Blind Separation of Convolutive Mixtures in the Time Domain - Separation of Speech Signals

Zbyněk Koldovský[1,2] and Petr Tichavský[1]

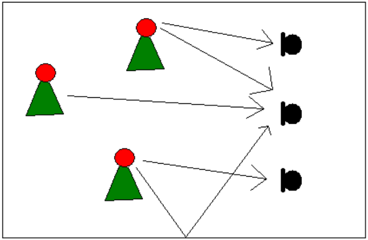
[1]*Institute of Information Theory and Automation,*
Academy of Sciences of the Czech Republic
[2]*Technical University of Liberec, Faculty of Mechatronic, Informatics, and Interdisciplinary Studies*, Liberec

Abstract

- We present a novel time-domain method for blind separation of convolutive mixture of audio sources.
- The method allows efficient separation using short data segments only.
- In practice, we are able to separate 2-4 speakers from audio recording of the length less than 6000 samples, which is less than 1 second in the 8 kHz sampling.
- The average time needed to process the data with filter of the length 20 was 2.2 seconds in Matlab v. 7.2 on an ordinary PC with 3GHz processor.

The Cocktail-Party Problem

Convolutive mixture: $$x_i(n) = \sum_{j=1}^{d} \sum_{\tau=0}^{M_{ij}} h_{ij}(\tau) s_j(n-\tau) \quad i = 1, \ldots, m$$



$s_j(n)$	...	original speakers' signals
$x_i(n)$	...	signals at microphones
$h_{ij}(\tau)$	...	impulse responses
M_{ij}	...	length of $h_{ij}(\tau)$

The goal: blind estimation of the original signals.

Blind Audio Source Separation via ICA

- Frequency-domain approach:

$$x_i(n) = \sum_{j=1}^{d} \sum_{\tau=0}^{M_{ij}} h_{ij}(\tau) s_j(n-\tau) \overset{\text{Fourier transf.}}{\longleftrightarrow} x_i(\omega) = \sum_{j=1}^{d} h_{ij}(\omega) s_j(\omega)$$

  $\Longrightarrow$ a set of instantaneous mixtures $\mathbf{x}(\omega) = \mathbf{H}(\omega)\mathbf{s}(\omega) \Longrightarrow$ application of complex ICA at each $\omega \Longrightarrow$ the so-called *permutation problem* due to indeterminacy of order of original frequency components

- Time-domain approach: Searching for independent components of the subspace spanned by

$$\begin{aligned}\mathbf{x}(n) = [&x_1(n), x_1(n-1), \ldots, x_1(n-L+1),\\ &x_2(n), x_2(n-1), \ldots, x_2(n-L+1), \ldots\\ &\ldots, x_m(n) \ldots, x_m(n-L+1)]^T\end{aligned}$$

Time-Domain Separation Procedure

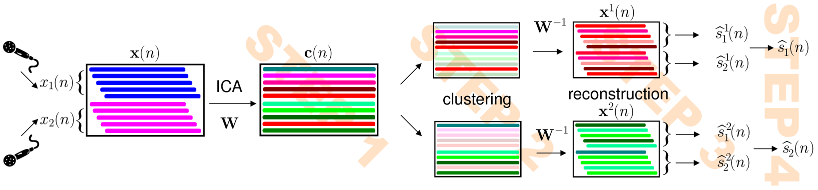
1. ICA decomposition of the whole subspace spanned by $\mathbf{x}(n)$ by means of an appropriate ICA algorithm $\longrightarrow$ results in a de-mixing transform $\mathbf{W}$
2. Grouping of independent components $\mathbf{c}(n) = \mathbf{W}\mathbf{x}(n)$ into clusters so that components in a cluster belong to the same original audio source
3. Reconstruction of original sources at microphones. For the jth cluster of components:
   1. Reconstruct the recorded signals $\mathbf{x}(n)$ by

      $$\mathbf{x}^j(n) = \mathbf{W}^{-1} \cdot \mathrm{diag}[\lambda_1, \ldots, \lambda_{mL}] \cdot \mathbf{W} \cdot \mathbf{x}(n),$$

      where $\lambda_1, \ldots, \lambda_{mL}$ are appropriately selected weights preferring components of the jth cluster (*fuzzy reconstruction*)
   2. Reconstruct the jth source at ith microphone as

      $$\widehat{s}_i^j(n) = \sum_{p=1}^{L} \mathbf{x}^j_{(i-1)L+p}(n+p-1)$$

Chart of the Method



STEP 1: ICA decomposition

Generally, ICA methods are expected to produce components $\mathbf{c}(n)$ in that the inter-sources interferences is cancelled as much as possible. We consider two different approaches.

- EFICA based on non-Gaussianity of the original sources $\longrightarrow$ in an ideal case produces components that are delayed innovations of the original sources
- BGL based on non-stationarity using approximate joint diagonalization of covariance matrices from different blocks $\longrightarrow$ in an ideal case produces clusters of components having the same dynamics

STEP 2: Grouping of the Components

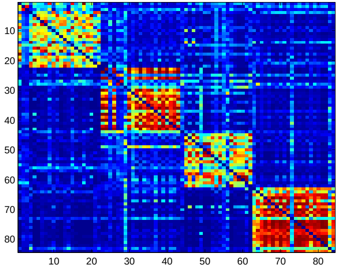
- Grouping can be done via clustering
- The distance between the i-th and $j-$th component can be measured as

$$D_{ij} = \hat{\mathrm{E}}[\mathbf{P}_i c_j(n)]^2$$

$\mathbf{P}_i \ldots$ projector on subspace spanned by

$$[c_i(n-L), \ldots, c_i(n+L)]$$

- We have used standard agglomerative hierarchical clustering.



STEP 3: Reconstruction

Reconstructed signals from the jth cluster are

$$\mathbf{x}^j(n) = \mathbf{W}^{-1} \cdot \mathrm{diag}[\lambda_1, \ldots, \lambda_{mL}] \cdot \mathbf{W} \cdot \mathbf{x}(n)$$

- a *hard* reconstruction:

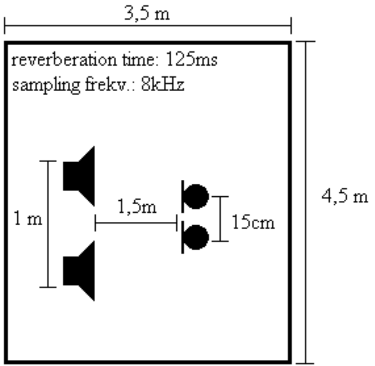
$$\lambda_k = \begin{cases} 1 & \text{the } k\text{th component belongs to the } j\text{th cluster} \\ 0 & \text{otherwise} \end{cases}$$

- a *fuzzy* reconstruction: uses the clustered matrix of distances

$$\lambda_k = \left(\frac{\sum_{i \in K_j, i \neq k} D_{ki}}{\sum_{i \notin K_j, i \neq \ell} D_{ki}} \right)^{\alpha},$$

$K_j \ldots$ indices of components in the ith cluster
$\alpha \ldots$ an adjustable positive parameter controlling "hardness" of the weighting

Experimental Setup



- Two sources played over loudspeakers in an ordinary room and recorded by two microphones
- Length of recordings: 18000, length of data used for ICA $K = 6000$, sampling frequency: 8kHz
- BSS_EVAL Toolbox used for evaluation of performance of algorithms

Results

algorithm	presented		Parra, Spence		Sawada et al.	
filter length L	20		128		400	
average comp. time (secs)	2.2		9.1		3.3	
	SIR	**SDR**	**SIR**	**SDR**	**SIR**	**SDR**
man's voice #1	17.49	11.56	6.16	4.64	10.68	5.7
man's voice #2	15.65	11.81	5.44	1.38	13.16	6.75
man's voice	20.62	13.43	9.79	2.97	8.57	3.87
woman's voice	7.43	4.53	6.97	4.12	10.56	4.9
man's voice	18.79	10.27	8.45	4.76	18.8	5.75
Gaussian noise	17.68	13.61	11.34	8.65	17.22	11.69
man's voice	18.09	10.7	7.82	2.69	18.83	5.8
typewriter	23.5	17.31	11.97	9.50	19.21	13.71

Conclusions

- We present a novel time-domain method for blind separation of convolutive mixture of audio sources.
- The method allows efficient separation using short data segments only.
- In practice, we are able to separate 2-4 speakers from audio recording of the length less than 6000 samples, which is less than 1 second in the 8 kHz sampling.
- The average time needed to process the data with filter of the length 20 was 2.2 seconds in Matlab v. 7.2 on an ordinary PC with 3GHz processor.
- Since the separating mechanism can be kept frozen for certain time, our future work will be to modify the algorithm for on-line signal processing.