time and in frequency, of the harmonics of the individual instruments, thus disturbing any estimation of an individual instrument (Virtanen 2006, p. 267; Barry et al. 2004, p. 3).

The source removal system described in this paper uses an existing source separation algorithm, ADResS, to perform the source separation task (Barry et al. 2004). This algorithm has the advantages of being both successful at source separation and relatively easy to implement. Once the ADResS algorithm computes the separated signal, it is then converted into a binary time-frequency mask. This mask is used to remove the
separated source from the original signal. The system employs two different methods of improving the quality of the initial time-frequency mask. In addition, the system uses a modified version of the source separation algorithm to produce the output signal (i.e., the original signal minus the separated source).

This paper will describe a number of common source separation algorithms, describe in detail a source removal system, evaluate the relative sound qualities of the outputs of a Matlab implementation of the system, and propose possible future improvements.
2. Techniques for Sound Source Separation

There are a wide variety of solutions to the sound source separation problem. The classes of source separation techniques are not mutually exclusive; different techniques are used as part of a larger source separation strategy (Wang & Brown 2006). In general, though, they fall into several categories. For example, some approaches to the problem attempt to model some aspect of human auditory perception or physiology; these methods seem to be viewed as belonging to the domain of computational auditory scene analysis (Wang & Brown 2006).

Other methods rely on certain assumptions about the nature of the recording to perform source separation. For example, beamforming techniques require a microphone array consisting of \( N + 1 \) microphones, arranged in a specific pattern, to separate \( N \) sources (Feng & Jones 2006, pp. 187 – 188). Binaural-based separation techniques make use of the features of human physiology and the resulting interaural time difference (ITD) and interaural intensity difference (IID) effects (Feng & Jones 2006; Stern et al. 2006). Therefore, this method is only applicable in situations where the sensors are arrayed in a manner consistent with human physiology. Music recordings do not often conform to this model, however.

In addition, there are methods that make use of various machine learning techniques (both supervised and unsupervised) to solve the problem. This section will describe some of the more significant unsupervised learning techniques that have had success separating music signals (Virtanen 2006, p. 268). Unsupervised learning refers to “algorithms which try to separate and learn the structure of sources in mixed data
based on information-theoretical principles, such as statistical independence between sources, instead of sophisticated modeling of the source characteristics or human auditory perception” (Virtanen 2006, p. 267). The unsupervised learning methods discussed below have all had some success at the task of source separation (Virtanen 2006, p. 268).

In addition, this section will describe ADRess, a source separation algorithm that takes advantage of the differing intensity levels between the left and right channels of a stereo recording (Barry et al. 2004). The ADRess algorithm differs from some of the other source separation algorithms in that it is not based on machine learning principles and does attempt to decompose a signal into its constituent components. Rather, it is designed to separate a particular source within a stereo mix, based on information provided by the user. While not as sophisticated as some of the other source separation methods, the ADRess algorithm has the advantage of being relatively easy to implement and quite successful for a variety of stereo recordings. This algorithm forms the basis of the source removal system described in this paper.

2.1 Unsupervised Learning Techniques for Source Separation

The source separation problem can be described in general terms as follows (O’Grady et al. 2005, pp. 18 – 20). A given set of \(N\) observations, denoted by \(x_1, x_2, \ldots, x_N\), consists of a linear combination of \(K\) latent variables:

\[
x_1 = a_{11}s_1 + a_{12}s_2 + \ldots + a_{1K}s_K \\
x_2 = a_{21}s_1 + a_{22}s_2 + \ldots + a_{2K}s_K \\
\ldots \\
x_N = a_{N1}s_1 + a_{N2}s_2 + \ldots + a_{NK}s_K
\]
where $s_1, s_2, \ldots, s_K$ are the latent variables, and $a_q$ are the mixture components. This can be re-written in vector-matrix form as:

$$x = As$$

where $x$ is the observation vector, $s$ is the latent variables vector, and $A$ is the mixing matrix. The task of a source separation algorithm is to estimate both the mixing matrix $A$ and the source vector $s$ (also referred to as basis functions). The unsupervised machine learning techniques described below make use of this linear signal model, which can be justified by the fact that real-world audio signals add linearly (Virtanen 2006, p. 268).

The observation vector $x$ can be represented as either time-domain or frequency-domain data. While computationally simpler than frequency-domain representations, time-domain representations are complicated by the fact that one must take into account the phase of the signal when formulating basis functions (Virtanen 2006, pp. 271 – 272). In addition, since they can take both positive and negative values, time-domain representations may not be appropriate in cases where the algorithm employs non-negativity constraints (Virtanen 2006, p. 271).

Frequency-domain signal representations, such as the Discrete Time Fourier Transform (DTFT) and its variants, are often employed (Virtanen 2006, p. 273). In many cases, the phase information is discarded, and the magnitude or power spectrum is used to represent the signal (Virtanen 2006, p. 273). Although some information is lost
in this approach (i.e., the phase information), it turns out that “many real-world sounds can be rather well approximated with a fixed magnitude spectrum and a time varying gain” (Virtanen 2006, p. 273). The magnitude and power spectrum representations have a number of disadvantages, however. For example, unlike time-domain representations neither magnitude nor power spectra add linearly. However, as long as the phases of the signals “are uniformly distributed and independent of each other… we can approximate time-domain summation in the power spectral domain” (Virtanen 2006, p. 273). It is interesting to note that although their use is widespread and usually yields good results, this approximation does not hold for magnitude spectra (Virtanen 2006, p. 273). In addition, the power spectrum representation is not sensitive to low-energy components; however, various researchers have developed techniques to alleviate this problem (Virtanen 2006, pp. 273 – 274).

2.1.1 Independent Component Analysis

Independent component analysis (ICA) is a data reduction technique that attempts “to estimate the latent source from a set of observations” (Gong & Xu 2007, p. 20). ICA used the signal model described in section 2.1, i.e., that the observation vector consists of a linear combination of latent sources \( \mathbf{x} = \mathbf{A}\mathbf{s} \). In order to ensure a unique solution, an additional constraint is added, namely, that “each latent variable \( s_i \) [has] the unit variance: \( E[s_i^2] = 1, \forall i \)” (Gong & Xu 2007, p. 21). An ambiguity that is not resolved is the fact that the latent variables can be multiplied by \(-1\) “without affecting the model” (Gong & Xu 2007, p. 21). However, “this ambiguity is not a serious problem in many applications” (Gong & Xu 2007, p. 21).
The ICA model also assumes “that the latent variables $s_i$ are statistically independent and must have non-Gaussian distribution” (Gong & Xu 2007, p. 21). It is also important to note that the “standard ICA model assumes that the mixing matrix $A$ is square” (Gong & Xu 2007, p. 21) and therefore the number of observations equals the number of latent sources. Given these assumptions, the ICA problem reduces to that of finding “a matrix $A$ such that the latent variables obtained by $s = A^{-1}x$ are as independent and non-Gaussian as possible” (Gong & Xu 2007, p. 21).

The metrics kurtosis, negentropy, and mutual information have all been used as measures of non-Gaussianity and component independence in ICA algorithms (Gong & Xu 2007, p. 21–22). Kurtosis measures the non-Gaussianity of a variable, and “is zero for Gaussian variables and non-zero for most (but not all) non-Gaussian variables” (Gong & Xu 2007, p. 21). Because of its sensitivity to outlying data, however, it is not well suited for use in ICA (Gong & Xu 2007, p. 22). Negentropy, like kurtosis, is a measure of non-Gaussianity that is zero if and only the variable is Gaussian, and is otherwise positive (Gong & Xu 2007, p. 22). Calculation of the precise value of the negentropy of a variable is difficult, as it requires knowledge of the probability density function of that variable, which is not known except in the simplest cases (Gong & Xu 2007, p. 22). As an alternative, an approximation of negentropy has been proposed (Gong & Xu 2007, p. 22). Mutual information can be used to measure the independence of the components. It takes non-negative values unless the “variables are statistically independent” (Gong & Xu 2007, p. 23).

Before attempting to estimate the mixing matrix $A$, a number of pre-processing steps may be required. In certain cases, it is necessary to use Principal Component
Analysis (PCA) to decorrelate the variables (Virtanen 2006, p. 275). In general, the observation vector \( \mathbf{x} \) must be centered. A centered vector has a mean of zero; thus the mean of \( \mathbf{x} \) must be subtracted from \( \mathbf{x} \). In addition, the vector \( \mathbf{x} \) must be whitened, a process which results in the transformation of the mixing matrix \( \mathbf{A} \) into an orthogonal matrix (Gong & Xu 2007, p. 24). The whitening of \( \mathbf{x} \) relies on the earlier assumptions “that the latent variables \( \mathbf{s} \) are independent, have zero mean and unit variance” (Gong & Xu 2007, p. 24). The whitening step reduces the degrees of freedom of the mixing matrix \( \mathbf{A} \), thereby significantly reducing the computational complexity of the problem (Gong & Xu 2007, p. 24).

As stated above, the objective of any ICA algorithm is to produce a set of latent variables \( \mathbf{s} \) that “look as far from Gaussian and as independent as possible” (Gong & Xu 2007, p. 25). The algorithm attempts to estimate inverse of the mixing matrix, \( \mathbf{A}^{-1} \) (also referred to as the unmixing matrix) (Virtanen 2006, p. 275). An estimate of the source vector \( \hat{\mathbf{s}} \) is calculated using \( \hat{\mathbf{s}} = \mathbf{Wx} \), where \( \mathbf{W} \) is the whitened version of the unmixing matrix and is estimated so that the elements of \( \hat{\mathbf{s}} \) “become maximally independent” (Virtanen 2006, p. 275). The problem is thus similar to a maximization problem, which can be solved using an algorithm such as gradient descent (Gong & Xu 2007, p. 25). Gradient descent is an iterative algorithm that should produce a local (if not a global) maximum given a sufficient number of iterations (Gong & Xu 2007, pp. 208 – 209).

Because the standard ICA algorithm assumes that the observation vector \( \mathbf{x} \) and the source vector \( \mathbf{s} \) are the same size, the standard ICA method, when applied to the sound source separation problem, requires that the number of sensors must equal the number of sources in the recording to be analyzed (Wang & Brown 2006, p. 29). If the
observation vector is represented by a frequency-domain representation such as the STFT, the number of frequency bins is likely greater than the size of the source vector and can thus be treated as independent observations (Virtanen 2006, p. 277). In this case, a dimension reduction technique such as Principal Component Analysis (PCA) or Singular Value Decomposition (SVD) must be used to transform the observation vector into a fixed spectrum and a set of time-varying gains with certain orthogonality properties (Virtanen 2006, p. 277).

2.1.2 Non-negative Matrix Factorization


The NMF algorithm decomposes an observation signal into a set of basis functions and a set of weights. As the name implies, the values of the observation signal, basis functions, and weights are all non-negative. The observation data is assumed to consist of “\(T\) measurements of \(N\) non-negative scalar variables” (Hoyer 2004, p. 1458), and is denoted by the \(N \times T\) matrix \(X\). The weighting coefficients are denoted by the \(N \times K\) matrix \(A\), and the basis functions are denoted by the \(K \times T\) matrix \(S\). NMF thus performs the following factorization:

\[ X \approx AS \]
There are a number of different cost functions one can use to assess the quality of the approximation. The NMF algorithm adjusts the values of $A$ and $S$ according to the cost function until the approximation falls within an accepted limit. Hoyer’s algorithm also includes sparseness constraints on $A$ and $S$ (Hoyer 2002; Hoyer 2004).

When applying the NMF approach to audio signals, the time-domain signal is not usable as it contains both positive and negative values. Therefore, an alternate representation of the signal must be used. Typically, the magnitude of the spectrogram is employed. In this case, the observation matrix $X$ consists of $N$ frequency bins taken at $T$ time frames, for instance as computed by the short-time Fourier transform (STFT). The columns of $A$ represent the frequency content of each signal component, and the rows of $S$ represent the timing information of each component (Smaragdis 2004, p. 2). In order to perform proper source separation, these components must be grouped properly into the different instruments present in the original signal (Smaragdis 2004, p. 2).
2.2 ADRess

The source separation approach employed in this thesis is known as Azimuth Discrimination and Resynthesis, or ADRess (Barry et al. 2004). The ADRess algorithm takes advantage of the differing intensities between the left and right channels of a stereo recording to create what the authors refer to as an azimugram. The azimugram is a representation of the individual sources of the input signal as they distributed in the stereo image. Typically, when a multi-track recording is mixed down to stereo, each instrument is placed at a certain location in the stereo image by scaling the intensity of the instrument track in the left and right channels. The ADRess algorithm makes use of the different intensity levels between the left and right channels to separate instruments from the stereo mix. This method assumes that “the phase of a source is coherent between left and right [channels], and only its intensity differs” (Barry et al. 2004, p. 1) and that each instrument is located at a specific (and constant) position in the stereo image.

This algorithm has a number of limitations. For example, it has varying success with recordings made with methods that do not adhere to these assumptions, such as those made using binaural, mid-side, and stereo pair recording techniques, though the algorithm does perform well “on the majority of recordings” (Barry et al. 2004, p. 1). In addition, the algorithm cannot separate two sources that are panned to the same location in the stereo image. Further, the algorithm cannot properly separate sources whose position in the stereo image varies over time.

The ADRess algorithm is based on the same signal model assumed by ICA and NMF, namely that the observed signals are a linear combination of individual source
signals (Barry et al. 2004, p. 2). In this case, the two observation vectors $L(t)$ and $R(t)$, the left and the right channels respectively, can be characterized as follows:

$$L(t) = \sum_{j=1}^{J} P_l \cdot S_j(t)$$  \hspace{1cm} (2.2.1a)

$$R(t) = \sum_{j=1}^{J} P_r \cdot S_j(t)$$  \hspace{1cm} (2.2.1b)

where $P_l$ and $P_r$ are the left and right panning coefficients for $S_j(t)$, i.e. the $j^{th}$ source.

The intensity ratio between the two channels is defined by the following equation:

$$g(j) = \frac{P_l}{P_r}$$  \hspace{1cm} (2.2.2)

The equation above can be rewritten $P_l = g(j) \cdot P_r$. Because $L(t)$ and $R(t)$ are linear combinations of sources, the $j^{th}$ source is cancelled out in the following expression:

$$L(t) - g(j) \cdot R(t)$$

This expression is valid if the $j^{th}$ source is predominant in the right channel, meaning that $P_r \geq P_l$, and thus $0 \leq g(j) \leq 1$. To cancel out a source that is predominant on the left channel, we would use the expression $R(t) - g(j) \cdot L(t)$ where $g(j) = P_r / P_l$.

The same principle can be applied in the frequency domain by applying an $N$ point STFT to the above equation. In this manner, we can create what the authors refer to as a “frequency-azimuth plane” (Barry et al. 2004, p. 2) or azimugram. As its
name implies, the frequency-azimuth plane plots the magnitude of a signal as a function of frequency and azimuth. Azimuth here “is purely a function of the intensity ratio [between left and right channels], created by the pan pot during mix down” (Barry et al. 2004, p. 2), and thus indicates a horizontal position in the stereo image. Figure 1 shows the right portion of the azimugram for a single complex waveform.

The gain scale $g(i)$ is defined as:

$$g(i) = i \times \frac{1}{\beta}$$  \hspace{1cm} (2.2.3)

where $0 \leq i \leq \beta$ and both $i$ and $\beta$ are integers. It follows that $0 \leq g(i) \leq 1$. $\beta$ is the azimuth resolution, a user selected parameter that “refers to how many equally spaced gain scaling values of $g$ we will use to construct our frequency-azimuth plane” (Barry et. al. 2004, p. 2). In figure 1, the azimuth resolution $\beta = 100$.  


Figure 1: right azimugram for single complex waveform

We can thus define $Az_R(k,i)$ and $Az_L(k,i)$, the right and left azimugrams, respectively, using the following equations:

$$AZ_R(k,i) = |Lf(k) - g(i) \cdot Rf(k)|$$  \hspace{1cm} (2.2.4a)

$$AZ_L(k,i) = |Rf(k) - g(i) \cdot Lf(k)|$$  \hspace{1cm} (2.2.4b)
where \( 1 \leq k \leq N \) and \( 0 \leq i \leq \beta \). Note that \( Lf(k) \) and \( Rf(k) \) are complex spectrograms. Each azimugram is an \( N \times \beta \) matrix. If each frequency bin \( k \) belongs to a source that has been panned to a certain location in the stereo spectrum, then the situation is much like the time-domain case described above. In other words, for a given frequency bin \( k \), \( Az_s(k, i) = 0 \) for a particular value of \( g(i) \). This value of \( g(i) \) corresponds to the intensity ratio of the left and right panning coefficients, in this case, \( g(i) = Pl_i/Pr_i \) (note that the value of \( g(i) \) is related to the value of \( i \) as given by the equation (2.2.3) above). All the frequency bins of that source should cancel out at that value of \( g(i) \). For the purposes of source separation, we need to reverse the above process. That is, given a position along the azimuth axis \( i \), we want to be able to locate all the frequency bins belonging to that source. As demonstrated above, if a source in frequency bin \( k \) is panned using a particular intensity ratio \( g(i) \) (which indicates a particular value of \( i \)), \( Az_s(k, i) = 0 \) for that frequency bin and azimuth index. Therefore, for a given value of \( i \), all the values of \( k \) at which the azimugram cancels out (i.e., where \( Az_s(k, i) = 0 \)) belong to a particular source.

However, because there is usually some overlap of the harmonics of sources in a mixture, the azimugram will not perfectly cancel out, since one side may have energy belonging to a different source. Therefore, for a given frequency bin \( k \), we must look for the value of \( i \) that minimizes the expression in equation (2.2.4a). The magnitude of the peak at the point of minimization is estimated to be \( Az_s(k)_{\text{max}} - Az_s(k)_{\text{min}} \). Thus, we modify the azimugrams computed using equations (2.2.4a) and (2.2.4b) using the following:
\[ A_{z_a}(k,i) = \begin{cases} A_{z_a}(k,i)_{\text{max}} - A_{z_a}(k,i)_{\text{min}} & \text{if } A_{z_a}(k,i) = A_{z_a}(k)_{\text{min}} \\ 0 & \text{otherwise} \end{cases} \] (2.2.5a)

\[ A_{z_l}(k,i) = \begin{cases} A_{z_l}(k,i)_{\text{max}} - A_{z_l}(k,i)_{\text{min}} & \text{if } A_{z_l}(k,i) = A_{z_l}(k)_{\text{min}} \\ 0 & \text{otherwise} \end{cases} \] (2.2.5b)

This process turns the “nulls” (i.e., minima) produced in equations (2.2.4a) and (2.2.4b) into peaks, and sets all other points in the frequency-azimuth plane to zero magnitude. The azimugrams now resemble the one shown in figure 1: at a given azimuth index \( i \), the azimugram has peaks at the frequency bins corresponding to the source located at that azimuth position; the other frequency bins are set to 0. Note that this azimugram represents only one time frame; since the observation signals are transformed via the STFT, it is necessary to compute a left and right azimugram for each STFT frame.

After the azimugrams of the input signal have been computed, the discrimination index \( d \) is identified. The discrimination index is the point on the azimuth axis representing “the apparent position of the source [of interest]” (Barry et al. 2004, p. 3), and takes the values \( 0 \leq d \leq \beta \). This may be done in a variety of ways, most simply via user input (for example, through trial and error). The azimugram at discrimination index \( d \), \( A_{z_a}(k,i = d) \), is essentially a magnitude spectrum. In the ideal case, this spectrum would perfectly represent the magnitude spectrum of the source of interest, making constructing the magnitude spectrogram of the source of interest an easy task.

However, real-world signals exhibit what the authors term “frequency-azimuth smearing” (Barry et al. 2004, p. 3). Because the harmonics of various sources present in
real-world musical signals overlap greatly, “two or more sources contain energy in a frequency bin” (Barry et al. 2004, p. 3), causing the peaks to drift away from the expected azimuth index. The authors thus introduce the “azimuth subspace width,” (Barry et al. 2004, p. 3), denoted by $H$, where $1 \leq H \leq \beta$. The azimuth subspace width “allows us to recover peaks within a given neighborhood” (Barry et al. 2004, p. 3) by including peaks in a window of azimuth indices around the discrimination index. Thus, during the resynthesis process, the magnitude spectra for each analysis frame are computed using the following formulae:

\[
Y_r(k) = \sum_{i=d-H/2}^{d+H/2} A_{r_i}(k) 1 \leq k \leq N \tag{2.2.6a}
\]

\[
Y_l(k) = \sum_{i=d-H/2}^{d+H/2} A_{l_i}(k) 1 \leq k \leq N \tag{2.2.6b}
\]

Equation (2.2.6a) is used to resynthesize the separated signal if it is predominant in the right channel in the original stereo mix, and equation (2.2.6b) is used if the separated signal is predominant in the left channel. The authors do not formally define what it means for a source to be predominant in one channel. Informally, however, one can say that a source is predominant in a particular channel if a listener perceives the source to be louder in that channel. The algorithm does not require a more formal definition.

In order to compute a time-domain representation of the separated signal, the phase of the signal as well as its magnitude spectrogram is required. A number of different phase estimation algorithms can be used for this step, including “magnitude only” reconstruction, an iterative method of phase estimation that employs the least
squared error metric (Barry et al. 2005a, pp. 3 – 4). However, the authors assert that “using the original bin phases is adequate” (Barry et al. 2004, p. 3). These values are given by:

\[ R(k) = |Rf(k)| \]
\[ L(k) = |Lf(k)| \]

If the signal is predominant in the right channel, equation (2.2.7a) should be used to estimate the phase, while if it is predominant in the left channel, equation (2.2.7b) should be used. Using the magnitude spectrogram computed from the azimugram and the above phase information, the complex spectrogram \( X(k) \) is computed using the equations below:

\[ X(k) = Y_R(k) \cdot e^{-i \Phi_R(k)} \]  
\[ X(k) = Y_L(k) \cdot e^{-i \Phi_L(k)} \]

As above, equation (2.2.8a) is used if the signal of interest is predominant in the right channel, and equation (2.2.8b) is used if it is predominant in the left channel. The time-domain representation of the separated signal can be computed from \( X(k) \) using the inverse fast Fourier transform (IFFT).
3. System Implementation

The general architecture of the source removal system is shown in figure 2. The basic steps in the computation of the original mix signal minus the separated signal (MMS) are as follows (each step will be discussed in detail in the following sections):

1. The (stereo) input signal is translated into a time-frequency representation (i.e., a complex spectrogram) via the short-time Fourier transform (STFT).
2. The left and right azimugrams are computed, as described in the previous section.
3. The magnitude spectrogram of the separated signal is calculated, as is its complement (the proper values for the discrimination index and azimuth subspace width are determined beforehand via a process of trial and error). The latter provides an estimate of the magnitude spectrogram of the MMS signal.
4. The magnitude spectrogram of the separated signal is then converted into a binary time-frequency (T-F) mask using thresholding.
5. The binary T-F mask is then filtered using a de-noising filter to produce a second binary T-F mask.
6. The original binary T-F mask is analyzed using Sinusoidal Modeling Synthesis (SMS) to produce a third binary T-F mask.
7. These three T-F masks are then applied to the magnitude spectrogram of the original signal to produce three different estimates of the magnitude spectrogram of the MMS signal.
8. A phase estimate of the MMS signal, with various corrections, is computed.
9. The complex spectrogram of the MMS signal estimate is produced by combining one of the four magnitude spectrogram estimates with the phase estimate. The IFFT in conjunction with the overlap/add method of signal reconstruction transforms the complex spectrogram into a time domain signal.

Note that the source removal system also produces a time-domain version of the separated signal. This feature was omitted from figure 2 for reasons of clarity. Also note that the system described in this section was implemented in Matlab, using Dan Ellis’ SMS Matlab toolbox (Ellis 2003), Pascal Getreuer’s Matlab implementation of a total variance de-noising filter (Getreuer 2007), as well as analysis/synthesis Matlab code written by De Götzen (De Götzen et al. 2000).
Figure 2: block diagram of source removal system
3.1 STFT

The short-time Fourier transform (STFT) is a technique that transforms a real-world, “sampled, finite-duration, time-varying [signal]” (Roads 1996, p. 550) into a frequency-domain representation. The STFT can be viewed as a windowed discrete Fourier transform (DFT); that is, it computes the DFT of a series of overlapping segments of the input signal (Arfib et al. 2002, pp. 237 – 239). A segment of size \( N \) samples is taken from the input signal \( x[n] \), and is multiplied by a window function \( h[n] \); the DFT is then computed for this windowed segment (Arfib et al. 2002, p. 238 – 239). In other words, the STFT is a series of frequency spectra, each representing a different segment of the input signal. The STFT \( X[n,k] \) can be represented mathematically as follows (Dolson 1986, p. 25):

\[
X[n,k] = \sum_{m=-\infty}^{\infty} x[m] \cdot h[n - m] \cdot e^{-j \frac{2\pi km}{N}}
\]

Equation (13) can also be rewritten as follows (Roads 1996, p. 1095):

\[
X[l,k] = \sum_{m=0}^{M-1} h[m] \cdot x[m + l \cdot H] \cdot e^{-j \frac{2\pi km}{N}}
\]

where \( l \) is the frame index and \( H \) is the hop size in samples. The hop size is simply the number of samples the STFT advances in the original signal before taking another analysis window, while the frame index is a pointer to the individual spectra.
A number of important details influence the analysis quality of the STFT. The window size $N$ affects the resolution of the calculated spectrum, as well as the time resolution of the analysis. In short, there is a trade-off between resolution in the frequency domain and in the time domain (Roads 1996, p. 1094). In addition, the type of window used affects the analysis quality – windows can add “unwanted noise and discontinuities in [the resulting] spectrum” (De Götzen et al. 2000, p. 2). Some of the common choices are the Hamming, Hanning, and Blackman windows (De Götzen et al. 2000, p. 3). Because the signal is eventually going to be resynthesized via the IFFT (as discussed below), the properties that result with overlapping and adding the windows are important to consider (De Götzen 2000, p. 3). De Götzen et al. suggest the use of a modified Hanning window, called hanningz, that begins “by a zero-valued sample and [ends] by a non-zero valued sample” (De Götzen et al. 2000, p. 4). This is the window employed in this implementation of the source removal system. The authors also suggest centering the FFT in order to prevent the phase from unwrapping “in opposite directions for odd and even bins” (De Götzen 2000, p. 4). This centering is easily implemented in Matlab using the \texttt{fftshift} function.

### 3.2 Azimuth computation

After the STFT of the input signal has been computed, the left and right azimugrams are computed, as described by equations (2.2.4a), (2.2.4b), (2.2.5.a), and (2.2.5b) in section 2. The discrimination index $d$ and azimuth subspace width $H$ are provided to the system as user input, these values being determined through a process of
trial and error. The user also specifies whether the source of interest is predominant in the left or right channels of the original signal.

### 3.3 Computation of Separated Signal and Azimugram Complement

The magnitude spectrum of the separated signal is computed, for each analysis frame, using equation (2.2.6a) if the signal is predominant in the right channel, or equation (2.2.6b) if the signal is predominant in the left channel. The magnitude spectrum of the separated signal, \(|Y_{sep}(k)|\), is monophonic. However, as will be seen below, the masking process requires a stereo version of this signal. Therefore, it is necessary to compute the left and right magnitude spectrograms of the separated signal, \(Y_{sepL}(k)\) and \(Y_{sepR}(k)\) respectively. If the source of interest is in the right channel, then equation (2.2.2) implies that \(Pl_j = g(j) \cdot Pr_j\) for the \(j^{th}\) source. The magnitude spectrum for this source computed using equation (2.2.6a), \(Y_{sR}(k)\), equals \(Pr_j \cdot |Y_{sep}(k,j)|\), or the magnitude spectrum for the separated signal panned in the right channel. Therefore, the magnitude spectrum for the separated signal in the left channel equals \(g(j) \cdot Y_{sL}(k)\). A similar relation holds if the source of interest is predominant in the left channel.

The azimugram complement describes the complement of the operation that computes \(Y_{sR}(k)\) and \(Y_{sL}(k)\), as shown in equations (2.2.6a) and (2.2.6b). These equations are used to compute the magnitude spectrum of the source of interest, which is bounded by the azimuth axis values \(d - H/2\) and \(d + H/2\). Since the azimugram in this region contains the magnitude spectrum of the separated signal, the azimugram in the complement of this region must contain the magnitude spectrum of the original signal.
minus the separated signal (i.e., the MMS signal). Therefore, the right and left magnitude spectra of the MMS signal can be calculated using the following formulae if the separated source is predominant in the right channel:

\[
Y_R(k) = \sum_{i=0}^{(d-H/2)-1} A_{z_R}(k,i) + \sum_{i=(d-H/2)+1}^{\beta} A_{z_R}(k,i) \\
Y_L(k) = \sum_{i=0}^{\beta} A_{z_L}(k,i)
\]  

(3.3.1)

and the following formulae if the separated source is predominant in the left channel:

\[
Y_R(k) = \sum_{i=0}^{\beta} A_{z_R}(k,i) \\
Y_L(k) = \sum_{i=0}^{(d-H/2)-1} A_{z_L}(k,i) + \sum_{i=(d-H/2)+1}^{\beta} A_{z_L}(k,i)
\]  

(3.3.2)

The appropriate set of equations is used to compute the MMS signal.

3.4 Thresholding/Computation of T-F Mask

In this stage, the stereo version of the separated signal is converted to a binary time-frequency (T-F) mask. The concept of a T-F mask “is motivated by the phenomenon of masking in auditory perception, in which a sound is rendered inaudible by a louder sound within a critical band” (Wang & Brown 2006, p. 22). As the name implies, a binary time-frequency mask has a value of either 0 or 1 at every point time-
frequency space: the mask takes the value 0 at time-frequency units one wishes to suppress, and the value 1 elsewhere (Wang & Brown 2006, p. 22). Since the purpose of this step is to remove the source signal from the original signal, it is clear that the mask must suppress in the original signal those time-frequency units that correspond to perceptually significant time-frequency units in the separated signal. Therefore, a binary T-F mask can be computed from the magnitude spectrogram of the separated signal computed in the previous step. Since it has been observed that a “target sound reconstructed from an ideal mask is of high perceptual quality” (Wang & Brown 2006, p. 23), this approach is justified. Thus, in this step, left and right binary T-F masks are computed from the left and right magnitude spectrograms of the separated signal using the following equations:

$$mask_R (n, k) = \begin{cases} 0 \text{ if } |Ysep_R (n, k)| > \alpha \\ 1 \text{ otherwise} \end{cases} \quad (3.4.1a)$$

$$mask_L (n, k) = \begin{cases} 0 \text{ if } |Ysep_L (n, k)| > \alpha \\ 1 \text{ otherwise} \end{cases} \quad (3.4.1b)$$

The value of the threshold $\alpha$ is set to a fixed ratio of the maximum amplitude in $Ysep_L$ and $Ysep_R$ (over all time units $n$ and frequency bins $k$).

3.5 Filtering the Binary T-F Mask

Even with the thresholding, the binary T-F masks created in the previous step contain a good deal of noise. This noise can be seen in figure 3, which shows the detail
of a plot of a binary T-F mask created, using the procedure described above, from the magnitude spectrogram of a source separated from a music signal. The noise in this mask might produce audible artifacts in the resynthesized MMS signal. To appearances, these masks are noisy images; thus, it should be possible to remove the artifacts by applying a de-noising filter designed for use in image processing to the binary T-F mask computed in the previous step.

The de-noising filter used to performing this filtering is based on the total variation (TV) norm (Rudin et al. 1992, p. 259). The algorithm performs the de-noising “by minimizing the total variation norm of the estimated solution” (Rudin et al. 1992, p. 259). Thus, the algorithm is a constrained minimization problem, which can be solved using an iterative method such as gradient descent (Rudin et al. 1992, pp. 259 – 261). Since the algorithm is iterative, the solution converges as the number of iterations approaches infinity (Rudin et al. 1992, p. 261). Although the algorithm computes a local minimum, the local minima are sufficient for the task of image de-noising (Rudin et al. 1992, p. 261). The authors provide a computationally efficient numerical method to compute an approximate solution (Rudin et al. 1992, p. 262), and an implementation

![Figure 3: Detail of T-F mask](image)

1992, p. 261). The authors provide a computationally efficient numerical method to compute an approximate solution (Rudin et al. 1992, p. 262), and an implementation
yields good results (Rudin et al. 1992, pp. 262 – 268). In this step, a Matlab implementation of this algorithm, tvdenoise (Getreuer 2007), is used to perform the filtering.

3.6 SMS Modeling

Another technique that can be employed to reduce the noise in the binary T-F masks is Sinusoidal Modeling Synthesis (SMS), an analysis and synthesis technique whose central assumption is that signals can be modeled with a deterministic and a stochastic component. In other words, “time-varying spectra [can be modeled] as (1) a collection of sinusoids controlled through time by piecewise linear amplitude and frequency envelopes (the deterministic part), and (2) a time-varying filtered noise component (the stochastic part)” (Serra & Smith 1990, p. 12). The stochastic component is sometimes referred to as the residual (Roads 1996, p. 152).

The signal model used in SMS is as follows:

\[
s(t) = \sum_{r=1}^{R} A_r(t) \cdot \cos[\theta_r(t)] + e(t) \tag{3.6.1}
\]

where \(s(t)\) is the input signal, \(A_r(t)\) is the instantaneous amplitude of the \(r^{th}\) sinusoid, \(\theta_r(t)\) is the instantaneous phase of the \(r^{th}\) sinusoid, \(e(t)\) is the noise component, and time is \(t\) (Serra & Smith 1990, p. 14). The SMS model assumes that “the sinusoids are stable partials of the sound and that each one has a slowly changing amplitude and frequency” (Serra & Smith 1990, p. 14). The sinusoids thus model the main modes of vibration of a
physical system (Serra 1997, p. 3). By contrast, the residual component consists of “the energy produced by the excitation mechanism [of a physical system] which is not transformed by the system into stationary vibrations plus any energy component that is not sinusoidal in nature” (Serra 1997, p. 3), such as attack transients. The goal of this step in the source removal process is to eliminate the noise in the binary T-F mask. Therefore, the modeling of the stochastic component is not relevant to this purpose.

Creating the sinusoidal model from a magnitude spectrogram consists of two steps: peak detection and peak continuation. An optional pitch detection stage may occur between these two steps, if the input sound is known to be “monophonic and pseudo-harmonic” (Amatriain 2002, p. 387). This assumption cannot be made for the purposes of the source removal system, as the source being removed is not necessarily monophonic.

The peak detection step detects the “prominent” peaks in the magnitude spectrum (Serra 1997, p. 8), where “a ‘peak’ is defined as a local maximum in the magnitude spectrum” (Serra 1997, p. 8), perhaps with additional constraints such as frequency range or magnitude threshold. The peak detection stage is liberal in its definition of a peak; the peak continuation step determines whether a particular peak represents a stable partial or transient noise (Serra 1997, p. 8).

The peak continuation algorithm attempts to place the peaks detected in the previous step into peak trajectories, which represent stable sinusoids (Serra & Smith 1990, p. 17). This algorithm also makes use of frequency guides, which advance “in time through the spectral peaks, looking for the appropriate ones (according to the specified
constraints), and forming trajectories out of them” (Serra & Smith 1990, p. 17). Each
guide maintains a running frequency estimate (Serra & Smith 1990, p. 17).

In a given time frame, existing frequency guides are first advanced through the
current frame “by finding the peak closest in frequency to its current value” (Serra &
Smith 1990, p. 18). The guide is continued through the frame if it finds a matching peak
within a specified frequency and amplitude deviation (Amatriain 2002, p. 393). If the
guide is unable to find a match, it (and its corresponding trajectory) is turned off (or set to
a “sleeping” state), and may be “killed” if the “guide has not found a continuation peak
for a given amount of time” (Amatriain 2002, p. 393). If two guides match the same
peak, the closest guide is deemed the “winner,” and the other guide looks for a match
among the remaining peaks (Serra & Smith 1990, p. 18). Next, the frequency estimates
of all the guides are updated, and any guide that has been sleeping for longer than the
allowed time is killed (Serra & Smith 1990, p. 18). Finally, “new guides, and therefore
new trajectories, are created from the peaks of the current frame that are not incorporated
into trajectories by the existing guides” (Amatriain 2002, p. 393). The output of this
process is a series of trajectories, each with a frequency estimate and a magnitude
estimate at each frame.

In this step of the source removal system, a Matlab SMS implementation (Ellis
2003) is used to model the sinusoidal components of the binary T-F mask created from
the magnitude spectrogram of the separated signal. The frequency and amplitude
estimates are used to create a new binary T-F mask. In addition, each trajectory is
widened in frequency range to account for the inaccuracies in converting the SMS
frequency estimates into STFT frequency bin numbers. In the ideal, the T-F mask
computed by this step will have none of the noisy components of the original T-F mask, and thus, when applied to the original spectrogram, produce a magnitude spectrogram with fewer audible artifacts.

3.7 Masking

Once the three binary T-F masks have been computed, they are applied to the magnitude spectrogram of the original signal as described by the following equations:

\[ Y_r(n,k) = R_f(n,k) \cdot \text{mask}_r(n,k) \]  
\[ Y_l(n,k) = L_f(n,k) \cdot \text{mask}_l(n,k) \]

where \( Y_r(n,k) \) and \( Y_l(n,k) \) are the right and left magnitude spectrograms of the MMS signal, and \( R_f(n,k) \) and \( L_f(n,k) \) are the right and left magnitude spectrograms of the input signal.

3.8 Estimation and Correction of MMS Phase

Following the suggestion of Barry (Barry et al. 2004, p. 3), the phases of the input signal, given by equations (2.2.7a) and (2.2.7b), are taken as the initial estimates for the phases of the MMS signal. In order to remove any contributions to the MMS phases from the separated signal, the MMS phases are set to 0 wherever the separated signal has non-zero magnitude. More precisely,
\[
\Phi_\sigma(n, k) = \begin{cases} 
0 \text{ where } |\text{Ysep}_\sigma(n, k)| > 0 \\
\Phi_\sigma(n, k) \text{ otherwise}
\end{cases} \quad (3.8.1a)
\]

\[
\Phi_\sigma(n, k) = \begin{cases} 
0 \text{ where } |\text{Ysep}_\sigma(n, k)| > 0 \\
\Phi_\sigma(n, k) \text{ otherwise}
\end{cases} \quad (3.8.1b)
\]

The phases \( \Phi_\sigma(n, k) \) and \( \Phi_\iota(n, k) \) are then smoothed using the algorithm described by De Götzen (De Götzen 2000, pp. 4 – 5).

### 3.9 IFFT/Resynthesis

The complex right and left spectrograms of the MMS signal are constructed using the phases computed above and one of the magnitude spectrograms computed using the azimugram complement, binary T-F mask, filtered binary T-F mask, or SMS modeled binary T-F mask. Specifically, the complex spectrograms are computed using the following:

\[
Y_{\text{mms}_\sigma}(n, k) = Y_\sigma(n, k) \cdot e^{-i\Phi_\sigma(n, k)} \quad (3.9.1a)
\]

\[
Y_{\text{mms}_\iota}(n, k) = Y_\iota(n, k) \cdot e^{-i\Phi_\iota(n, k)} \quad (3.9.1b)
\]

where \( Y_{\text{mms}_\sigma}(n, k) \) and \( Y_{\text{mms}_\iota}(n, k) \) are the right and left complex spectrograms of the MMS signal.

The inverse fast Fourier transform (IFFT), in conjunction with the overlap-add technique, is employed to transform these complex spectrograms into a (stereo) time-domain signal. The general form of the IFFT is given by the equation below:
\[ x[n] = \frac{1}{N} \cdot \sum_{k=0}^{N-1} X[k] \cdot e^{\frac{2\pi j mk}{N}} \]  \hspace{1cm} (3.9.2)

where \( X[k] \) is a spectrogram and \( x[n] \) is the time-domain signal. The inverse of the STFT can be expressed using the following equation (Dolson 1986, p. 25):

\[ x[n] = \sum_{m=-\infty}^{\infty} h[n - m] \cdot \frac{1}{N} \sum_{k=0}^{N-1} X[m,k] \cdot e^{\frac{2\pi j mk}{N}} \]  \hspace{1cm} (3.9.3)

where \( X[m,k] \) is the spectrogram of the time-domain signal \( x[n] \) and \( h[n] \) is a window function. According to equation (3.9.2), the output signal \( x[n] \) can be computed with what is known as the overlap-add procedure, which can be summarized as follows (Roads 1996, p. 1105; De Götzen et al. 2000, p.5). Initialize an output buffer \( x \) to zero and initialize an output pointer \( m \) to 0. For each frame of the complex spectrogram \( l \), compute the IFFT using equation (3.9.2), yielding the time-domain signal \( y_l[n] \). Multiply \( y_l[n] \) by a resynthesis window \( h[n] \) and add this product to the output buffer at the position defined by the output pointer \( m \). Advance \( m \) by a hop size and continue the procedure for all spectrogram frames. The overlap-add procedure is used to compute the right and left channels of the output signal with the right and left complex spectrograms, respectively, computed in equations (3.9.1a) and (3.9.1b). The synthesis window type, window size, and hop size used here are the same as the analysis window type, window size, and hop size used in the STFT step (described in section 3.1).
4. Results

4.1 Experimental Design

In order to test the accuracy of the source removal system implementation described in the previous section, it is necessary to perform quantitative as well as qualitative measurements. The former requires both a database of test files and a proper measurement metric. The signal to noise ratio (SNR), also referred to as the signal to distortion ratio, is one such metric (Virtanen 2006; Wang & Plumbley 2005). The SNR (in decibels) can be defined as follows:

\[
\text{SNR}[\text{dB}] = 10 \cdot \log_{10} \left\{ \frac{\sum_{t} s(t)^2}{\sum_{t} [\hat{s}(t) - s(t)]^2} \right\}
\]  

(4.1.1)

where “\( s(t) \) is a reference signal of the source before mixing, and \( \hat{s}(t) \) is the separated signal [as computed by a source separation algorithm]” (Virtanen 2006, p. 294). Equation (4.1.1) can be used to compute both the SNR for the separated signal and for the MMS signal (i.e., the original signal minus the separated source signal). The closer separated signal is to the reference signal, the closer the denominator of equation (4.1.1) approaches 0, and thus the larger the value of SNR. In other words, the higher the SNR, the closer the separated signal is to the reference signal.

It is clear that the test database requires, for each “song”, a full mix of all instruments, a track consisting solely of the instrument to be separated, and a mix of all instruments with the exception of the instrument to be separated. The latter two files serve as the “ground truth” against which the quality of the separated and MMS signals
can be measured. Such a set of files is most easily constructed using multitrack recordings. A number of artists make individual tracks (also referred to as “stems”) available online for remixing purposes. I obtained the individual tracks for the songs “Help Me Somebody” (Eno & Byrne 2006a), “A Secret Life” (Eno & Byrne 2006b), “Kunlarim Sensiz” (Sevara 2008), “Nude” (Radiohead 2008), “Only” (Nine Inch Nails 2008), “Resistencia” (Los De Abajo 2008), “Salala” (Kidjo 2008), and “Shock the Monkey” (Gabriel 1982). In each case, the highest audio quality files were downloaded and remixed using Logic Pro 7 and a MOTU 828mkII DAW. In addition, I used two songs from my own recordings, with the evocative titles “acoustic guitars” and “ring mod guitars” (these were mixed using Audio Desk 2.04 and a MOTU 828mkII).

In each case no additional audio processing was added to the tracks to be separated, with the exception of the limiting added to the two-track mixes. Note that some audio processing was added to tracks that were not intended to be separated, and some tracks already contained audio processing such as reverberation. I created a two-track (stereo) mix for each song and bounced the full stereo mix, a stereo mix of the track to be separated (the full mix with this track “soloed”), and a stereo mix of the mix without the track to be separated (the full mix with this track muted). These tracks were bounced as 16-bit wave files with a sampling rate of 44.1kHz and were each of 10 seconds in duration.

Next, I used my Matlab implementation of the source removal system to determine the proper values of the discrimination index $d$ and azimuth subspace width $H$ for each song. These values were set through a process of trial and error. The value for both $d$ and $H$ were set so as to maximize the quality of the separated source. For
each song, I then used the source removal system to produce output files of the separated source signal and the MMS signal produced by each of the four methods described in the previous section (azimugram complement, binary T-F mask, filtered T-F mask, and SMS modeled T-F mask). I repeated this process for each file using window sizes $N$ of 512, 1024, 2048, 4096, 8192, and 16384 samples, and hop sizes of $N/4$ and $N/8$. I then computed the SNR for each of these sets of files. The results of these tests are described below. Note that in all tests I used a value of 10 for $\beta$, the azimuth resolution (see section 2.2).

4.2 Results and Analysis

The results are summarized in tables 1 – 6 and plots 1 – 7 in appendix A. In tables 1a and 1b, the methods “az-compl,” “mask,” “filt-mask,” and “SMS mask” refer to the four methods used to compute the MMS signal, namely the azimugram complement, binary T-F mask, filtered binary T-F mask, and SMS modeled binary T-F mask methods described in the previous section. The “sep” method refers to the separated signal, computed using the azimugram resynthesis method described above. Because the three masking methods rely on the quality of the separated signal, this information was added to tables 1a and 1b to serve as a comparison between the average SNR of the separated signal versus the average SNR of the MMS signals as computed by the three masking methods. Each entry in tables 1a and 1b is the mean of the SNR computed for the results of the test file run created using the specified method, window size, and hop size. For example, in one test run, the system was used to create MMS signals for each of the ten test files using the azimugram complement source removal method with a window size $N$. 
of 1024 and a hop size of $N/8$. The SNR was computed for each of these MMS files, and the mean of these SNR values was computed. This value, 6.4154, is found in table 2 in the “1024” row and the “az-compl” column. Tables 2 – 6 show the SNR values computed for each file in each of the test runs. The average SNR from the previous example was computed using the data in the “1024” column in table 2b. Table 7 shows the abbreviated file name and its corresponding full name. In addition, this table shows the information used to separate out a specific source, namely the channel in which that source is predominant (left or right), the discrimination index $d$ and azimuth subspace width $H$ used in the computation of the separated signal and the azimugram complement.

Note that some of the entries in the tables are listed as “n/a” (not available). In these cases, I was unable to compute SMS model of the binary T-F mask due to hardware limitations (a lack of sufficient memory). These cases occur at the smaller window size settings. As the window size is decreased, the total number of frames used to analyze a given signal increases. The greater the number of frames, the larger the frequency estimate and magnitude estimate matrices used by the SMS function used to perform the SMS modeling - in this case, extractrax.m (Ellis 2003).

4.2.1 Source Separation and Azimugram Complement Source Removal

Plots 1a and 1b plot the data in tables 1a and 1b respectively (with the exception of the “sep” columns), and show rather conclusively that the azimugram complement method of signal removal is greatly superior to the three masking-based methods. This method produced SNR values of about 6 dB, regardless of window size or hop size. By
contrast, the other three methods produced SNR values ranging from roughly -1.5 dB to -1.0 dB.

Plots 2a and 2b, which plot the data from tables 2a and 2b respectively, are very similar to one another, indicating (as do plots 1a and 1b) that window size does not play a particularly important role in the performance of this method. However, the plots show that the performance of the azimugram complement method varies greatly depending on the input source. The method performed very well on the songs “Kunlarim Sensiz,” “A Secret Life,” “Shock the Monkey,” and “Resistencia,” with SNR values ranging from roughly 8.5 to 11 dB. By contrast, the performance of this method when using “Acoustic Guitars” as the input source was consistently close to 0 dB, roughly the same performance as the three masking-based methods.

This variation in the quality of the performance of the source separation method and the azimugram complement source removal method may be due in part to a flaw in the method used to create the left and right azimugrams. Consider equation (2.2.4a), which is used to construct the right azimugram:

\[
Az_R(k, i) = Lf(k) - g(i) \cdot Rf(k)
\]

(2.2.4a)

The next step in the computation of the right azimugram involves minimizing the right hand side of this equation, as described in equation (2.2.5a):

\[
Az_R(k, i) = \begin{cases} 
Az_R(k, i)_{\text{max}} - Az_R(k, i)_{\text{min}} & \text{if } Az_R(k, i) = Az_R(k)_{\text{min}} \\
0 & \text{otherwise}
\end{cases}
\]

(2.2.5a)
The right azimugram is used to separate a source that is predominant in the right channel (Barry et al. 2004, p. 2). However, consider the case of a source that is predominant in the left channel. For all the frequency bins $k$ that belong to that source, $L_f(k) > R_f(k)$. Therefore, equation (2.2.4a) is minimized when $g(i) = 1$ (recall that $0 \leq g(i) \leq 1$). This value of $g(i)$ corresponds to a discrimination index value ($d$) equal to $\beta$, which represents the middle of the stereo image. Thus, a source that is predominant in the left channel can appear in the center ($d = \beta$) of the right azimugram. If the signal is panned hard left, then, for all frequency bins $k$ belonging to that source, $R(f) = 0$, and equation (2.2.4a) yields the same value for all $g(i)$. In this case, in my implementation of the azimugram computation, $\theta$ is selected as the value of $g(i)$ that minimizes equation (2.2.4a). The left azimugram exhibits similar behavior.

In order to demonstrate this problem, I computed the left and right azimugrams of a test signal consisting of a single complex tone panned entirely in the right channel (test_tone_0_R.wav), shown in figure 4. As expected, the left azimugram contains no energy, while the right azimugram shows peaks at an azimuth of 0, which corresponds to the extreme right of the stereo image. I also computed the left and right azimugrams of a test signal consisting of a single complex tone panned in roughly midway between the center and the extreme right of the stereo image (test_tone_R.wav). These azimugrams are shown in figure 5. As expected, the right azimugram shows peaks at an azimuth value of approximately 45, about midway between azimuth 100 (center) and azimuth 0 (extreme right). However, these peaks also appear in the left azimugram at azimuth 100.
If the azimugrams represent the sources as they are distributed in the stereo image, one would not expect to find any energy in left azimugram. Also note that in figure 4, the maximum amplitude of the signal is approximately 300, whereas in figure 5, the maximum amplitude of the portion of the signal is approximately 160, and the maximum amplitude of the signal in the left azimugram is approximately 140. The energy of the signal is being split between the two azimugrams.

This problem affects the quality of the MMS signal as computed by the azimugram complement method. The computation of the azimugram complement will necessarily include unwanted peaks, since this computation will always include the center position of the azimugram corresponding to the channel where the source of interest is “non-predominant.” As an example, suppose we compute the MMS signal using the azimugram complement method with the signal in test_tone_R.wav as an input (see figure 5). In this case, the right channel of the MMS signal will include all the peaks in the right azimugram bounded by $d - H/2$ and $d + H/2$ (where $d$ in this case would be approximately 45). The left channel of the MMS signal will contain all the peaks of the left azimugram, including the unwanted ones at the center ($d=100$) azimuth.
Figure 4: left and right azimugrams for test_tone_0_R.wav
Figure 5: left and right azimugrams of test_tone_R.wav
Beyond the fact that this problem does not occur if the source to be separated is panned hard left or hard right (i.e., with a discrimination index $d=0$), it is not clear whether or not the value of the discrimination index $d$ affects the extent of the problem. For example, the distribution of the energy of a source between its “proper” location in the azimugram and the energy found in the center azimuth of the opposite azimugram depends on the value of $d$. However, it is important to note that this problem will in all likelihood occur with all the sources in a signal. In other words, it is likely that the azimugrams at the center azimuth will contain peaks that properly belong in the opposite azimugram, in addition to the peaks belonging to source panned to the middle.

Plot 7 plots the data in the “sep” columns of the tables 1a and 1b, and shows that the source separation algorithm on average yields an SNR between roughly 3.6 and 5.0 dB, with the best performance found with a window size of 2048 or 4096 samples. Other source separation algorithms perform at a somewhat higher level (Virtanen 2006, p. 294; Virtanen 2007 pp. 1071 – 1072), though the performance here seems reasonable, and is, in fact better than some other source separation implementations. As in the case of the azimugram complement source removal method, the performance of the source separation method does not vary much with hop size. Like the azimugram complement method, however, the performance of the separation method varies greatly depending on the input source, as show in plots 6a and 6b.

The problem with the azimugram computation, described above, may affect the quality of the separated signal. As demonstrated in the example above, the energy of a signal is distributed between the left and right azimugrams. Thus, when computing the separated signal, the peaks included in the magnitude spectrogram will have a smaller
magnitude than expected. It may be that, after reconstructing the time domain separated signal via the IFFT/overlap-add, proper normalization will mitigate this problem.

The quality of the separated signal is also dependent on the value of the azimuth subspace width, $H$. In general, larger values of $H$ will include more peaks that have drifted from the azimuth specified by the discrimination index $d$ and thus result in a higher quality separated signal. However, as the value of $H$ is increased, more peaks belonging to sources other than the source of interested are captured. The extent to which this effect occurs is dependent on various properties of the source, such as how the sources are distributed in azimuth space and how much the peaks of the sources drift from their respective discrimination indices.

4.2.2 Binary T-F Mask based Source Removal Methods

Plots 3 – 5 show the performance of the binary T-F mask, filtered binary T-F mask, and SMS modeled binary T-F mask for each of the 10 input sources (the corresponding data is in tables 3 – 5). As these plots demonstrate, the performance of each of these methods varies little with the input source. In almost every case, the SNR is between -2.0 and 0 dB.

The quality of all the T-F based masks (i.e., the binary T-F mask, the filtered binary T-F mask, and the SMS modeled binary T-F mask) ultimately depends on the quality of the separated signal. As shown in the previous section, the quality of this signal varies greatly. However, as plots 3 – 5 show, the T-F mask based methods exhibit roughly the same performance for each file. Therefore, other factors must be affecting the performance of the T-F mask based methods.
In the case of the filtered binary T-F mask, the de-noising filter often introduces smearing into the mask. This can be seen in figures 6a and 6b. Figure 6a shows the initial binary T-F mask computed from the separated electric guitar in the song “Nude.” Figure 6b shows this mask after the de-noising process. The attacks at roughly 1 sec., 4 sec., and 10 sec. are highly smeared by the de-noising filter. When the mask shown in figure 6b is applied to the magnitude spectrogram of the input signal, it removes many more frequencies than one would want.

The SMS modeling of the binary T-F mask is also highly problematic, as the output of the model varies greatly with the input signal. For example, figures 7a and 7b show SMS models computed from the binary T-F masks produced from the separated signals from “Help Me Somebody” and “Shock the Monkey,” respectively (note that the source of interest in “Help Me Somebody” was panned hard right; thus the left mask is empty). It is clear that the SMS modeling produced very different results, despite the fact that the separated sources were not radically different (an electric guitar and a synthesized marimba sound). The SMS process is quite complex, and its performance must be tuned using a number of different parameters.
Figure 6a: binary T-F mask
Figure 6b: filtered binary T-F mask
Figure 7a: SMS modeled binary T-F mask
Figure 7b: SMS modeled binary T-F mask
5. Conclusions and Future Work

5.1 Conclusions

Both quantitative and qualitative analysis indicate that this implementation of the ADRess source separation method is of reasonably good quality. Virtanen reports SNR values of 9.6 dB, 6.4 dB, and less than 0 dB for a number of different source separation methods (Virtanen 2006, p. 294; Virtanen 2007, pp. 1071 – 1072), while the ADRess algorithm yielded average SNR values between roughly 3.6 and 5.0 dB.

Quantitative and qualitative analysis of the shows that the azimugram method of source removal yields much higher quality results than the binary T-F mask based methods (i.e., the binary T-F mask, the filtered binary T-F mask, and the SMS modeled binary T-F mask methods). Further, it is likely that the performance of the azimugram complement method would be improved if the problem with the azimugram computation (described in section 4.2.1) were resolved. This problem seems to be relatively easy to fix, as its cause is evident. By contrast, the computation of a high quality binary T-F mask is a difficult one, in part due to the great variation in the nature of the signals used to compute the mask.

The azimugram complement method also has the advantage of computationally efficiency. From experience gained from performing the experiments described in section 4.1, filtering the binary T-F mask and computing the SMS model of the binary T-F mask is computationally costly; the computational cost of the latter method is made clear by tables 5a and 5b and plots 5a and 5b (computing the initial binary T-F mask did not incur an undue computational cost). The filtering method might be improved with the use of a more suitable de-noising filter. SMS modeling is also sensitive to the nature
of the input signal, and thus must be tuned to each input source for optimal performance. The complexity of SMS modeling seems to make it impractical for use in a source removal system.

Neither the ADRes method of source separation nor the azimugram complement method of source removal are applicable to all recordings (Barry et al. 2004, p. 1). However, for appropriate recordings, the azimugram complement method of source removal should be capable of producing high quality results. The binary T-F mask method may be useful for the task of source removal if a more sophisticated thresholding scheme were devised.

5.2 Future Work

The results described above suggest a number of improvements to the source removal and source separation methods. Because the azimuth complement method of source removal yielded the best results, fixing the error in the azimuth computation (described in section 4.2.1) would likely do much to improve the performance of the source removal. Although the poor results produced by the time-frequency mask based source removal methods suggest that they not useful for this task, they might be improved in a number of ways. The simple T-F mask computation would probably benefit from a more sophisticated method of computing the threshold values, perhaps one based on perceptual phenomena such as auditory masking. Improving the quality of simple T-F mask would also improve the quality of the filtered and SMS modeled T-F masks. The filtered T-F mask could also be improved with the use a more appropriate filter than the one employed here. The implementation of the system could also benefit from some relatively simple improvements such as a graphical user interface (GUI),
improved computational efficiency and memory usage, and improved resynthesis
techniques.

Further exploration into the causes of the differences in SNR values (for both the
separated and the MMS signals) between different sources might also suggest
improvements to the source removal and separation system. For example, it would likely
be useful to explore the relation between the azimuth subspace width $H$ and the quality of
the MMS and separated signals. Similarly, it might prove useful to explore the
relationship between the value of the discrimination index $d$ and the resulting quality, and
the relationship between the azimuth resolution $\beta$ and the resulting quality. It would also
be useful to explore how the source removal quality and source separation quality depend
on the frequency ranges of the source of interest. In this paper, the sources of interest
consisted of instruments with higher frequency ranges; it might be useful to test the
system using a lower frequency instrument, such as a bass guitar, as a source of interest.
The frequency resolution in lower frequencies could be increased through the use of a
time-frequency representation such as the Wavelet transform, instead of the Fourier
transform. This increased frequency resolution should improve the quality of the source
separation and source removal.

The source removal system could also benefit from a number of additional
features. There are a number of more complex features that would likely be useful. For
example, a method to compute optimal values of $d$ and $H$ could improve the quality of
the MMS and separated signals. By computing the values of $d$ and $H$ for each time
frame, this feature could also be used to track sources that move in the stereo image over
time.
## Appendix A: Test Results

**Table 1a: SNR means for hop size N/4**

<table>
<thead>
<tr>
<th>window size</th>
<th>method</th>
<th>az-compl</th>
<th>mask</th>
<th>filt-mask</th>
<th>SMS mask</th>
<th>sep</th>
</tr>
</thead>
<tbody>
<tr>
<td>512</td>
<td></td>
<td>6.2502</td>
<td>-1.5545</td>
<td>-1.2938</td>
<td>n/a</td>
<td>3.5954</td>
</tr>
<tr>
<td>1024</td>
<td></td>
<td>6.3748</td>
<td>-1.1454</td>
<td>-0.95387</td>
<td>n/a</td>
<td>4.1681</td>
</tr>
<tr>
<td>2048</td>
<td></td>
<td>6.4508</td>
<td>-0.91947</td>
<td>-0.84404</td>
<td>0.82728</td>
<td>4.5815</td>
</tr>
<tr>
<td>4096</td>
<td></td>
<td>6.4271</td>
<td>-0.73318</td>
<td>-0.70443</td>
<td>-0.72054</td>
<td>4.9072</td>
</tr>
<tr>
<td>8192</td>
<td></td>
<td>6.2853</td>
<td>-0.59938</td>
<td>-0.55515</td>
<td>-0.60453</td>
<td>4.993</td>
</tr>
<tr>
<td>16384</td>
<td></td>
<td>5.9307</td>
<td>-0.52111</td>
<td>-0.47627</td>
<td>-0.59492</td>
<td>4.745</td>
</tr>
</tbody>
</table>

**Table 1b: SNR means for hop size N/8**

<table>
<thead>
<tr>
<th>window size</th>
<th>method</th>
<th>az-compl</th>
<th>mask</th>
<th>filt-mask</th>
<th>SMS mask</th>
<th>sep</th>
</tr>
</thead>
<tbody>
<tr>
<td>512</td>
<td></td>
<td>6.292</td>
<td>-1.7978</td>
<td>-1.5376</td>
<td>n/a</td>
<td>3.6875</td>
</tr>
<tr>
<td>1024</td>
<td></td>
<td>6.4154</td>
<td>-1.279</td>
<td>-1.0643</td>
<td>n/a</td>
<td>4.2139</td>
</tr>
<tr>
<td>2048</td>
<td></td>
<td>6.5053</td>
<td>-1.0204</td>
<td>-0.88775</td>
<td>n/a</td>
<td>4.6481</td>
</tr>
<tr>
<td>4096</td>
<td></td>
<td>6.4383</td>
<td>-1.0318</td>
<td>-0.99277</td>
<td>n/a</td>
<td>4.996</td>
</tr>
<tr>
<td>8192</td>
<td></td>
<td>6.2856</td>
<td>-0.8356</td>
<td>-0.80433</td>
<td>-0.80889</td>
<td>5.0619</td>
</tr>
<tr>
<td>16384</td>
<td></td>
<td>5.9376</td>
<td>-0.83071</td>
<td>-0.80307</td>
<td>-0.78078</td>
<td>4.7911</td>
</tr>
</tbody>
</table>
Table 2a: SNR for az-compl with hop size N/4

<table>
<thead>
<tr>
<th>file</th>
<th>window size</th>
<th>512</th>
<th>1024</th>
<th>2048</th>
<th>4096</th>
<th>8192</th>
<th>16384</th>
</tr>
</thead>
<tbody>
<tr>
<td>ac gtrs</td>
<td></td>
<td>0.40205</td>
<td>0.32698</td>
<td>0.20642</td>
<td>0.21617</td>
<td>0.20212</td>
<td>0.091168</td>
</tr>
<tr>
<td>help me</td>
<td></td>
<td>5.5296</td>
<td>5.8814</td>
<td>6.2723</td>
<td>6.752</td>
<td>6.6871</td>
<td>6.5225</td>
</tr>
<tr>
<td>kunlarim</td>
<td></td>
<td>11.11</td>
<td>11.178</td>
<td>11.142</td>
<td>10.691</td>
<td>9.9775</td>
<td>8.7171</td>
</tr>
<tr>
<td>nude</td>
<td></td>
<td>4.8605</td>
<td>5.9152</td>
<td>6.4107</td>
<td>7.2705</td>
<td>8.0114</td>
<td>7.668</td>
</tr>
<tr>
<td>only</td>
<td></td>
<td>4.3373</td>
<td>4.407</td>
<td>4.3701</td>
<td>4.4844</td>
<td>4.2927</td>
<td>4.1835</td>
</tr>
<tr>
<td>rmg</td>
<td></td>
<td>3.5921</td>
<td>3.8886</td>
<td>3.8589</td>
<td>3.6653</td>
<td>3.6537</td>
<td>3.6066</td>
</tr>
<tr>
<td>salala</td>
<td></td>
<td>2.6645</td>
<td>2.3808</td>
<td>2.1145</td>
<td>1.9041</td>
<td>1.815</td>
<td>1.6717</td>
</tr>
<tr>
<td>secret</td>
<td></td>
<td>10.208</td>
<td>0.272</td>
<td>10.216</td>
<td>9.8552</td>
<td>9.2913</td>
<td>8.9089</td>
</tr>
</tbody>
</table>

Table 2b: SNR for az-compl with hop size N/8

<table>
<thead>
<tr>
<th>file</th>
<th>window size</th>
<th>512</th>
<th>1024</th>
<th>2048</th>
<th>4096</th>
<th>8192</th>
<th>16384</th>
</tr>
</thead>
<tbody>
<tr>
<td>ac gtrs</td>
<td></td>
<td>0.40479</td>
<td>0.29747</td>
<td>0.17427</td>
<td>0.19757</td>
<td>0.18834</td>
<td>0.083475</td>
</tr>
<tr>
<td>help me</td>
<td></td>
<td>5.6213</td>
<td>5.9152</td>
<td>6.3238</td>
<td>6.8078</td>
<td>6.7788</td>
<td>6.5564</td>
</tr>
<tr>
<td>kunlarim</td>
<td></td>
<td>10.966</td>
<td>11.156</td>
<td>11.418</td>
<td>10.733</td>
<td>10.076</td>
<td>8.732</td>
</tr>
<tr>
<td>only</td>
<td></td>
<td>4.4408</td>
<td>4.4016</td>
<td>4.4123</td>
<td>4.4424</td>
<td>4.3184</td>
<td>4.1855</td>
</tr>
<tr>
<td>rmg</td>
<td></td>
<td>3.6163</td>
<td>3.9006</td>
<td>3.8729</td>
<td>3.6671</td>
<td>3.6614</td>
<td>3.6075</td>
</tr>
<tr>
<td>salala</td>
<td></td>
<td>2.6599</td>
<td>2.4038</td>
<td>2.1038</td>
<td>1.9462</td>
<td>1.7717</td>
<td>1.6832</td>
</tr>
</tbody>
</table>
Table 3a: SNR for binary T-F mask with hop size N/4

<table>
<thead>
<tr>
<th>file</th>
<th>window size</th>
<th>512</th>
<th>1024</th>
<th>2048</th>
<th>4096</th>
<th>8192</th>
<th>16384</th>
</tr>
</thead>
<tbody>
<tr>
<td>ac gtrs</td>
<td></td>
<td>-0.43271</td>
<td>-0.84231</td>
<td>-0.45438</td>
<td>-0.16034</td>
<td>-0.10955</td>
<td>-0.077588</td>
</tr>
<tr>
<td>help me</td>
<td></td>
<td>-1.5716</td>
<td>-1.0623</td>
<td>-0.72888</td>
<td>-0.42657</td>
<td>-0.31701</td>
<td>-0.33019</td>
</tr>
<tr>
<td>kunlarim</td>
<td></td>
<td>-2.928</td>
<td>-1.2388</td>
<td>-0.77646</td>
<td>-0.91224</td>
<td>-0.75029</td>
<td>-0.69243</td>
</tr>
<tr>
<td>nude</td>
<td></td>
<td>-1.9993</td>
<td>-2.0694</td>
<td>-1.4338</td>
<td>-1.53</td>
<td>-1.8418</td>
<td>-1.1695</td>
</tr>
<tr>
<td>only</td>
<td></td>
<td>-0.44908</td>
<td>-0.64177</td>
<td>-0.65739</td>
<td>-0.59595</td>
<td>-0.39111</td>
<td>-0.41424</td>
</tr>
<tr>
<td>resist.</td>
<td></td>
<td>-2.1317</td>
<td>-1.1032</td>
<td>-0.88028</td>
<td>-0.69765</td>
<td>-0.5303</td>
<td>-0.46575</td>
</tr>
<tr>
<td>rmg</td>
<td></td>
<td>-2.0223</td>
<td>-2.0868</td>
<td>-1.799</td>
<td>-1.3081</td>
<td>-1.2122</td>
<td>-0.89455</td>
</tr>
<tr>
<td>salala</td>
<td></td>
<td>-0.62702</td>
<td>-0.34375</td>
<td>-0.33901</td>
<td>-0.22521</td>
<td>-0.19088</td>
<td>-0.24878</td>
</tr>
<tr>
<td>secret</td>
<td></td>
<td>-1.5329</td>
<td>-0.76085</td>
<td>-0.69429</td>
<td>-0.54347</td>
<td>0.06881</td>
<td>-0.20466</td>
</tr>
<tr>
<td>shock</td>
<td></td>
<td>-1.8502</td>
<td>-1.3051</td>
<td>-1.4312</td>
<td>-0.93227</td>
<td>-0.7195</td>
<td>-0.71342</td>
</tr>
</tbody>
</table>

Table 3b: SNR for binary T-F mask for hop size N/8

<table>
<thead>
<tr>
<th>file</th>
<th>window size</th>
<th>512</th>
<th>1024</th>
<th>2048</th>
<th>4096</th>
<th>8192</th>
<th>16384</th>
</tr>
</thead>
<tbody>
<tr>
<td>ac gtrs</td>
<td></td>
<td>-1.0668</td>
<td>-0.93808</td>
<td>-0.66646</td>
<td>-0.36661</td>
<td>-0.24819</td>
<td>-0.1795</td>
</tr>
<tr>
<td>help me</td>
<td></td>
<td>-1.8132</td>
<td>-0.92347</td>
<td>-0.69628</td>
<td>-0.71629</td>
<td>-0.44774</td>
<td>-0.4435</td>
</tr>
<tr>
<td>kunlarim</td>
<td></td>
<td>-2.9284</td>
<td>-1.8014</td>
<td>-0.70714</td>
<td>-0.80905</td>
<td>-0.99882</td>
<td>-1.0327</td>
</tr>
<tr>
<td>nude</td>
<td></td>
<td>-1.6413</td>
<td>-1.4858</td>
<td>-0.94408</td>
<td>-1.4295</td>
<td>-1.3512</td>
<td>-2.1989</td>
</tr>
<tr>
<td>only</td>
<td></td>
<td>-0.81291</td>
<td>-0.67492</td>
<td>-1.1226</td>
<td>-1.0546</td>
<td>-0.94775</td>
<td>-0.61331</td>
</tr>
<tr>
<td>resist.</td>
<td></td>
<td>-2.99</td>
<td>-1.5659</td>
<td>-1.0881</td>
<td>-1.1868</td>
<td>-0.93022</td>
<td>-0.6599</td>
</tr>
<tr>
<td>rmg</td>
<td></td>
<td>-1.9674</td>
<td>-1.9922</td>
<td>-2.0574</td>
<td>-1.8177</td>
<td>-1.6577</td>
<td>-1.6291</td>
</tr>
<tr>
<td>salala</td>
<td></td>
<td>-0.85089</td>
<td>-0.78741</td>
<td>-0.50999</td>
<td>-0.53322</td>
<td>-0.28726</td>
<td>-0.26786</td>
</tr>
<tr>
<td>secret</td>
<td></td>
<td>-2.5431</td>
<td>-1.2755</td>
<td>-1.1758</td>
<td>-0.87885</td>
<td>-0.50362</td>
<td>-0.16356</td>
</tr>
<tr>
<td>shock</td>
<td></td>
<td>-1.3637</td>
<td>-1.3449</td>
<td>-1.2356</td>
<td>-1.5252</td>
<td>-0.98345</td>
<td>-1.1189</td>
</tr>
</tbody>
</table>
Table 4a: SNR for filtered binary T-F mask for hop size N/4

<table>
<thead>
<tr>
<th>file</th>
<th>window size</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>512</td>
</tr>
<tr>
<td>ac gtrs</td>
<td>-0.42889</td>
</tr>
<tr>
<td>help me</td>
<td>-1.5658</td>
</tr>
<tr>
<td>kunlarim</td>
<td>-2.8874</td>
</tr>
<tr>
<td>nude</td>
<td>-1.011</td>
</tr>
<tr>
<td>only</td>
<td>-0.14889</td>
</tr>
<tr>
<td>resist.</td>
<td>-2.0023</td>
</tr>
<tr>
<td>rmg</td>
<td>-1.3299</td>
</tr>
<tr>
<td>salala</td>
<td>-0.39166</td>
</tr>
<tr>
<td>secret</td>
<td>-1.4759</td>
</tr>
<tr>
<td>shock</td>
<td>-1.6957</td>
</tr>
</tbody>
</table>

Table 4b: SNR for filtered binary T-F mask for hop size N/8

<table>
<thead>
<tr>
<th>file</th>
<th>window size</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>512</td>
</tr>
<tr>
<td>ac gtrs</td>
<td>-0.81042</td>
</tr>
<tr>
<td>help me</td>
<td>-1.5813</td>
</tr>
<tr>
<td>kunlarim</td>
<td>-2.8908</td>
</tr>
<tr>
<td>nude</td>
<td>-1.403</td>
</tr>
<tr>
<td>only</td>
<td>-0.46</td>
</tr>
<tr>
<td>resist.</td>
<td>-2.5878</td>
</tr>
<tr>
<td>rmg</td>
<td>-1.1404</td>
</tr>
<tr>
<td>salala</td>
<td>-0.60023</td>
</tr>
<tr>
<td>secret</td>
<td>-2.5533</td>
</tr>
<tr>
<td>shock</td>
<td>-1.3489</td>
</tr>
</tbody>
</table>
Table 5a: SNR for SMS modeled binary T-F mask for hop size N/4

<table>
<thead>
<tr>
<th>file</th>
<th>window size</th>
<th>512</th>
<th>1024</th>
<th>2048</th>
<th>4096</th>
<th>8192</th>
<th>16384</th>
</tr>
</thead>
<tbody>
<tr>
<td>ac gtrs</td>
<td>n/a</td>
<td>n/a</td>
<td>-0.33092</td>
<td>-0.54283</td>
<td>-0.16446</td>
<td>-0.53151</td>
<td></td>
</tr>
<tr>
<td>help me</td>
<td>n/a</td>
<td>n/a</td>
<td>-1.0248</td>
<td>-0.46136</td>
<td>-0.4023</td>
<td>-0.37858</td>
<td></td>
</tr>
<tr>
<td>kunlarim</td>
<td>n/a</td>
<td>n/a</td>
<td>-0.78714</td>
<td>-0.92358</td>
<td>-0.77368</td>
<td>-0.71444</td>
<td></td>
</tr>
<tr>
<td>nude</td>
<td>n/a</td>
<td>n/a</td>
<td>-1.6611</td>
<td>-1.2819</td>
<td>-2.0071</td>
<td>-1.7108</td>
<td></td>
</tr>
<tr>
<td>only</td>
<td>n/a</td>
<td>n/a</td>
<td>-0.44606</td>
<td>-0.21996</td>
<td>-0.09349</td>
<td>-0.044714</td>
<td></td>
</tr>
<tr>
<td>resist.</td>
<td>n/a</td>
<td>n/a</td>
<td>-0.747</td>
<td>-0.63226</td>
<td>-0.41907</td>
<td>-0.42569</td>
<td></td>
</tr>
<tr>
<td>rmg</td>
<td>n/a</td>
<td>n/a</td>
<td>-0.83032</td>
<td>-1.3327</td>
<td>-1.1481</td>
<td>-0.85841</td>
<td></td>
</tr>
<tr>
<td>salala</td>
<td>n/a</td>
<td>n/a</td>
<td>-0.26648</td>
<td>-0.21133</td>
<td>-0.26991</td>
<td>-0.26235</td>
<td></td>
</tr>
<tr>
<td>secret</td>
<td>n/a</td>
<td>n/a</td>
<td>-0.70871</td>
<td>-0.61951</td>
<td>0.027259</td>
<td>-0.25063</td>
<td></td>
</tr>
<tr>
<td>shock</td>
<td>n/a</td>
<td>n/a</td>
<td>-1.4702</td>
<td>-0.97997</td>
<td>-0.79447</td>
<td>-0.77208</td>
<td></td>
</tr>
</tbody>
</table>

Table 5b: SNR for SMS modeled binary T-F mask for hop size N/8

<table>
<thead>
<tr>
<th>file</th>
<th>window size</th>
<th>512</th>
<th>1024</th>
<th>2048</th>
<th>4096</th>
<th>8192</th>
<th>16384</th>
</tr>
</thead>
<tbody>
<tr>
<td>ac gtrs</td>
<td>n/a</td>
<td>n/a</td>
<td>n/a</td>
<td>n/a</td>
<td>-0.70978</td>
<td>-0.43681</td>
<td>-0.29178</td>
</tr>
<tr>
<td>help me</td>
<td>n/a</td>
<td>n/a</td>
<td>n/a</td>
<td>n/a</td>
<td>-0.91718</td>
<td>-0.49902</td>
<td>-0.52025</td>
</tr>
<tr>
<td>kunlarim</td>
<td>n/a</td>
<td>n/a</td>
<td>n/a</td>
<td>n/a</td>
<td>-0.80933</td>
<td>-0.89129</td>
<td>-1.0491</td>
</tr>
<tr>
<td>nude</td>
<td>n/a</td>
<td>n/a</td>
<td>n/a</td>
<td>n/a</td>
<td>-1.3271</td>
<td>-1.3501</td>
<td>-1.7052</td>
</tr>
<tr>
<td>only</td>
<td>n/a</td>
<td>n/a</td>
<td>n/a</td>
<td>n/a</td>
<td>n/a</td>
<td>-0.31095</td>
<td>-0.28102</td>
</tr>
<tr>
<td>resist.</td>
<td>n/a</td>
<td>n/a</td>
<td>n/a</td>
<td>n/a</td>
<td>n/a</td>
<td>-0.87675</td>
<td>-0.54528</td>
</tr>
<tr>
<td>rmg</td>
<td>n/a</td>
<td>n/a</td>
<td>n/a</td>
<td>n/a</td>
<td>n/a</td>
<td>-1.7669</td>
<td>-1.5458</td>
</tr>
<tr>
<td>salala</td>
<td>n/a</td>
<td>n/a</td>
<td>n/a</td>
<td>n/a</td>
<td>n/a</td>
<td>-0.32895</td>
<td>-0.41547</td>
</tr>
<tr>
<td>secret</td>
<td>n/a</td>
<td>n/a</td>
<td>n/a</td>
<td>n/a</td>
<td>n/a</td>
<td>-0.59298</td>
<td>-0.23545</td>
</tr>
<tr>
<td>shock</td>
<td>n/a</td>
<td>n/a</td>
<td>n/a</td>
<td>n/a</td>
<td>n/a</td>
<td>-1.0352</td>
<td>-1.2184</td>
</tr>
</tbody>
</table>

58
### Table 6a: SNR for separated signal for hop size N/4

<table>
<thead>
<tr>
<th>file</th>
<th>512</th>
<th>1024</th>
<th>2048</th>
<th>4096</th>
<th>8192</th>
<th>16384</th>
</tr>
</thead>
<tbody>
<tr>
<td>ac gtrs</td>
<td>4.3294</td>
<td>4.1569</td>
<td>3.6358</td>
<td>3.7841</td>
<td>3.8489</td>
<td>3.4571</td>
</tr>
<tr>
<td>help me</td>
<td>5.4908</td>
<td>5.9376</td>
<td>5.9419</td>
<td>6.3788</td>
<td>6.2933</td>
<td>6.1885</td>
</tr>
<tr>
<td>kunlarim</td>
<td>4.4891</td>
<td>4.3477</td>
<td>3.9817</td>
<td>3.3904</td>
<td>3.5944</td>
<td>2.8112</td>
</tr>
<tr>
<td>only</td>
<td>-0.3563</td>
<td>-0.73637</td>
<td>-0.84032</td>
<td>-0.38168</td>
<td>-0.34828</td>
<td>-0.62419</td>
</tr>
<tr>
<td>resist.</td>
<td>0.87776</td>
<td>0.87721</td>
<td>1.7862</td>
<td>2.3389</td>
<td>1.2905</td>
<td>0.97396</td>
</tr>
<tr>
<td>rmg</td>
<td>4.2718</td>
<td>5.4984</td>
<td>6.3835</td>
<td>6.8566</td>
<td>7.1349</td>
<td>7.3742</td>
</tr>
<tr>
<td>salala</td>
<td>-0.17652</td>
<td>-0.30786</td>
<td>0.10243</td>
<td>0.32332</td>
<td>0.38827</td>
<td>0.13571</td>
</tr>
<tr>
<td>secret</td>
<td>5.7863</td>
<td>8.7579</td>
<td>9.6509</td>
<td>9.999</td>
<td>10.499</td>
<td>10.294</td>
</tr>
<tr>
<td>shock</td>
<td>5.6241</td>
<td>6.0823</td>
<td>6.8867</td>
<td>7.3699</td>
<td>7.7367</td>
<td>6.9839</td>
</tr>
</tbody>
</table>

### Table 6b: SNR for separated signal for hop size N/8

<table>
<thead>
<tr>
<th>file</th>
<th>512</th>
<th>1024</th>
<th>2048</th>
<th>4096</th>
<th>8192</th>
<th>16384</th>
</tr>
</thead>
<tbody>
<tr>
<td>ac gtrs</td>
<td>4.3224</td>
<td>4.0961</td>
<td>3.4652</td>
<td>3.7549</td>
<td>3.7766</td>
<td>3.4816</td>
</tr>
<tr>
<td>help me</td>
<td>5.536</td>
<td>5.9354</td>
<td>5.943</td>
<td>6.3079</td>
<td>6.3832</td>
<td>6.0182</td>
</tr>
<tr>
<td>only</td>
<td>-0.30569</td>
<td>-0.93865</td>
<td>-0.75625</td>
<td>-0.29111</td>
<td>-0.36618</td>
<td>-0.56995</td>
</tr>
<tr>
<td>resist.</td>
<td>1.1328</td>
<td>0.90334</td>
<td>1.8959</td>
<td>2.3674</td>
<td>1.2063</td>
<td>0.78222</td>
</tr>
<tr>
<td>salala</td>
<td>-0.21186</td>
<td>-0.29562</td>
<td>0.11283</td>
<td>0.40901</td>
<td>0.41511</td>
<td>0.12576</td>
</tr>
<tr>
<td>shock</td>
<td>5.6028</td>
<td>6.1881</td>
<td>7.0089</td>
<td>7.5923</td>
<td>7.7682</td>
<td>7.0255</td>
</tr>
</tbody>
</table>

### Table 7: Song abbreviations, names, discrimination indices, and azimuth subspace widths and side used in separation/removal (note that $\beta=10$).

<table>
<thead>
<tr>
<th>abbreviation</th>
<th>song name</th>
<th>side</th>
<th>$d$</th>
<th>$H$</th>
</tr>
</thead>
<tbody>
<tr>
<td>ac gtrs</td>
<td>acoustic guitars</td>
<td>L</td>
<td>9</td>
<td>2</td>
</tr>
<tr>
<td>help me</td>
<td>Help Me Somebody</td>
<td>R</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>kunlarim</td>
<td>Kunlarim Sensiz</td>
<td>L</td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>nude</td>
<td>Nude</td>
<td>L</td>
<td>4</td>
<td>6</td>
</tr>
<tr>
<td>only</td>
<td>Only</td>
<td>L</td>
<td>8</td>
<td>2</td>
</tr>
<tr>
<td>resist.</td>
<td>Resistencia</td>
<td>L</td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>rmg</td>
<td>ring mod guitars</td>
<td>R</td>
<td>7</td>
<td>4</td>
</tr>
<tr>
<td>salala</td>
<td>Salala</td>
<td>R</td>
<td>9</td>
<td>2</td>
</tr>
<tr>
<td>secret</td>
<td>A Secret Life</td>
<td>L</td>
<td>4</td>
<td>2</td>
</tr>
<tr>
<td>shock</td>
<td>Shock the Monkey</td>
<td>R</td>
<td>2</td>
<td>2</td>
</tr>
</tbody>
</table>
Plots 1a and 1b: mean SNR of MMS signals (all source removal methods)
Plots 2a and 2b: SNR of MMS signals (azimugram complement)
Plots 3a and 3b: SNR of MMS signals (binary T-F mask)
Plots 4a and 4b: SNR of MMS signals (filtered binary T-F mask)
Plots 5a and 5b: SNR of MMS signals (SMS modeled binary T-F mask)
Plot 6a and 6b: SNR of separated signals
Plot 7: mean SNR of separated signals

![Graph showing mean SNR of separated signals with window size (samples) on the x-axis and SNR (dB) on the y-axis. The graph compares different hop sizes (N/4 and N/8).]
References


