Improving speed with orthogonal beamforming

Ennes Sarradj

Brandenburg University of Technology Cottbus

Berlin Beamforming Conference 2010
Airfoil trailing edge noise
Setup in open jet aeroacoustic wind tunnel

Porous airfoils

- experimental survey using 56ch phased array
- 17+ different airfoils
- $U$: 25...50 m/s
- $\alpha$: -16...24°
- $\approx$ 3500 measurements
- fast method for absolute level determination needed!
Airfoil trailing edge noise
Porous airfoils: some results

How does it work?

Phased array beamforming (frequency domain)

- uses information from $N$ microphone signals
- $N \times N$ cross spectral matrix (CSM)
- $M$ sources (wanted+unwanted, $N > M$)

$\rightarrow$ CSM has $M$ non-zero eigenvalues
Eigendecomposition

CSM eigenvalues and eigenvectors

- contain all information about the sources
- \( M \) eigenvalues / eigenvectors \( \longrightarrow \) \( M \) sources
- practical complication: "noise" in the signals

\( \longrightarrow \) CSM has full rank \((N > M)\) eigenvalues

- two groups:
  - "large" eigenvalues (eigenvectors span signal subspace)
  - "small" eigenvalues (eigenvectors span noise subspace)

Hypothesis

- signal subspace eigenvalues map to sources
Example: Airfoil
2 kHz octave band

- conventional delay & sum beamforming
- CSM resynthesised from eigenvalue / eigenvector pairs
- seems to work, but mapping is only *approximate*
- spatial resolution not improved
Improving resolution

Beamformer as spatial filter

A source signal passes the filter without attenuation
B all other signals are attenuated as much as possible

Source location

▶ single source: maximum in map = source location

▶ multiple sources: ??? (maxima do not need to be sources)
▶ eigendecomposition may help!
Orthogonal beamforming

Location and strength from eigendecomposition

CSM eigen-decomposition

multiplication by eigenvalues

\[ \lambda_1 \cdot \lambda_2 \cdot \lambda_3 \]
Algorithm (simplified)

- for each frequency:
  - compute cross spectral matrix (CSM, $N \times N$)
  - compute eigendecomposition ($\lambda_i, v_i$)
  - estimate number of sources $M$
  - for each $i$ in $(1, \ldots, M)$:
    - compute beamforming map from resynthesised CSM
    - store location of map maximum
    - store eigenvalue $\lambda_i$ as source strength
  - new map: accumulate all strengths at stored locations
Generic test case

Four loudspeakers

- four "identical" tweeters
- narrow spacing (10 cm)
- 56ch array, aperture (150 cm)
- distance 72 cm
- uncorrelated noise signals:
  - case I: "identical" amplitude
  - case II: 0, -6, -12, -18 dB
Case I: identical amplitudes
Maps for 2 kHz and 15 kHz frequency line

- conventional delay & sum (CB)
- DAMAS (5000 iterations)
- orthogonal beamforming (OB) with $M=20$ and with $M=6$
Case I: identical amplitudes

Spectrum

- integration over loudspeaker sectors A, B, C, D
- B, C, D shifted by -10 dB, -20 dB, -30 dB respectively
Case II: different amplitudes
Maps for 2 kHz and 15 kHz frequency line

- conventional delay & sum (CB)
- DAMAS (5000 iterations)
- orthogonal beamforming (OB) with $M=20$ and with $M=6$
Case II: different amplitudes

Spectrum

- integration over loudspeaker sectors A, B, C, D
- B, C, D shifted by -10 dB, -20 dB, -30 dB respectively
Case II: additional noise

**Spectrum**

- **no noise**
- **+ noise (SNR 3 dB)**

- **additional noise in each microphone channel (-3 dB)**
Errors

From test case:

- good performance for high frequencies and different source strengths
- low frequencies: imprecise localisation
- same source strengths: errors in source strength estimation

Theory

- mapping assumption (signal subspace eigenvalues - sources) is approximate
- theoretical error bounds depend on difference of source strengths (Gershgorin Circle Theorem 1931)
- practical error bounds from Monte Carlo simulation
Errors: Monte Carlo simulation

- 4 loudspeakers at random positions
- Statistics from 10,000 runs
- Small error in many relevant cases
Practical test case
Airfoil trailing edge noise - setup

- **airfoil**
- **nozzle**
- **flow**
- **core jet**
- **mixing region**
- **TE source region**
- **microphone positions at \( z = 0.68 \) m

- **\( x, m \)**
- **\( y, m \)**

- **-0.5**
- **-0.268**
- **0**
- **0.5**

- **-0.2**
- **-0.1**
- **0**
- **0.1**
- **0.2**
Practical test case
Airfoil trailing edge noise - results

- medium and high frequencies:
  - good agreement with theory
  - performance comparable to DAMAS/CLEAN-SC
  - better than integration of maps from delay and sum (CB)
Improving \textbf{speed} ... ?

\textbf{Delay and Sum}

\begin{itemize}
  \item vector-matrix-vector multiplication ($\hat{G} = \text{CSM}$)
  \end{itemize}

\begin{equation*}
B(x_t) = h^H(x_t)\hat{G}h(x_t)
\end{equation*}

\begin{itemize}
  \item $4(N^2 + N)$ flop per grid point
\end{itemize}

\textbf{Orthogonal beamforming}

\begin{itemize}
  \item vector-vector multiplication ($\hat{G}_i = \lambda_i v_i v_i^H$, resynthesised CSM)
  \end{itemize}

\begin{equation*}
B_i(x_t) = h^H(x_t)\hat{G}_i h(x_t) = \lambda_i |h^H(x_t)v_i|^2, \quad i = 1 \ldots M
\end{equation*}

\begin{itemize}
  \item $(4N + 1)M$ flop per grid point (+ maximum finding)
  \item can be faster than delay and sum!
  \item easily parallelisable (one thread per eigenvalue)
\end{itemize}
Speed
Practical results

Loudspeaker example

- $N = 56$ (microphones)
- grid size $41 \times 41 = 1681$
- 2048 frequency bins
- time (incl. steering vector calculation)
  - CB: 190 s
  - OB ($M = 6$): 119 s
  - OB ($M = 20$): 185 s
Orthogonal beamforming - conclusions

Orthogonal beamforming

- signal subspace method
- suppresses noise effectively
- very fast: parallelisable, can be used for huge grids

Determination of absolute levels

- results comparable to deconvolution methods for medium and high frequencies
- works with minor sources (-20 dB)
- theoretical errors bounds established

Speed

- very fast, feasible for huge grids
- parallelisable

- full mathematical details
- error bounds
- detailed example results
and now for something completely different ...
3D beamforming

- beamforming maps usually on planar grids (2D)
- reality is 3D! (at least ;-) 
- problem: 3D grids are huge (e.g. 50×50×50=125000 points)
- solution: fast method
- problem: planar arrays – bad 3D resolution
- solution: deconvolution or similar method
3D beamforming
Four loudspeakers - 4 kHz octave band

delay & sum

orthogonal beamforming
3D beamforming

Airfoil - 4 kHz octave band - orthogonal beamforming