A Probabilistic Graphical Model for Microphone Array Source Separation using Rich Pre-Trained Source Models

H.T. Attias

Golden Metallic, Inc, P.O. Box 475608, San Francisco, CA 94147, USA

Abstract

Voice based computing applications, such as phone communication and speech recognition, use microphone arrays to capture voice from a human speaker. In many environments of interest, however, sounds from other sources interfere with the speaker's voice, posing severe problems for subsequent processing. This paper describes a new framework for treating this problem, and presents and demonstrates a new algorithm for the cancellation of interfering sounds. Our framework combines techniques from statistical machine learning with ideas from speech and audio processing. An important feature involves training rich probabilistic models on data from different types of relevant sound sources. Those source models are then incorporated into a larger probabilistic model of the observed microphone data. Using that model we derive our algorithm, which is of the expectation-maximization type and infers from data the clean sound of separate individual sources. We report very good results on data recorded in different environments.

Index Terms: source separation, array processing, speech enhancement, machine learning, probabilistic models.

1. Introduction

Voice interfaces for computing devices are quickly becoming a hot topic. One reason is the tremendous popularity of cell phones and PDAs. A voice interface would allow people to control the phone by speaking to it, replacing (or complementing) the small, inconvenient keyboard. This is particularly relevant in Asian markets, where demand for personal computing and internet access is increasing fast, and many prefer to use cell phones over PCs due to their lower price. Another reason is the growing trend of using voice over internet protocol for phone and conference calls. A third reason is the proliferation of computing devices that allow people to perform specific tasks, e.g., obtain driving directions or control a smart home or office (e.g., change TV channels, turn lights on/off). A voice interface could make using such devices more efficient.

However, for voice interfaces to cross the chasm into mainstream use, they must operate reliably in everyday environments like a car, restaurant, airport lounge, living room, or hotel lobby. Such environments contain loud ambient sounds that interfere with the voice of the person speaking. Those sounds significantly reduce voice intelligibility in communication systems. They also confuse speech recognition engines and sharply degrade their accuracy.

Several approaches to this problem have been developed. On the hardware side, they include sound capturing devices such as directional microphones and microphone arrays. On the software (algorithmic) side, they include array based noise suppression techniques, such as beamforming and blind source separation (see, e.g., [1]-[7]).

This paper presents a new solution to this problem. It is a software solution, which requires only low-cost, off-the-shelf microphones, although its performance can be improved with high quality microphones. Our solution is based on a new approach to source separation, which considerably extends previous work on the subject (see, e.g., [3]-[7]). This approach combines speech and audio processing ideas with techniques from statistical machine learning, particularly probabilistic graphical models [8]. An important feature of our approach is the use of rich probabilistic models of different types of sounds, that may be present in relevant environments. Those models are trained offline on appropriate data samples. Following training, they are incorporated as sound source models into a larger probabilistic graphical model of the observed microphone data. That model describes the data as arising from unobserved sound sources, that have been combined by an unknown convolutive mixing transformation. Applying probabilistic inference techniques, we derive from the model an expectation-maximization (EM) algorithm which infers from data the signals of the unobserved sources. Voice interfaces can apply this algorithm to extract a clean version of the human speaker's voice from the microphone data, and feed it into a voice communication, speech recognition, or another voice based applications.

This paper is organized as follows. Section 2 provides a mathematical definition of the source separation task. Section 3 defines the probabilistic model used to described sound sources, and presents an algorithm that infers the model parameters from data. Section 4 describes the probabilistic model of the observed microphone data, which includes the sound source models as building blocks. It also outlines the derivation of our EM algorithm for source separation. Section 5 describes experiments. Section 6 discusses extensions of this work.

2. Problem Formulation

This paper focuses on the scenario where the number of sources of interest equals the number of sensors, and the background noise is vanishingly small. This condition is known by the technical term 'square, zero-noise convolutive mixing'. Whereas our algorithm may produce satisfactory results under other conditions, its performance would in general be suboptimal.

Let L denote the number of sensors, and let y_{in} denote the signal waveform captured by sensor i at time $n = 0, 1, 2, ...$, where $i = 1 : L$. Let x_{in} denote the signal emitted by source i at time n. Then $y_{in} = \sum_{jm} H_{ijm} x_{jn-m}$. The filters H_{ijm} model the convolutive mixing transformation. To achieve source separation, the algorithm must infer the individual source signals x_{in}, which are unobserved, from the sensor signals. For this purpose we seek an unmixing transformation G_{ijm} such that $x_{in} = \sum_{jm} G_{ijm} y_{jn-m}$.

Rather than working with signal waveforms in the time domain, it turns out to be more computationally efficient, as well as mathematically convenient, to work with signal frames in the frequency domain. Frames are obtained by applying windowed DFT to the waveform. Let $X_{im}[k]$ denote the frames of source i. They are computed by multiplying the waveform x_{in} by an N-point window w_n at J-point shifts, $X_{im}[k] = \sum_{n=0}^{N-1} e^{-i\omega_k n} w_n x_{i,Jm+n}$, where $m = 0 : M-1$ is the frame index and $k = 0 : N-1$ is the frequency index. The number of frames M is determined by the waveform's length and the window shift. The sensor frames $Y_{im}[k]$ are computed from y_{in} in the same manner.

In the frequency domain, the task is to infer from sensor data an unmixing transformation $G_{ij}[k]$ for each frequency k, such that $X_{im}[k] = \sum_j G_{ij}[k] Y_{jm}[k]$. In vector notation we have

$$X_m[k] = G[k] Y_m[k] \ , \tag{1}$$

where $X_m[k], Y_m[k]$ are complex $L \times 1$ vectors and $G[k]$ is a complex $L \times L$ matrix. Once the algorithm infers the source frames from the sensor frames via (1), their time domain waveforms x_n are synthesized by an overlap-and-add procedure.

Notation. We often use a collective notation obtained by dropping the frequency index k from the frames. X_{im} denotes the set of $X_{im}[k]$ values at all frequencies, and X_m denotes the set of $L \times 1$ vectors $X_m[k]$ at all frequencies. Also, we define a Gaussian distribution with parameters μ, ν over a complex variable Z by $\mathcal{N}(Z \mid \mu, \nu) = \frac{\nu}{\pi} e^{-\nu |Z-\mu|^2}$. Two moments are $EZ = \mu$ and $E \mid Z \mid^2 = 1/\nu$, hence μ is termed the mean of Z and ν is termed the precision. This is a joint distribution over the real and imaginary parts of Z. A Gaussian over a real variable z is defined as usual by $\mathcal{N}(z \mid \mu, \nu) = \sqrt{\frac{\nu}{2\pi}} e^{-0.5\nu(z-\mu)^2}$.

3. Source Models

Our algorithm employs parametric probabilistic models for different types of source signals. This section describes the source model, and presents an algorithm for inferring the model parameters from clean sound samples of a given source.

3.1. Source model definition

We describe a source signal by a probabilistic mixture model over its frames. The model for source i has S_i components,

$$p(X_{im}) = \sum_{s=1}^{S_i} p(X_{im} \mid S_{im} = s) p(S_{im} = s) \ . \tag{2}$$

Here we assume that the frames are mutually independent, hence

$$p(X_{i,m=0:M-1}) = \prod_m p(X_{im}) \ . \tag{3}$$

It is straightforward to relax this assumption and use, e.g., a hidden Markov model.

We model each component by a zero-mean Gaussian factorized over frequencies, where component s has precision $\nu_{is}[k]$ at frequency k, and prior probability π_{is},

$$\begin{aligned} p(X_{im} \mid S_{im} = s) &= \prod_{k=0}^{N/2} \mathcal{N}(X_{im}[k] \mid 0, \nu_{is}[k]) \\ p(S_{im} = s) &= \pi_{is} \ . \end{aligned} \tag{4}$$

It is sufficient to consider $k = 0 : N/2$ since $X_{im}[N-k] = X_{im}[k]^\star$. Notice that the precisions $\nu_{is}[k]$ form the inverse spectrum of component s, since the spectrum is the second moment $E(\mid X_{im}[k] \mid^2 \mid S_{im} = s) = 1/\nu_{is}[k]$, and the first moment vanishes. The inverse-spectra and prior probabilities, collectively denoted by $\theta_i = \{\nu_{is}[k], \pi_{is} \mid s = 1 : S_i, k = 0 : N/2\}$, constitute the parameters of source i.

3.2. An algorithm for learning source model parameters

This section describes a maximum likelihood (ML) algorithm for inferring the model parameters θ_i for source i from sample data X_{im}. Generally, ML infers parameter values by maximizing the observed data likelihood $\mathcal{L}_i = \sum_m \log p(X_{im})$ w.r.t. the parameters. In our case, however, we have a hidden variable model, since the source states S_{im} are not directly observable. As is standard procedure, we use an expectation-maximization (EM) algorithm, which is an iterative technique for ML in hidden variable models. An appealing way to derive it, following the article by Neal & Hinton in [8], starts by considering the objective function

$$\mathcal{F}_i(\bar{\pi}_i, \theta_i) = \sum_{m=0}^{M-1} \sum_{s=1}^{S_i} \bar{\pi}_{ism} \left[\log p(X_{im}, S_{im=s}) - \log \bar{\pi}_{ism} \right] \tag{5}$$

which depends on the parameters θ_i, as well as on $\bar{\pi}_i$ which denotes collectively the posterior distributions over the states of source i. Each posterior $\bar{\pi}_{ism}$ is the probability that source i is in state $S_{im} = s$ at time m, conditioned on the frame X_{im}.

Each EM iteration maximizes $\mathcal{F}_i$ alternately w.r.t. to the parameters and the posteriors. The E-step maximizes $\mathcal{F}_i$ w.r.t. to the state posteriors by the update rule

$$\begin{aligned} \bar{\pi}_{ism} &= p(S_{im} = s \mid X_{im}) \\ &= \frac{p(X_{im}, S_{im} = s)}{\sum_{s'=1:S_i} p(X_{im}, S_{im} = s')} \ , \end{aligned} \tag{6}$$

keeping constant the current values of the parameters (note that the r.h.s. depends on θ_i). The M-step maximizes $\mathcal{F}_i$ w.r.t. the model parameters by the update rule

$$\begin{aligned} \nu_{is}[k]^{-1} &= \frac{\sum_{m=0}^{M-1} \bar{\pi}_{ism} \mid X_{im}[k] \mid^2}{\sum_{m=0}^{M-1} \bar{\pi}_{ism}} \\ \pi_{is} &= \frac{1}{M} \sum_{m=0}^{M-1} \bar{\pi}_{ism} \ , \end{aligned}$$

keeping constant the current values of the posteriors. To prove the convergence of this procedure, we use the fact that $\mathcal{F}_i$ is upper bounded by the likelihood,

$$\mathcal{F}_i(\bar{\pi}_i, \theta_i) \le \mathcal{L}_i(\theta_i) = \sum_{m=0}^{M-1} \log p(X_{im}) \ , \tag{7}$$

where equality is obtained when $\bar{\pi}_i$ is set according to (6), with the posterior being computed using θ_i. We use $\mathcal{F}_i$ as a convergence criterion, and stop the EM iteration when the change in $\mathcal{F}_i$

is below than a pre-determined threshold. One may also define a convergence criterion using the change in the model parameters in addition to, or instead of, the change in $\mathcal{F}_i$.

Since in typical scenarios one uses a DFT length N between a few 100's and a few 1000's, depending on the sampling rate and the mixing complexity, a direct application of the algorithm above would be attempting to perform maximization in a parameter space θ_i of a very high dimension. Hence, a good initialization procedure is critical. We use the following procedure. (1) Compute the cepstra, defined as the DFT of the log-spectra $\log \mid X_{im}[k] \mid^2$, and keep only the low $N' \ll N$ cepstral coefficients. (2) Perform vector quantization on the low cepstra, and obtain the cluster means $\xi_{is}[n]$. (3) Initialize the state spectra to $\nu_{is}[k] = \exp[-\frac{1}{N}(\xi_{is}[0] + 2\sum_{n=0}^{N'-1}\cos(\omega_n k)\xi_{is}[n])]$. Following initialization, we iterate the EM algorithm to convergence. This procedure produced very good performance. We also tested two algorithms that maximize $\mathcal{F}_i$ directly w.r.t. the low cepstrals $\xi_{is}[n]$, but we omit them here.

4. Separation Algorithm

This section presents an EM algorithm for inferring the unmixing transformation $G[k]$ from sensor frames $Y_m[k]$. It assumes that the model parameters θ_i for all sources $i = 1 : L$ are given.

4.1. Sensor model

Since the source frames and the sensor frames are related by (1), we have

$$p(Y_m) = \prod_{k=0}^{N/2} \mid G[k] \mid^2 p(X_m) \ , \qquad (8)$$

except for $k = 0, N/2$ where, since $X_m[k], Y_m[k]$ are real, we must use $\mid G[k] \mid$ instead of its square. As above, the sources are mutually independent, $p(X_m) = \prod_{i=1}^{L} p(X_{im})$, where $p(X_{im})$ is given by (4). The sensor likelihood is therefore given by

$$\begin{aligned}\mathcal{L}(G) &= \sum_{m=0}^{M-1} \log p(Y_m) \qquad (9)\\ &= M\sum_{k=0}^{N/2}\log \mid G[k] \mid^2 + \sum_{m=0}^{M-1}\sum_{i=1}^{L}\log p(X_{im}) \ ,\end{aligned}$$

where $X_m[k] = G[k]Y_m[k]$. Inferring the unmixing transformation is done by maximizing this likelihood w.r.t. G.

4.2. An algorithm for learning the unmixing transformation

Like the source signals, the sensor signals are also described by a hidden variable model, since the states S_{im} are unobserved. Here we derive an EM algorithm for ML estimation of G. Consider the objective function

$$\mathcal{F}(\bar{\pi}_{1:L}, G) = M\sum_{k=0}^{N/2}\log \mid G[k] \mid^2 + \sum_{i=1}^{L}\mathcal{F}_i(\bar{\pi}_i, \theta_i, G) \qquad (10)$$

where $\mathcal{F}_i$ is given by (5); we have added G as an argument since $\mathcal{F}_i$ depends on it via X_i (1). Each EM iteration maximizes $\mathcal{F}$ alternately w.r.t. the unmixing G and the posteriors $\bar{\pi}_i$, where π_{ism} is the probability that source i is in state S_{im} = at time m, as before, except now this probability is conditioned on the sensor frame Y_m. The parameters $\theta_{1:L}$ are held fixed.

The E-step maximizes $\mathcal{F}$ w.r.t. the state posteriors $\bar{\pi}_{ism}$, keeping constant the current values of G. It uses an update rule that is formally identical to (6), except now the X_{im} are given by $X_m[k] = G[k]Y_m[k]$. The M-step maximizes $\mathcal{F}$ w.r.t. the unmixing transformation G.

Before presenting the update rule, we rewrite $\mathcal{F}$ as follows. Let $C^i[k]$ denote the ith weighted correlation of the sensor frames at frequency k. It is a Hermitian $L \times L$ matrix defined by

$$C^i_{jj'}[k] = \frac{1}{M}\sum_{m=0}^{M-1} \bar{\nu}_{im}[k] Y_{jm}[k] Y^{\star}_{j'm}[k] \qquad (11)$$

where the weight for C^i is given by the precisions of source i's states, averaged w.r.t. their posterior, $\bar{\nu}_{im}[k] = \sum_{s=1}^{S_i}\bar{\pi}_{ism}\nu_{is}[k]$. $\mathcal{F}$ of (10) is now given by $\mathcal{F} = M\log \mid G[k] \mid^2 - M\sum_{i=1}^{L}(G[k]C^i[k]G[k]^\dagger)_{ii} + f$ where f is independent of G.

Computing G can generally be done efficiently by an iterative method, based on the concept of the relative (a.k.a. natural) gradient. Consider the ordinary gradient $\partial\mathcal{F}/\partial G_{ij}[k] = 2(G[k]^{\dagger\ -1} - G[k]C^i[k])_{ij}$. To maximize $\mathcal{F}$, we increment $G[k]$ by an amount proportional to $(\partial\mathcal{F}/\partial G[k])G[k]^\dagger G[k]$. We obtain

$$G_{ij}[k] \to G_{ij}[k] + \epsilon(G[k] - G[k]C^i[k]G[k]^\dagger G[k])_{ij} \qquad (12)$$

where ϵ is the adaptation rate. For each M-step, the update rule (12) is iterated to convergence. Alternatively, one may stop short of convergence and move on to the E-step of the next iteration, as this would still result in increasing $\mathcal{F}$.

4.2.1. The case of two sensors

The special case of $L = 2$ sensors is by far the most common one in practical applications. Incidentally, in this case there exists an M-step solution for G which is even more efficient than the iterative procedure of (12). This is because the M-step maximization of $\mathcal{F}$ for $L = 2$ can be performed analytically. This section describes the solution.

At a maximum of $\mathcal{F}$ its gradient vanishes, hence the matrix $G[k]$ satisfies

$$(G[k]C^i[k]G[k]^\dagger)_{ij} = \delta_{ij} \ . \qquad (13)$$

Let us write $G[k]$ as a product of a diagonal matrix $U[k]$ and a matrix $V[k]$ with ones on its diagonal,

$$\begin{aligned} G[k] &= U[k]V[k] \qquad (14)\\ U[k] &= \begin{pmatrix} u_1[k] & 0 \\ 0 & u_2[k]\end{pmatrix} , \quad V[k] = \begin{pmatrix} 1 & v_1[k] \\ v_2[k] & 1\end{pmatrix} .\end{aligned}$$

With these definitions, the zero-gradient condition leads to the following equations, for all i, j,

$$\begin{aligned} (V[k]C^i[k]V[k]^\dagger)_{i\neq j} &= 0 \\ \mid u_i[k] \mid^2 (V[k]C^i[k]V[k]^\dagger)_{ii} &= 1 \ . \qquad (15)\end{aligned}$$

We now turn to the case $L = 2$, where all matrices are 2×2. The first line in (15) then implies that v_1 depends linearly on v_2 and v_2 satisfies the quadratic equation $av_2^2 + bv_2 + c = 0$. Hence

$$v_1 = \frac{(av_2 + d)^\star}{c} \ , \qquad v_2 = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a} \ , \qquad (16)$$

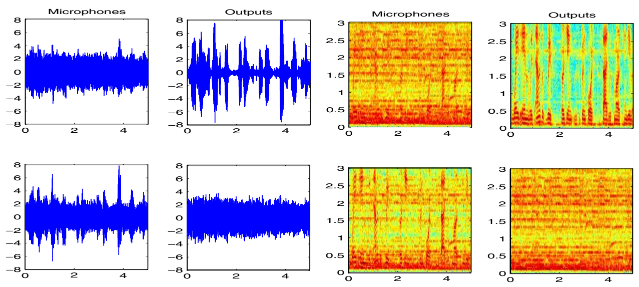

Figure 1: Results on 5sec overlapping sound data captured by 2 microphones in an office environment. Sources include a male English speaker and a radio playing music. From left: 1st column: microphone waveforms. 2nd column: waveforms of output sounds, with the speaker's voice on top (notice the choppiness of the waveform characteristic of speech). 3rd column; microphone spectrograms (up to 3kHz). 4th column: spectrograms of output sounds, with the voice on top (harmonics are visible).

where the k-dependence is omitted. The second line in (15) identifies the u_i within a phase, reflecting the identifiability properties of G. Constraining them to be real nonnegative, we obtain

$$\begin{aligned} u_1 &= (\alpha_1 + 2\mathrm{Re}\beta_1^\star v_1 + \gamma_1 \mid v_1 \mid^2)^{-1/2} \\ u_2 &= (\gamma_2 + 2\mathrm{Re}\beta_2 v_2 + \alpha_2 \mid v_2 \mid^2)^{-1/2} \; . \end{aligned} \tag{17}$$

The quantities $\alpha_i[k], \beta_i[k], \gamma_i[k]$ denote the elements of the weighted correlation matrices (11) for each frequency k,

$$C^i[k] = \begin{pmatrix} \alpha_i[k] & \beta_i[k] \\ \beta_i^\star[k] & \gamma_i[k] \end{pmatrix} \; , \qquad i = 1, 2 \tag{18}$$

where $\alpha_i[k], \gamma_i[k]$ are real nonnegative and $\beta_i[k]$ is complex. The coefficients $a[k], b[k], c[k], d[k]$ are given by

$$\begin{aligned} a &= \alpha_1\beta_2 - \alpha_2\beta_1 \; , & b &= \alpha_1\gamma_2 - \alpha_2\gamma_1 + d \\ c &= \beta_1^\star\gamma_2 - \beta_2^\star\gamma_1 \; , & d &= 2i\mathrm{Im}\beta_1^\star\beta_2 \; . \end{aligned} \tag{19}$$

Hence, the result of the M-step for the case $L = 2$ is the unmixing transformation G of (14), obtained using Eqs. (11,16–19).

5. Experiments

We have tested the algorithm in 4 different environments using two scenarios, (1) two human speakers, (2) a human speaker and a rock music source. The human speakers included 2 males and 2 females. We used CD players, connected to speakers, as sound sources. The sources were placed $20 - 200$cm away from the microphone, at a relative angle of $90 - 180$deg. Background noise of various types (fan, hard disk, movement) was present at a low level. An inexpensive stereo microphone was used to capture sounds, with the microphones approximately 2in apart. In each scenario, we recorded 5 different 10sec long data segments with both sources simultaneously active. The sound level was tuned such that both sources had approximately the same energy (SIR$\approx$ 0). To facilitate SIR computation, we repeated each recording with only one source active at a given time.

Separately, we collected additional data from the same sound sources for model training. We captured a 30sec, 1-microphone sound data from each human speaker and a 120sec segment of rock music. Those data were used to train 2 source models using the algorithm of Section 3: a model for a human speaker (gender independent), and another model for music.

Next, we applied the algorithm of Section 4 to the stereo recordings. We used 8kHz sampling rate, $N = 2048$ DFT length, and a Hanning window with a 1024-point overlap. For scenario 1 we used the human speaker model for both sources, and for scenario 2 we used that model for one source, and the music model for the other. Convergence was defined by a relative change $< 10^{-2}$ in $\mathcal{F}$. Fig. 1 shows a typical example. We compared the performance of our algorithm to 2 published algorithms, termed Benchmark 1 [4] and Benchmark 2 [6]. SIR results (averaged over recordings) are shown in Table 1. Our algorithm outperformed both benchmarks. Performance on the first scenario was better, probably due to the fact that the energy in speech signals is less distributed among frequencies than in music.

Table 1: *Performance of the separation algorithm (dB)*

	New algorithm	Benchmark 1	Benchmark 2
Scenario 1	21.2	16.7	14.5
Scenario 2	18.4	15.9	13.1

We have repeated these experiments, replacing one CD player in scenario 1 by actual human speakers. Whereas it was difficult to compute the SIR accurately, the separation results were audibly similar to those in the experiments above.

6. Extensions

We are currently extending this algorithm in several directions. One extension performs sequential processing could adapt online to changes in the acoustic environment, e.g., source movements. Another extension identifies automatically, from data, which source models should be used in a given environment, and retrieves them from a directory of stored models. We have also developed an extension that handles cases with different numbers of sources and sensors and a significant background/sensor noise level.

7. References

[1] S Haykin (2001). *Adaptive Filter Theory (4th Ed.)*. Prentice Hall.
[2] JG Proakis et al. (2002). *Algorithms for Statistical Signal Processing*. Prentice Hall.
[3] TW Lee, AJ Bell, R Lambert (1997). Blind separation of convolved and delayed sources. *Adv. NIPS* 9, 758-764.
[4] H Attias, CE Schreiner (1998). Blind source separation and deconvolution: the dynamic component analysis algorithm. *Neural Computation* 10, 1373-1424.
[5] A Acero, S Altschuler, L Wu (2000). Speech/noise separation using two microphones and a VQ model of speech signals. *Proc. ICSLP 2000*, 4 532-535.
[6] L Parra, C Spence (2000). Convolutive blind source separation of non-stationary sources. *IEEE Trans. on Speech and Audio Processing* 8, 320-327.
[7] H Attias (2003). New EM algorithms for source separation and deconvolution. *Proc. ICASSP 2003.*
[8] MI Jordan (Ed.) (1998). *Learning in Graphical Models*. MIT Press.