8 The DUET Blind Source Separation Algorithm*

Scott Rickard

University College Dublin
Belfield, Dublin 4, Ireland
E-mail: scott.rickard@ucd.ie

Abstract. This chapter presents a tutorial on the DUET Blind Source Separation method which can separate any number of sources using only two mixtures. The method is valid when sources are W-disjoint orthogonal, that is, when the supports of the windowed Fourier transform of the signals in the mixture are disjoint. For anechoic mixtures of attenuated and delayed sources, the method allows one to estimate the mixing parameters by clustering relative attenuation-delay pairs extracted from the ratios of the time–frequency representations of the mixtures. The estimates of the mixing parameters are then used to partition the time–frequency representation of one mixture to recover the original sources. The technique is valid even in the case when the number of sources is larger than the number of mixtures. The method is particularly well suited to speech mixtures because the time–frequency representation of speech is sparse and this leads to W-disjoint orthogonality. The algorithm is easily coded and a simple MATLAB® implementation is presented[1]. Additionally in this chapter, two strategies which allow DUET to be applied to situations where the microphones are far apart are presented; this removes a major limitation of the original method.

8.1 Introduction

In the field of blind source separation (BSS), assumptions on the statistical properties of the sources usually provide a basis for the demixing algorithm [1]. Some common assumptions are that the sources are statistically independent [2, 3], are statistically orthogonal [4], are nonstationary [5], or can be generated by finite dimensional model spaces [6]. The independence and orthogonality assumptions can be verified experimentally for speech signals. Some of these methods work well for instantaneous demixing, but fail if propagation delays are present. Additionally, many algorithms are computationally intensive as they require the estimation of higher-order statistical moments or the optimization of a nonlinear cost function.

One area of research in blind source separation that is particularly challenging is when there are more sources than mixtures. We refer to such a case

*This material is based upon work supported by the Science Foundation Ireland under the PIYRA Programme.

[1]The author is happy to provide the MATLAB® code implementation of DUET presented here.

S. Makino et al. (eds.), Blind Speech Separation, 217–241.

as *degenerate*. Degenerate blind source separation poses a challenge because the mixing matrix is not invertible. Thus the traditional method of demixing by estimating the inverse mixing matrix does not work. As a result, most BSS research has focussed on the square or overdetermined (nondegenerate) case.

Despite the difficulties, there are several approaches for dealing with degenerate mixtures. For example, [7] estimates an arbitrary number of sources from a single mixture by modeling the signals as autoregressive processes. However, this is achieved at a price of approximating signals by autoregressive stochastic processes, which can be too restrictive. Another example of degenerate separation uses higher order statistics to demix three sources from two mixtures [8]. This approach is not feasible however for a large number of sources since the use of higher order statistics of mixtures leads to an explosion in computational complexity. Similar in spirit to DUET, van Hulle employed a clustering method for relative amplitude parameter estimation and degenerate demixing [9]. The assumptions used by van Hulle were that only one signal at a given time is nonzero and that mixing is instantaneous, that is, there is only a relative amplitude mixing parameter associated with each source. In real world acoustic environments, these assumptions are not valid.

DUET, the Degenerate Unmixing Estimation Technique, solves the degenerate demixing problem in an efficient and robust manner. The underlying principle behind DUET can be summarized in one sentence:

> It is possible to blindly separate an arbitrary number of sources given just two anechoic mixtures provided the time–frequency representations of the sources do not overlap too much, which is true for speech.

The way that DUET separates degenerate mixtures is by partitioning the time–frequency representation of one of the mixtures. In other words, DUET assumes the sources are already 'separate' in that, in the time–frequency plane, the sources are *disjoint*. The 'demixing' process is then simply a partitioning of the time–frequency plane. Although the assumption of disjointness may seem unreasonable for simultaneous speech, it is approximately true. By approximately, we mean that the time–frequency points which contain significant contributions to the average energy of the mixture are very likely to be dominated by a contribution from only one source. Stated another way, two people rarely excite the same frequency at the same time.

This chapter has the following structure. In Sect. 8.2 we discuss the assumptions of anechoic mixing, W-disjoint orthogonality, local stationarity, closely spaced microphones, and different source spatial signatures which lead to the main observation. In Sect. 8.3 we describe the construction of the 2D weighted histogram which is the key component of the mixing parameter estimation in DUET and we describe the DUET algorithm. In Sect. 8.4

we propose two possible extensions to DUET which eliminate the requirement that the microphones be close together. In Sect. 8.5 we discuss a proposed measure of disjointness. After the conclusions in Sect. 8.6, we provide the MATLAB® utility functions used in the earlier sections.

8.2 Assumptions

8.2.1 Anechoic Mixing

Consider the mixtures of N source signals, $s_j(t), j = 1, \ldots, N$, being received at a pair of microphones where only the direct path is present. In this case, without loss of generality, we can absorb the attenuation and delay parameters of the first mixture, $x_1(t)$, into the definition of the sources. The two **anechoic mixtures** can thus be expressed as,

$$x_1(t) = \sum_{j=1}^{N} s_j(t), \tag{8.1}$$

$$x_2(t) = \sum_{j=1}^{N} a_j s_j(t - \delta_j), \tag{8.2}$$

where N is the number of sources, δ_j is the arrival delay between the sensors, and a_j is a relative attenuation factor corresponding to the ratio of the attenuations of the paths between sources and sensors. We use Δ to denote the maximal possible delay between sensors, and thus, $|\delta_j| \leq \Delta, \forall j$. The anechoic mixing model is not realistic in that it does not represent echoes, that is, multiple paths from each source to each mixture. However, in spite of this limitation, the DUET method, which is based on the anechoic model, has proven to be quite robust even when applied to echoic mixtures.

8.2.2 W-Disjoint Orthogonality

We call two functions $s_j(t)$ and $s_k(t)$ **W-disjoint orthogonal** if, for a given windowing function $W(t)$, the supports of the windowed Fourier transforms of $s_j(t)$ and $s_k(t)$ are disjoint. The windowed Fourier transform of $s_j(t)$ is defined,

$$\hat{s}_j(\tau, \omega) := F^W[s_j](\tau, \omega) := \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} W(t - \tau) s_j(t) e^{-i\omega t} dt. \tag{8.3}$$

The W-disjoint orthogonality assumption can be stated concisely,

$$\hat{s}_j(\tau,\omega)\hat{s}_k(\tau,\omega) = 0, \quad \forall\tau,\omega, \quad \forall j \neq k. \tag{8.4}$$

This assumption is the mathematical idealization of the condition that it is likely that every time–frequency point in the mixture with significant energy is dominated by the contribution of one source. Note that, if $W(t) \equiv 1$, $\hat{s}_j(\tau,\omega)$ becomes the Fourier transform of $s_j(t)$, which we will denote $\hat{s}_j(\omega)$. In this case, W-disjoint orthogonality can be expressed,

$$\hat{s}_j(\omega)\hat{s}_k(\omega) = 0, \forall j \neq k, \forall\omega, \tag{8.5}$$

which we call **disjoint orthogonality**.

W-disjoint orthogonality is crucial to DUET because it allows for the separation of a mixture into its component sources using a binary mask. Consider the mask which is the indicator function for the support of $\hat{s}_j$,

$$M_j(\tau,\omega) := \begin{cases} 1 & \hat{s}_j(\tau,\omega) \neq 0 \\ 0 & \text{otherwise.} \end{cases} \tag{8.6}$$

M_j separates $\hat{s}_j$ from the mixture via

$$\hat{s}_j(\tau,\omega) = M_j(\tau,\omega)\hat{x}_1(\tau,\omega), \forall\tau,\omega. \tag{8.7}$$

So if we could determine the masks which are the indicator functions for each source, we can separate the sources by partitioning. The question is: how do we determine the masks? As we will see shortly, the answer is we label each time–frequency point with the delay and attenuation differences that explain the time–frequency magnitude and phase between the two mixtures, and these delay-attenuation pairs cluster into groups, one group for each source.

8.2.3 Local Stationarity

A well-known Fourier transform pair is:

$$s_j(t-\delta) \leftrightarrow e^{-i\omega\delta}\hat{s}_j(\omega). \tag{8.8}$$

We can state this using the notation of (8.3) as,

$$F^W[s_j(\cdot-\delta)](\tau,\omega) = e^{-i\omega\delta}F^W[s_j(\cdot)](\tau,\omega), \tag{8.9}$$

when $W(t) \equiv 1$. The above equation is not necessarily true, however, when $W(t)$ is a windowing function. For example, if the windowing function were a Hamming window of length 40 ms, there is no reason to believe that two 40 ms windows of speech separated by, say, several seconds are related by a phase shift. However, for shifts which are small relative to the window size, (8.9) will hold even if $W(t)$ has finite support. This can be thought of as a

form of a narrowband assumption in array processing [10], but this label is perhaps misleading in that speech is not narrowband, and *local stationarity* seems a more appropriate moniker. What is necessary for DUET is that (8.9) holds for all δ, $|\delta| \leq \Delta$, even when $W(t)$ has finite support, where Δ is the maximum time difference possible in the mixing model (the microphone separation divided by the speed of signal propagation). We formally state the **local stationarity** assumption as,

$$F^W[s_j(\cdot - \delta)](\omega, \tau) = e^{-i\omega\delta} F^W[s_j(\cdot)](\omega, \tau), \quad \forall \delta, |\delta| \leq \Delta, \tag{8.10}$$

where the change from (8.9) is the inclusion of the limitation of the range of δ for which the equality holds.

8.2.4 Microphones Close Together

Additionally, one crucial issue is that DUET is based, as we shall soon see, on the extraction of attenuation and delay mixing parameters estimates from each time–frequency point. We will utilize the local stationarity assumption to turn the delay in time into a multiplicative factor in time–frequency. Of course, this multiplicative factor $e^{-i\omega\delta}$ only uniquely specifies δ if $|\omega\delta| < \pi$ as otherwise we have an ambiguity due to phase-wrap. So we require,

$$|\omega\delta_j| < \pi, \quad \forall \omega, \forall j, \tag{8.11}$$

to avoid phase ambiguity. This is guaranteed when the microphones are separated by less than $\pi c/\omega_m$ where ω_m is the maximum frequency present in the sources and c is the speed of sound. For example, for a maximum frequency of 16 kHz the microphones must be placed within approximately 1 cm of each other, and for a maximum frequency of 8 kHz the microphones must be placed within approximately 2 cm of each other.

8.2.5 Different Spatial Signatures

In the anechoic mixing model described by (8.1) and (8.2), if two sources have identical spatial signatures, that is identical relative attenuation and relative delay mixing parameters, then they can be combined into one source without changing the model. In this case, the DUET technique will fail to separate them from each other because DUET uses only the relative attenuation and relative delay mixing parameters to identify the components of each source. Note that physical separation of the sources does not necessarily result in different spatial signatures. For omnidirectional microphones,

each attenuation-delay pair is associated with a circle (in 3-space) of possible physical locations. In the no-attenuation zero-delay case, the ambiguity circle becomes a plane. For directional microphones, the incidence of spatial ambiguities can be reduced [11]. We will thus assume that the sources have different spatial signatures,

$$(a_j \neq a_k) \text{ or } (\delta_j \neq \delta_k), \quad \forall j \neq k. \tag{8.12}$$

8.3 DUET

8.3.1 Main Observation and Outline

The assumptions of anechoic mixing and local stationarity allow us to rewrite the mixing equations (8.1) and (8.2) in the time–frequency domain as,

$$\begin{bmatrix} \hat{x}_1(\tau,\omega) \\ \hat{x}_2(\tau,\omega) \end{bmatrix} = \begin{bmatrix} 1 & \dots & 1 \\ a_1 e^{-i\omega\delta_1} & \dots & a_N e^{-i\omega\delta_N} \end{bmatrix} \begin{bmatrix} \hat{s}_1(\tau,\omega) \\ \vdots \\ \hat{s}_N(\tau,\omega) \end{bmatrix}. \tag{8.13}$$

With the further assumption of W-disjoint orthogonality, at most one source is active at every (τ,ω), and the mixing process can be described,

$$\text{for each } (\tau,\omega), \quad \begin{bmatrix} \hat{x}_1(\tau,\omega) \\ \hat{x}_2(\tau,\omega) \end{bmatrix} = \begin{bmatrix} 1 \\ a_j e^{-i\omega\delta_j} \end{bmatrix} \hat{s}_j(\tau,\omega) \quad \text{for some } j. \tag{8.14}$$

Of course, in the above equation, j depends on (τ,ω) in that j is the index of the source active at (τ,ω). The main observation that DUET leverages is that the ratio of the time–frequency representations of the mixtures does not depend on the source components but only on the mixing parameters associated with the active source component:

$$\forall (\tau,\omega) \in \Omega_j, \quad \frac{\hat{x}_2(\tau,\omega)}{\hat{x}_1(\tau,\omega)} = a_j e^{-i\omega\delta_j}, \tag{8.15}$$

where

$$\Omega_j := \{(\tau,\omega) : \hat{s}_j(\tau,\omega) \neq 0\}. \tag{8.16}$$

The mixing parameters associated with each time–frequency point can be calculated:

$$\tilde{a}(\tau,\omega) := \left|\hat{x}_2(\tau,\omega)/\hat{x}_1(\tau,\omega)\right|, \tag{8.17}$$

$$\tilde{\delta}(\tau,\omega) := (-1/\omega)\angle(\hat{x}_2(\tau,\omega)/\hat{x}_1(\tau,\omega)). \tag{8.18}$$

Under the assumption that the microphones are sufficiently close together so that the delay estimate is not incorrect due to wrap-around, the *local attenuation estimator* $\tilde{a}(\tau,\omega)$ and the *local delay estimator* $\tilde{\delta}(\tau,\omega)$ can only take on the values of the actual mixing parameters. Thus, the union of the $(\tilde{a}(\tau,\omega),\tilde{\delta}(\tau,\omega))$ pairs taken over all (τ,ω) is the set of mixing parameters (a_j,δ_j):

$$\bigcup_{(\tau,\omega)} \{(\tilde{a}(\tau,\omega),\tilde{\delta}(\tau,\omega))\} = \{(a_j,\delta_j) \,:\, j=1,\ldots,N\}. \tag{8.19}$$

So, we now know the set of mixing parameter pairs. As we saw in (8.7), we can demix via binary masking if we can determine the indicator function of each source. We can now determine the indicator functions via

$$M_j(\tau,\omega) = \begin{cases} 1 & (\tilde{a}(\tau,\omega),\tilde{\delta}(\tau,\omega)) = (a_j,\delta_j) \\ 0 & \text{otherwise} \end{cases} \tag{8.20}$$

and then demix using the masks.

In summary, the essentials to the DUET method are:

1. Construct the time–frequency representation of both mixtures.
2. Take the ratio of the two mixtures and extract local mixing parameter estimates.
3. Combine the set of local mixing parameter estimates into N pairings corresponding to the true mixing parameter pairings.
4. Generate one binary mask for each determined mixing parameter pair corresponding to the time–frequency points which yield that particular mixing parameter pair.
5. Demix the sources by multiplying each mask with one of the mixtures.
6. Return each demixed time–frequency representation to the time domain.

In practice, because not all of the assumptions are strictly satisfied, the local mixing parameter estimates will not be precisely the mixing parameters. In this case, we can replace the definition of Ω_j with

$$\Omega_j := \{(\tau,\omega) : |\hat{s}_j(\tau,\omega)| >> |\hat{s}_k(\tau,\omega)|, \quad \forall k \neq j\} \tag{8.21}$$

and then

$$\forall (\tau,\omega) \in \Omega_j, \quad \frac{\hat{x}_2(\tau,\omega)}{\hat{x}_1(\tau,\omega)} \approx a_j e^{-i\omega\delta_j} \tag{8.22}$$

and the estimates will cluster around the mixing parameters. These clusters will be sufficiently far apart to be identifiable as long as our spatially separate assumption is satisfied. The determination of these clusters is the topic of the next section.

8.3.2 Two-Dimensional Smoothed Weighted Histogram

In order to account for the fact that our assumptions made previously will not be satisfied in a strict sense, we need a mechanism for clustering the relative attenuation-delay estimates. In [12], we considered the maximum-likelihood (ML) estimators for a_j and δ_j in the following mixing model:

$$\begin{bmatrix} \hat{x}_1(\tau,\omega) \\ \hat{x}_2(\tau,\omega) \end{bmatrix} = \begin{bmatrix} 1 \\ a_j e^{-i\omega\delta_j} \end{bmatrix} \hat{s}_j(\tau,\omega) + \begin{bmatrix} \hat{n}_1(\tau,\omega) \\ \hat{n}_2(\tau,\omega) \end{bmatrix}, \quad \forall (\tau,\omega) \in \Omega_j, \tag{8.23}$$

where $\hat{n}_1$ and $\hat{n}_2$ are noise terms which represent the assumption inaccuracies. Rather than estimating a_j, we estimate

$$\alpha_j := a_j - \frac{1}{a_j}, \tag{8.24}$$

which we call the *symmetric attenuation* because it has the property that if the microphone signals are swapped, the attenuation is reflected symmetrically about a center point ($\alpha = 0$). That is, swapping the microphone signals changes α_j to $-\alpha_j$. Signals that are louder on microphone 1 will have $\alpha_j < 0$ and signals that are louder on microphone 2 will have $\alpha_j > 0$. In contrast, there is an inequity of treatment if we estimate a_j directly. Swapping the microphone signals changes a_j to $1/a_j$, and this results in signals louder on microphone 1 occupying $1 > a_j > 0$ and signals louder on microphone 2 occupying $\infty > a_j > 1$. We, therefore, define the local symmetric attenuation estimate,

$$\tilde{\alpha}(\tau,\omega) := \left| \frac{\hat{x}_2(\tau,\omega)}{\hat{x}_1(\tau,\omega)} \right| - \left| \frac{\hat{x}_1(\tau,\omega)}{\hat{x}_2(\tau,\omega)} \right|. \tag{8.25}$$

Motivated by the form of the ML estimators [12], the following pair of estimators emerge:

$$\tilde{\alpha}_j = \frac{\iint_{(\tau,\omega)\in\Omega_j} |\hat{x}_1(\tau,\omega)\hat{x}_2(\tau,\omega)|^p \, \omega^q \tilde{\alpha}(\tau,\omega) \mathrm{d}\tau \mathrm{d}\omega}{\iint_{(\tau,\omega)\in\Omega_j} |\hat{x}_1(\tau,\omega)\hat{x}_2(\tau,\omega)|^p \, \omega^q \mathrm{d}\tau \mathrm{d}\omega} \tag{8.26}$$

and

$$\tilde{\delta}_j = \frac{\iint_{(\tau,\omega)\in\Omega_j} |\hat{x}_1(\tau,\omega)\hat{x}_2(\tau,\omega)|^p \, \omega^q \tilde{\delta}(\tau,\omega) \mathrm{d}\tau \mathrm{d}\omega}{\iint_{(\tau,\omega)\in\Omega_j} |\hat{x}_1(\tau,\omega)\hat{x}_2(\tau,\omega)|^p \, \omega^q \mathrm{d}\tau \mathrm{d}\omega}, \tag{8.27}$$

which have been parameterized by p and q. Note that the form of each estimator is that of a weighted average of the local symmetric attenuation and local delay estimators with the weight for a given time–frequency point being $|\hat{x}_1(\tau,\omega)\hat{x}_2(\tau,\omega)|^p \, \omega^q$. Various choices for p and q are noteworthy:

- $p = 0$, $q = 0$: the counting histogram proposed in the original DUET algorithm [13]
- $p = 1$, $q = 0$: motivated by the ML symmetric attenuation estimator [12]
- $p = 1$, $q = 2$: motivated by the ML delay estimator [12]
- $p = 2$, $q = 0$: in order to reduce delay estimator bias [12]
- $p = 2$, $q = 2$: for low signal-to-noise ratio or speech mixtures [14]

The reason why the delay estimator gives more weight to higher frequencies is that the delay is calculated from a noisy phase estimate $\omega\delta + n$ by dividing the phase by ω. So in effect, the delay estimate is $\delta + n/\omega$ and the higher the frequency, the smaller the noise term. Our practical experience with DUET suggests that $p = 1$, $q = 0$ is a good default choice. When the sources are not equal power, we would suggest $p = 0.5$, $q = 0$ as it prevents the dominant source from hiding the smaller source peaks in the histogram.

Regardless of the choice of p and q, the difficulty with the estimators is that they require knowledge of the time–frequency supports of each source, but that is exactly what we are solving for. Based on the observation in the previous section that the local symmetric attenuation and delay estimates will cluster around the actual symmetric attenuation and delay mixing parameters of the original sources, we need a mechanism for determining these clusters. The estimators (8.26) and (8.27) suggest the construction of a two-dimensional weighted histogram to determine the clusters and the estimate mixing parameters (a_j, δ_j). The histogram is the key structure used for localization and separation. By using $(\tilde{\alpha}(\tau,\omega), \tilde{\delta}(\tau,\omega))$ pairs to indicate the indices into the histogram and using $|\hat{x}_1(\tau,\omega)\hat{x}_2(\tau,\omega)|^p\,\omega^q$ for the weight, clusters of weight will emerge centered on the actual mixing parameter pairs (a_j, δ_j). Assuming the (a_j, δ_j) pairs are reasonably separated from each other, smoothing the histogram with a window which is large enough to capture all the contributions of one source without capturing the contributions of multiple sources will result in N distinct peaks emerging with centers $(a_j, \delta_j), j = 1, \ldots, N$. Thus:

The weighted histogram separates and clusters the parameter estimates of each source. The number of peaks reveals the number of sources, and the peak locations reveal the associated source's anechoic mixing parameters.

We now formally define the two-dimensional weighted histogram by first defining the set of points which will contribute to a given location in the histogram,

$$I(\alpha,\delta) := \{(\tau,\omega) : |\tilde{\alpha}(\tau,\omega) - \alpha| < \Delta_\alpha, |\tilde{\delta}(\tau,\omega) - \delta| < \Delta_\delta\}, \tag{8.28}$$

where Δ_α and Δ_δ are the smoothing resolution widths. The two-dimensional smoothed weighted histogram is constructed,

$$H(\alpha,\delta) := \iint_{(\tau,\omega)\in I(\alpha,\delta)} |\hat{x}_1(\tau,\omega)\hat{x}_2(\tau,\omega)|^p\,\omega^q \mathrm{d}\tau\mathrm{d}\omega. \tag{8.29}$$

All the weight associated with time–frequency points yielding local estimates $(\tilde{\alpha}(\tau,\omega),\tilde{\delta}(\tau,\omega))$ within $(\Delta_\alpha,\Delta_\delta)$ of (α,δ) contributes to the histogram at (α,δ). If all the assumptions are satisfied, the histogram will have the form,

$$H(\alpha,\delta)=\begin{cases}\iint |\hat{s}_j(\tau,\omega)|^{2p}\,\omega^q \mathrm{d}\tau\mathrm{d}\omega & |\alpha_j-\alpha|<\Delta_\alpha, |\delta_j-\delta|<\Delta_\delta \\ 0 & \text{otherwise,}\end{cases} \tag{8.30}$$

and the N peaks corresponding to the N sources will be clearly visible.

A MATLAB® implementation of the histogram calculation is shown in Fig. 8.5 (see Appendix). While the presentation in this chapter is in continuous time, the end implementation is in discrete time. The conversion of the presented concepts into discrete time is straightforward. One slight difference in the implementation is that the delay calculation results in the unit of the delay being samples, which can easily be converted into seconds by multiplying by the sampling rate. It is convenient for the purposes of demonstration that the delay be in samples, so we will use this convention in the coded implementation. The code in Fig. 8.5 was run on two anechoic mixtures of five speech sources. The five 16 kHz speech signals were taken from the TIMIT database and looped so that each signal has length six seconds. The sum of the five signals was used as the first mixture. The second mixture contained the five sources with relative amplitudes $(1.1, 0.9, 1.0, 1.1, 0.9)$ and sample delays $(-2,-2,0,2,2)$. Figure 8.1 shows the five peaks associated with five original speech sources. The peak locations are exactly the mixing parameter pairs for each of the sources.

8.3.3 Separating the Sources

The mixing parameters can be extracted by locating the peaks in the histogram. We have investigated several different automatic peak enumeration/identification methods including weighted k-means, model-based peak removal, and peak tracking [15], but no single technique appears to be appropriate in all settings and often we resort to manual identification of the peaks. Once the peaks have been identified, our goal is to determine the time–frequency masks which will separate each source from the mixtures. This is a trivial task once the mixing parameters have been determined. We simply assign each time–frequency point to the peak location which is closest to the local parameter estimates extracted from the time–frequency point. In [12] we proposed using the likelihood function to produce a measure of closeness. Given histogram peak centers $(\tilde{\alpha}_j,\tilde{\delta}_j), j=1,\ldots,N$, we convert the symmetric attenuation back to attenuation via

$$\tilde{a}_j=\frac{\tilde{\alpha}_j+\sqrt{\tilde{\alpha}_j^2+4}}{2} \tag{8.31}$$

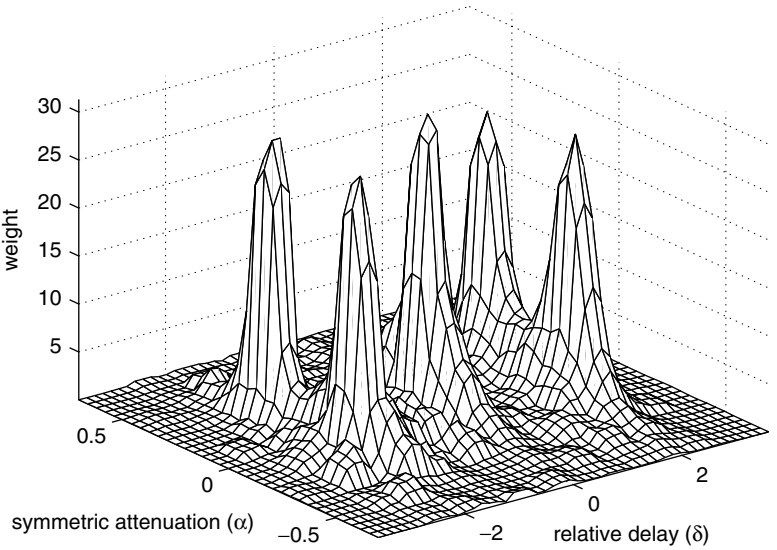

Fig. 8.1. DUET two-dimensional cross power weighted ($p = 1, q = 0$) histogram of symmetric attenuation ($a - 1/a$) and delay estimate pairs from two mixtures of five sources. Each peak corresponds to one source and the peak locations reveal the source mixing parameters.

assign a peak to each time–frequency point via

$$J(\tau,\omega) := \underset{k}{\operatorname{argmin}} \frac{\left|\tilde{a}_k e^{-i\tilde{\delta}_k\omega}\hat{x}_1(\tau,\omega) - \hat{x}_2(\tau,\omega)\right|^2}{1+\tilde{a}_k^2} \tag{8.32}$$

and then assign each time–frequency point to a mixing parameter estimate via

$$\tilde{M}_j(\tau,\omega) := \begin{cases} 1 & J(\tau,\omega) = j \\ 0 & \text{otherwise.} \end{cases} \tag{8.33}$$

Essentially, (8.32) and (8.33) assign each time–frequency point to the mixing parameter pair which best explains the mixtures at that particular time–frequency point. We demix via masking and ML combining [12],

$$\tilde{\hat{s}}_j(\tau,\omega) = \tilde{M}_j(\tau,\omega)\left(\frac{\hat{x}_1(\tau,\omega) + \tilde{a}_j e^{i\tilde{\delta}_j\omega}\hat{x}_2(\tau,\omega)}{1+\tilde{a}_j^2}\right). \tag{8.34}$$

We can then reconstruct the sources from their time–frequency representations by converting back into the time domain. We are now ready to summarize the DUET algorithm.

8.3.4 The DUET BSS Algorithm

1. Construct time–frequency representations $\hat{x}_1(\tau,\omega)$ and $\hat{x}_2(\tau,\omega)$ from mixtures $x_1(t)$ and $x_2(t)$
2. Calculate $\left(\left|\frac{\hat{x}_2(\tau,\omega)}{\hat{x}_1(\tau,\omega)}\right| - \left|\frac{\hat{x}_1(\tau,\omega)}{\hat{x}_2(\tau,\omega)}\right|, \frac{-1}{\omega}\angle\left(\frac{\hat{x}_2(\tau,\omega)}{\hat{x}_1(\tau,\omega)}\right)\right)$
3. Construct 2D smoothed weighted histogram $H(\alpha,\delta)$ as in (8.29)
4. Locate peaks and peak centers which determine the mixing parameter estimates
5. Construct time–frequency binary masks for each peak center $(\tilde{\alpha}_j, \tilde{\delta}_j)$ as in (8.33)
6. Apply each mask to the appropriately aligned mixtures as in (8.34)
7. Convert each estimated source time–frequency representation back into the time domain

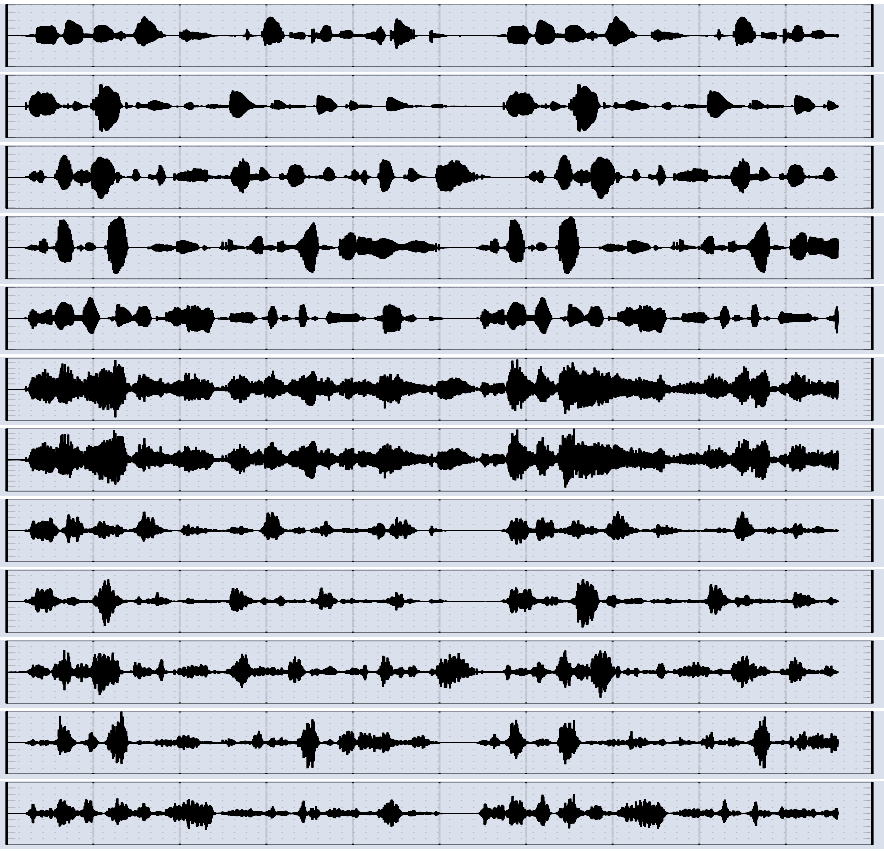

Fig. 8.2. Five original sources, two mixtures, and the five estimates of the original sources.

For the mixtures used to create the histogram in Fig. 8.1, the peaks were identified and the code presented in Fig. 8.8 was used to implement steps 5–7 in the algorithm. The original sources, mixtures, and demixed sources are presented in Fig. 8.2.

8.4 Big Delay DUET

The DUET technique presented in the previous section is limited to being able to estimate the mixing parameters and separate sources that arrive within an intra-mixture delay of less than $\frac{1}{2f_m}$ where f_m is the highest frequency of interest in the sources. This can be seen from the condition for the accurate determination of δ from $e^{i\omega_m\delta}$ requires $|\omega_m\delta| < \pi$. With $\omega_m = 2\pi f_m$, we arrive at $\delta < \frac{1}{2f_m}$. If we are sampling at the Nyquist rate ($2f_m$), then δ must be less than one sampling interval. If we are sampling at twice Nyquist, then δ must be less than two sampling intervals. Thus, DUET is applicable when the sensors are separated by at most $\frac{c}{2f_m}$ meters where c is the speed of propagation of the signals. For example, for voice mixtures where the highest frequency of interest is 4000 Hz and the speed of sound is 340 m/s, the microphones must be separated by less than 4.25 cm in order for DUET to be able to localize and separate the sources correctly for all frequencies of interest for all possible source locations. In some applications, microphones cannot be placed so closely together. In this section, we present two possible extensions to DUET that allow for arbitrary microphone spacing [16].

The first extension involves analyzing the phase difference between frequency adjacent time–frequency ratios to estimate the delay parameter. This technique increases the maximum possible separation between sensors from $\frac{1}{2f_m}$ to $\frac{1}{2\Delta f}$ where Δf is the frequency spacing between adjacent frequency bins in the time–frequency representation. As one can choose Δf, this effectively removes the sensor spacing constraint.

The second extension involves progressively delaying one mixture against the second and constructing a histogram for each delay. When the delaying of one mixture moves the intersensor delay of a source to less than $\frac{1}{2f_m}$, the delay estimates will align and a peak will emerge. When the inter-sensor delay of a source is larger than $\frac{1}{2f_m}$, the delay estimates will spread and no dominant peak will be visible. The histograms are then tiled to produce a histogram which covers a large range of possible delays and the true mixing parameter peaks should be dominant in this larger histogram.

We now describe each method in more detail.

8.4.1 Method One: Differential

The first method makes the additional assumption that,

$$|\hat{s}_j(\tau,\omega)| \approx |\hat{s}_j(\tau,\omega+\Delta\omega)|, \forall j, \forall\omega, \tau. \tag{8.35}$$

That is, the power in the time–frequency domain of each source is a smooth function of frequency. From before we have $\forall(\tau,\omega) \in \Omega_j$,

$$\begin{bmatrix} \hat{x}_1(\tau,\omega) \\ \hat{x}_2(\tau,\omega) \end{bmatrix} = \begin{bmatrix} 1 \\ a_j e^{-i\omega\delta_j} \end{bmatrix} \hat{s}_j(\tau,\omega), \tag{8.36}$$

and now, in addition, we have,

$$\begin{bmatrix} \hat{x}_1(\tau,\omega+\Delta\omega) \\ \hat{x}_2(\tau,\omega+\Delta\omega) \end{bmatrix} = \begin{bmatrix} 1 \\ a_j e^{-i(\omega+\Delta\omega)\delta_j} \end{bmatrix} \hat{s}_j(\tau,\omega+\Delta\omega). \tag{8.37}$$

Thus,

$$\left(\frac{\hat{x}_2(\tau,\omega)}{\hat{x}_1(\tau,\omega)}\right) \overline{\frac{\hat{x}_2(\tau,\omega+\Delta\omega)}{\hat{x}_1(\tau,\omega+\Delta\omega)}} = (a_j e^{-i\omega\delta_j})(a_j e^{i(\omega+\Delta\omega)\delta_j}) = a_j^2 e^{i\Delta\omega\delta_j}, \tag{8.38}$$

and the $|\omega\delta_j| < \pi$ constraint has been relaxed to $|\Delta\omega\delta_j| < \pi$. Note that $\Delta\omega$ is a parameter that can be made arbitrarily small by oversampling along the frequency axis. As the estimation of the delay from (8.38) is essentially the estimation of the derivative of a noisy function, results can be improved by averaging delay estimates over a local time–frequency region.

8.4.2 Method Two: Tiling

The second method constructs a number of histograms by iteratively delaying one mixture against the other. The histograms are appropriately overlapped corresponding to the delays used and summed to form one large histogram with the delay range of the summation histogram much larger than the delay range of the individual histograms. Specifically, consider the histogram created from the two mixtures where the second mixture has been delayed by β (that is, the second mixture is replaced with $x_2(t-\beta)$ in effect, changing each δ_j to $\delta_j + \beta$),

$$H_\beta(\alpha,\delta) := \begin{cases} \iint_{(\tau,\omega)\in I(\alpha,\delta)} |\hat{x}_1(\tau,\omega)\hat{x}_2(\tau,\omega)|^p \, \omega^q \mathrm{d}\tau\mathrm{d}\omega & \delta \in (-m_\delta, m_\delta) \\ 0 & \text{otherwise,} \end{cases} \tag{8.39}$$

where $2m_\delta$ is the delay width of the histogram. If we happen to choose β such that $|w_m(\delta_j+\beta)| < \pi$, then there will be no phase-wrap issue, all the delay estimates extracted will correspond to a delay of $\delta_j + \beta$, and a clear peak should emerge in the histogram for that source. After peak identification, we can subtract back out the shift by β for the delay estimate to determine the true δ_j.

We construct the summary tiled histogram T by appropriately overlapping and adding a series of local histograms H_β constructed for linearly spaced values of β,

$$T(\alpha,\delta) := \sum_{k=-K}^{K} H_{k\delta_m}(\alpha, \delta - k\delta_m), \quad \delta \in (-K\delta_m - m_\delta, K\delta_m + m_\delta). \tag{8.40}$$

The peaks that emerge in the overall histogram will correspond to the true delays. Demixing is accomplished using the standard DUET demixing by using the histogram tile that contains the source peak to be separated. As the interference from other sources will tend to be separated at zero delay, it is preferred to use a histogram tile where the peak is not centered at zero for separation.

Figure 8.3 shows a standard DUET histogram and a tiled DUET histogram for a five source mixing example. The same speech files were used as in Fig. 8.5, the only difference being that the sample delays used this time were $(-170, -100, 0, 50, 150)$. The standard DUET histogram fails to resolve any source mixing parameter pairs and simply has one peak at the origin. The tiled DUET histogram, on the other hand, has a clear peak located at each of the mixing parameter pairings. The differential method has similar performance to the tiled method. Thus, DUET can be extended to the case when large delays are present and the close microphone limitation has been eliminated.

8.5 Approximate W-Disjoint Orthogonality

In this section we discuss a quantitative measure of W-disjoint orthogonality as the W-disjoint orthogonality assumption is not strictly satisfied for our signals of interest. In order to measure to what degree the condition is approximately satisfied, we consider the following generalization, which has been discussed in [17–19]. We propose the normalized difference between the signal energy maintained in masking and the interference energy maintained in masking as a measure of the approximate disjoint orthogonality associated with a particular mask, $M(\tau,\omega)$, $0 \le M(\tau,\omega) \le 1$:

$$D_j(M) := \frac{\iint |M(\tau,\omega)\hat{s}_j(\tau,\omega)|^2 \,\mathrm{d}\tau\mathrm{d}\omega - \iint |M(\tau,\omega)\hat{y}_j(\tau,\omega)|^2 \,\mathrm{d}\tau\mathrm{d}\omega}{\iint |\hat{s}_j(\tau,\omega)|^2 \,\mathrm{d}\tau\mathrm{d}\omega}, \tag{8.41}$$

where

$$\hat{y}_j(\tau,\omega) := \sum_{\substack{k=1 \\ k\neq j}}^{N} \hat{s}_k(\tau,\omega) \tag{8.42}$$

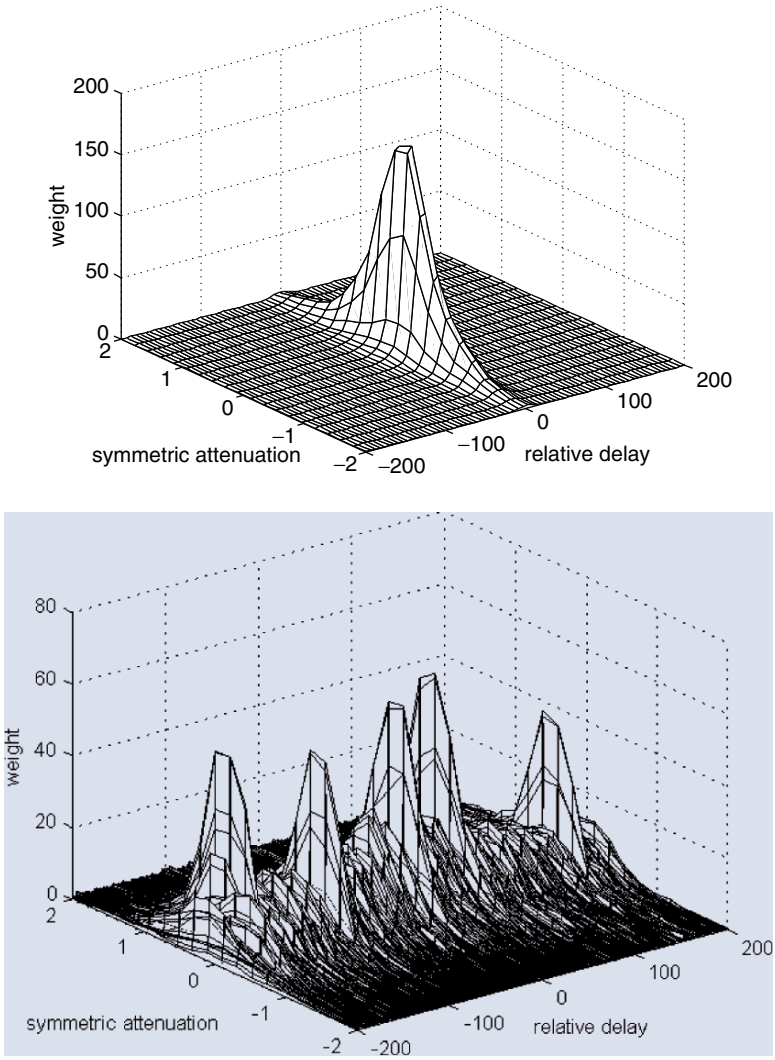

Fig. 8.3. Comparison of standard DUET (top) and tiled DUET (bottom) in a five source mixing example with large relative delays: $(-170, -100, 0, 50, 150)$ samples. Standard DUET fails when the delays are large, but the extension, tiled DUET, succeeds in identifying the number of sources and their associated mixing parameters.

so that $\hat{y}_j(\tau, \omega)$ is the summation of the sources interfering with the jth source. $D_j(M)$ combines two important performance criteria: (1) how well a demixing mask preserves the source of interest, and (2) how well a demixing mask suppresses the interfering sources. It is easily shown that $D_j(M)$ is bounded by $-\infty$ and 1. If it is desirable that the W-disjoint orthogonality measure be bounded between 0 and 1, then we suggest the following

mapping:

$$d_j(M) := 2^{D_j(M)-1}, \tag{8.43}$$

which has the properties:

1. $d_j(M) = 1$ implies that s_j is W-disjoint orthogonal with all interfering signals,
2. $d_j(M) = 1/2$ implies that application of mask M results in a demixture with equal source of interest and interference energies, and
3. $d_j(M) \approx 0$ implies that the mask M results in a demixture dominated by interference energy.

Now that we have a quantitative measure which describes the demixing performance of masking, we ask the question which mask should we use in order to maximize (8.43). It follows from (8.41) that

$$M_j^*(\tau,\omega) := \begin{cases} 1 & |\hat{s}_j(\tau,\omega)| > |\hat{y}_j(\tau,\omega)| \\ 0 & |\hat{s}_j(\tau,\omega)| \leq |\hat{y}_j(\tau,\omega)| \end{cases} \tag{8.44}$$

maximizes $d_j(M)$ as it turns 'on' signal coefficients where the source of interest dominates the interference and turns 'off' the remaining coefficients. The terms of equal magnitude in (8.44) we have arbitrarily turned 'off', but including them or excluding them makes no difference to the W-disjoint orthogonal measure as the terms cancel. The mask M_j^* is the optimal mask for demixing from a W-disjoint orthogonal performance standpoint. In order to determine the level of approximate W-disjoint orthogonality in a given mixture, we construct the optimal masks as in (8.44) and examine $d_j(M)$ for each source. We perform this analysis for mixtures of simultaneous speech in the next section. We use the results to determine the appropriate window size for use in the DUET algorithm.

8.5.1 Approximate W-Disjoint Orthogonality of Speech

In Fig. 8.4 we measure the W-disjoint orthogonality for pairwise (and 3-way, 4-way, up to 8-way) mixing as a function of window size. For the tests, N speech files, sampled at 16 kHz, were selected at random from the TIMIT database. One signal was randomly selected to be the target and the remaining $N-1$ signals were summed to form an interference signal. Both the target signal and interference signal were then transformed into the time–frequency domain using a Hamming window of size $\{2^0, 2^1, \ldots, 2^{15}\}$. The magnitude of the coefficients of a target source was compared to the sum of the remaining sources to generate the mask M_j^*. Using the mask, $d_{M_j^*}$ was calculated. Over 300 mixtures were generated and the results averaged to form each data point shown in the figure. In all three cases the Hamming window of size 1024 produced the representation that was the most W-disjoint orthogonal. A similar

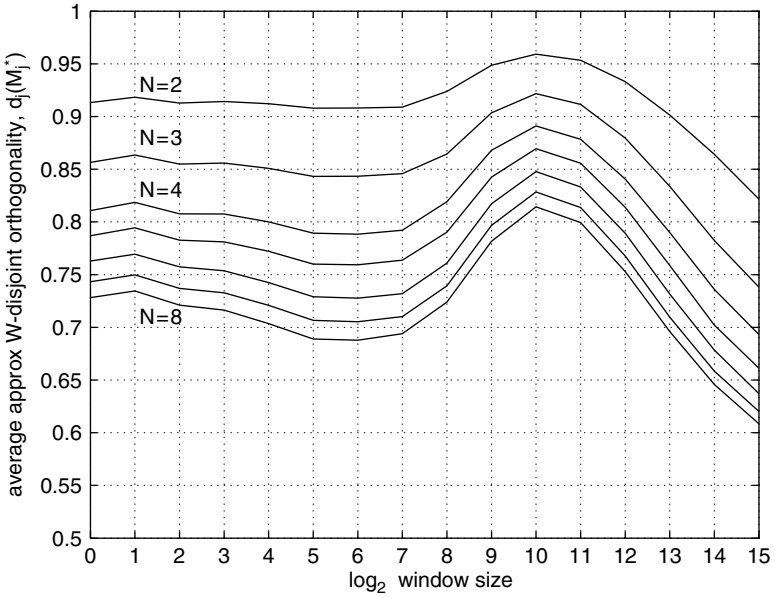

Fig. 8.4. W-disjoint orthogonality for time–frequency representations of mixtures of $N = 2, 3, \ldots, 8$ speech sources as a function of window size used in the time–frequency transformation. Speech is most W-disjoint orthogonal when a window of 1024 samples is used, corresponding to 64 ms length.

conclusion regarding the optimal time–frequency resolution of a window for speech separation was arrived at in [20]. Note that even when the window size is 1 (i.e., time domain), the mixtures still exhibit a high level of W-disjoint orthogonality. This fact was exploited by those methods that used the time-disjoint nature of speech [9, 21–23]. Figure 8.4 clearly shows the advantage of moving from the time domain to the time–frequency domain: the speech signals are more disjoint in the time–frequency domain provided the window size is sufficiently large. Choosing the window size too large, however, results in reduced W-disjoint orthogonality.

8.6 Conclusions

In this chapter we presented a tutorial overview of the DUET blind source separation method. DUET assumes that the source signals are disjoint in the time–frequency domain (W-disjoint orthogonal) and exploits the fact that the ratio of the time–frequency representations of the mixtures can be used to partition the mixtures into the original sources. The key construct in DUET is the two-dimensional smoothed weighted histogram which is used

to cluster the mixing parameter estimates. By assuming an anechoic mixing model, all time–frequency points provide input to the histogram as we can eliminate the frequency variable and extract the delay. The fact that both estimation and separation can be done when the number of sources is larger than the number of mixtures without significant computational complexity, as is demonstrated by the MATLAB® code in Sect. 8.6, is testimony to the usefulness of the technique.

The extraction of the delay relies on the assumption that the microphone must be close together, which limits the environments in which DUET can be applied. In Sect. 8.4 we presented two possible extensions to DUET which remove this limitation and demonstrated that the extended DUET can estimate and demix sources regardless of the microphone separation.

Additionally, in Sect. 8.5 we verified that speech signals satisfy W-disjoint orthogonality enough to allow for mixing parameter estimation and source separation. Our own experience with DUET over the past eight years shows that multiple speakers talking simultaneously can be demixed with two microphones with high fidelity of recovered signals using DUET.

References

1. A. Cichocki and S. Amari, *Adaptive Blind Signal and Image Processing*. Wiley, 2002.
2. A. Bell and T. Sejnowski, "An information-maximization approach to blind separation and blind deconvolution," *Neural Computation*, vol. 7, pp. 1129–1159, 1995.
3. J. Cardoso, "Blind signal separation: Statistical principles," *Proceedings of IEEE, Special Issue on Blind System Identification and Estimation*, pp. 2009–2025, Oct. 1998.
4. E. Weinstein, M. Feder, and A. Oppenheim, "Multi-channel signal separation by decorrelation," *IEEE Trans. on Speech and Audio Processing*, vol. 1, no. 4, pp. 405–413, Oct. 1993.
5. L. Parra and C. Spence, "Convolutive blind source separation of non-stationary sources," *IEEE Transactions on Speech and Audio Processing*, pp. 320–327, May 2000.
6. H. Broman, U. Lindgren, H. Sahlin, and P. Stoica, "Source separation: A TITO system identification approach," *Signal Processing*, vol. 73, pp. 169–183, 1999.
7. R. Balan, A. Jourjine, and J. Rosca, "A particular case of the singular multivariate AR identification and BSS problems," in *1st International Conference on Independent Component Analysis*, Assuis, France, 1999.
8. P. Comon, "Blind channel identification and extraction of more sources than sensors," in *SPIE Conference*, San Diego, July 19–24 1998, pp. 2–13.
9. M. V. Hulle, "Clustering approach to square and non-square blind source separation," in *IEEE Workshop on Neural Networks for Signal Processing (NNSP)*, Madison, Wisconsin, Aug. 23–25 1999, pp. 315–323.
10. H. Krim and M. Viberg, "Two Decades of Array Signal Processing Research, The Parametric Approach," *IEEE Signal Processing Magazine*, pp. 67–94, July 1996.

11. B. Coleman and S. Rickard, "Cardioid microphones and DUET," in *IEE Irish Signals and Systems Conference (ISSC2004)*, July 2004, pp. 264–269.
12. O. Yilmaz and S. Rickard, "Blind separation of speech mixtures via time–frequency masking," *IEEE Transactions on Signal Processing*, vol. 52, no. 7, pp. 1830–1847, July 2004.
13. A. Jourjine, S. Rickard, and O. Yılmaz, "Blind Separation of Disjoint Orthogonal Signals: Demixing N Sources from 2 Mixtures," in *Proc. ICASSP2000, June 5–9, 2000, Istanbul, Turkey*, June 2000.
14. T. Melia, "Underdetermined blind source separation in echoic environments using linear arrays and sparse representtions," Ph.D. dissertation, University College Dublin, Dublin, Ireland, Mar. 2007.
15. S. Rickard, R. Balan, and J. Rosca, "Real-time time–frequency based blind source separation," in *3rd International Conference on Independent Component Analysis and Blind Source Separation (ICA2001)*, Dec. 2001.
16. S. Rickard and R. Balan, "Method for estimating mixing parameters and separating multiple sources from signal mixtures," Dec. 2003, US Patent Application no. 20030233227.
17. S. Rickard and O. Yilmaz, "On the approximate W-disjoint orthogonality of speech," in *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Orlando, Florida, USA, May 2002, pp. 529–532.
18. O. Yilmaz and S. Rickard, "Blind separation of speech mixtures via time–frequency masking," *IEEE Transactions on Signal Processing*, vol. 52, no. 7, pp. 1830–1847, July 2004.
19. S. Rickard, "Sparse sources are separated sources," in *14th European Signal Processing Conference (EUSIPCO)*, Sept. 2006.
20. M. Aoki, M. Okamoto, S. Aoki, H. Matsui, T. Sakurai, and Y. Kaneda, "Sound source segregation based on estimating incident angle of each frequency component of input signals acquired by multiple microphones," *Acoustical Science and Technology*, vol. 22, no. 2, pp. 149–157, 2001.
21. J.-K. Lin, D. G. Grier, and J. D. Cowan, "Feature extraction approach to blind source separation," in *IEEE Workshop on Neural Networks for Signal Processing (NNSP)*, Amelia Island Plantation, Florida, Sept. 24–26 1997, pp. 398–405.
22. T.-W. Lee, M. Lewicki, M. Girolami, and T. Sejnowski, "Blind source separation of more sources than mixtures using overcomplete representations," *IEEE Signal Processing Letters*, vol. 6, no. 4, pp. 87–90, Apr. 1999.
23. L. Vielva, D. Erdogmus, C. Pantaleon, I. Santamaria, J. Pereda, and J. C. Principe, "Underdetermined blind source separation in a time-varying environment," in *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, vol. 3, Orlando, Florida, USA, May 13–17 2002, pp. 3049–3052.

Appendix: MATLAB functions

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% 1. analyze the signals - STFT
wlen = 1024; timestep = 512; numfreq = 1024;
awin = hamming(wlen); % analysis window is a Hamming window

tf1 = tfanalysis(x1,awin,timestep,numfreq); % time-freq domain
tf2 = tfanalysis(x2,awin,timestep,numfreq); % time-freq domain
tf1(1,:)=[]; tf2(1,:)=[]; % remove dc component from mixtures
% to avoid dividing by zero frequency in the delay estimation

% calculate pos/neg frequencies for later use in delay calc
freq = [(1:numfreq/2) ((-numfreq/2)+1:-1)]*(2*pi/(numfreq));
fmat = freq(ones(size(tf1,2),1),:)';
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% 2. calculate alpha and delta for each t-f point
R21 = (tf2+eps)./(tf1+eps); % time-freq ratio of the mixtures

%%% 2.1 HERE WE ESTIMATE THE RELATIVE ATTENUATION (alpha) %%%
a = abs(R21); % relative attenuation between the two mixtures
alpha = a - 1./a; % 'alpha' (symmetric attenuation)
%%% 2.2 HERE WE ESTIMATE THE RELATIVE DELAY (delta) %%%%
delta = -imag(log(R21))./fmat; % 'delta' relative delay
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% 3. calculate weighted histogram
p = 1; q = 0; % powers used to weight histogram
tfweight = (abs(tf1).*abs(tf2)).^p .* abs(fmat).^q; % weights
maxa = 0.7; maxd = 3.6; % hist boundaries
abins = 35; dbins = 50; % number of hist bins
% only consider time-freq points yielding estimates in bounds
amask=(abs(alpha)<maxa)&(abs(delta)<maxd);
alpha_vec = alpha(amask);
delta_vec = delta(amask);
tfweight = tfweight(amask);
% determine histogram indices
alpha_ind = round(1+(abins-1)*(alpha_vec+maxa)/(2*maxa));
delta_ind = round(1+(dbins-1)*(delta_vec+maxd)/(2*maxd));

% FULL-SPARSE TRICK TO CREATE 2D WEIGHTED HISTOGRAM
A = full(sparse(alpha_ind,delta_ind,tfweight,abins,dbins));
% smooth the histogram - local average 3-by-3 neighboring bins
A = twoDsmooth(A,3);
% plot 2-D histogram
mesh(linspace(-maxd,maxd,dbins),linspace(-maxa,maxa,abins),A);
```

Fig. 8.5. MATLAB® code which constructs the two-dimensional weighted histogram.

```
function tfmat = tfanalysis(x,awin,timestep,numfreq)
% time-frequency analysis
% X is the time domain signal
% AWIN is an analysis window
% TIMESTEP is the # of samples between adjacent time windows.
% NUMFREQ is the # of frequency components per time point.
%
% TFMAT complex matrix time-freq representation

x = x(:); awin = awin(:); % make inputs go columnwise

nsamp = length(x); wlen = length(awin);

% calc size and init output t-f matrix
numtime = ceil((nsamp-wlen+1)/timestep);
tfmat = zeros(numfreq,numtime+1);

for i = 1:numtime
    sind = ((i-1)*timestep)+1;
    tfmat(:,i) = fft(x(sind:(sind+wlen-1)).*awin,numfreq);
end
i = i+1;
sind = ((i-1)*timestep)+1;
lasts = min(sind,length(x));
laste = min((sind+wlen-1),length(x));
tfmat(:,end) = fft([x(lasts:laste) ;
                    zeros(wlen-(laste-lasts+1),1)].*awin,numfreq);
```

Fig. 8.6. The *tfanalysis.m* function.

```
function smat = twoDsmooth(mat,ker)
%TWO2SMOOTH - Smooth 2D matrix.
%
% smat = twoDsmooth(mat,ker)
%
% MAT is the 2D matrix to be smoothed.
% KER is either
%  (1) a scalar, in which case a ker-by-ker matrix of
%      1/ker^2 is used as the matrix averaging kernel
%  (2) a matrix which is used as the averaging kernel.
%
% SMAT is the smoothed matrix (same size as mat).

if prod(size(ker))==1, % if ker is a scalar
    kmat = ones(ker,ker)/ker^2;
else
    kmat = ker;
end

% make kmat have odd dimensions
[kr kc] = size(kmat); if rem(kr,2) == 0,
    kmat = conv2(kmat,ones(2,1))/2;
    kr = kr + 1;
end if rem(kc,2) == 0,
    kmat = conv2(kmat,ones(1,2))/2;
    kc = kc + 1;
end

[mr mc] = size(mat);
fkr = floor(kr/2); % number of rows to copy on top and bottom
fkc = floor(kc/2); % number of columns to copy on either side
smat = conv2(...
    [mat(1,1)*ones(fkr,fkc) ones(fkr,1)*mat(1,:) ...
    mat(1,mc)*ones(fkr,fkc);
    mat(:,1)*ones(1,fkc) mat mat(:,mc)*ones(1,fkc)
    mat(mr,1)*ones(fkr,fkc) ones(fkr,1)*mat(mr,:) ...
    mat(mr,mc)*ones(fkr,fkc)],...
    flipud(fliplr(kmat)),'valid');
```

Fig. 8.7. The *twoDsmooth.m* function.

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% 4. peak centers (determined from histogram)
numsources = 5;
peakdelta = [-2 -2 0 2 2];
peakalpha = [.19 -.21 0 .19 -.21];

% convert alpha to a
peaka = (peakalpha+sqrt(peakalpha.^2+4))/2;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% 5. determine masks for separation
bestsofar=Inf*ones(size(tf1));
bestind=zeros(size(tf1));
for i = 1:length(peakalpha)
    score = abs(peaka(i)*exp(-sqrt(-1)*fmat*peakdelta(i)) ...
        .*tf1-tf2).^2/(1+peaka(i)^2);
    mask = (score < bestsofar);
    bestind(mask) = i;
    bestsofar(mask) = score(mask);
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% 6. & 7. demix with ML alignment and convert to time domain
est = zeros(numsources,length(x1)); % demixtures
for i=1:numsources
    mask = (bestind==i);
    esti = tfsynthesis([zeros(1,size(tf1,2)) ;
      ((tf1+peaka(i)*exp(sqrt(-1)*fmat*peakdelta(i)).*tf2)...
        ./(1+peaka(i)^2)).*mask], ...
        sqrt(2)*awin/1024, timestep, numfreq);
    est(i,:) = esti(1:length(x1))';
    % add back into the demix a little bit of the mixture
    % as that eliminates most of the masking artifacts
    soundsc(est(i,:)+0.05*x1,fs); pause; % play demixture
end
```

Fig. 8.8. MATLAB® code for separating the sources from the mixtures, given estimates of the mixing parameters.

```
function x = tfsynthesis(timefreqmat,swin,timestep,numfreq)
% time-frequency synthesis
% TIMEFREQMAT is the complex matrix time-freq representation
% SWIN is the synthesis window
% TIMESTEP is the # of samples between adjacent time windows.
% NUMFREQ is the # of frequency components per time point.
%
% X contains the reconstructed signal.

swin = swin(:); % make synthesis window go columnwise

winlen = length(swin);
[numfreq numtime] = size(timefreqmat);
ind = rem((1:winlen)-1, numfreq) +1;
x = zeros((numtime-1)*timestep + winlen,1);

for i = 1:numtime % overlap, window, and add
    temp = numfreq*real(ifft(timefreqmat(:,i)));
    sind = ((i-1)*timestep);
    rind = (sind+1):(sind+winlen);
    x(rind) = x(rind) + temp(ind).*swin;
end
```

Fig. 8.9. The *tfsynthesis.m* function.