Sensor Arrays
Today’s lecture

- Sensor arrays
- Beamforming and localization
- Array-based source separation
A question

- Why do you have two ears? (or eyes)
Sensor arrays

- Multiple sensors allow us to do more
  - They reveal the “spatial dimension”

- We can thus do things involving space
  - Focus on one direction only
  - Locate the emanating source of a signal
One more question

• Why does your ear have that strange shape on the outside?
Collecting from multiple sources

- A parabolic sensor
  - Focuses right ahead and ignores other directions
Known for a while
Discretizing the parabolic dish

- Emulating that process with a small number of discrete sensors
That’s been known for a while too
A more flexible setup

- Use adjustable delays for each sensor
- We can now "steer" the array
The delay and sum beamformer

- Adjust sensor delays to focus on a source coming from a specific direction

\[ y(t) = \sum \tau_i \ast x_i(t) \]

- The delays depend on the array layout
  - We need to know the layout a priori
Uniform Linear Array (ULA)

- Far field assumption
  - “Wavefront is flat”

\[ \tau_i(t) = \begin{cases} 
1 & t = \frac{d \cos \phi}{C} \\
0 & \text{otherwise} 
\end{cases} \]
Some problems

- Direction confusion
  \[ \delta t = \frac{d \cos \phi}{C} = \frac{d \cos(-\phi)}{C} \]

- Need at least \( D + 1 \) sensors for \( D \) dimensions
- But also the more the better!
The array response

- Spatial aliasing = ambiguities in high frequencies
- Sensor distance must be less than half wavelength of highest freq

![Diagram showing array response with spatial aliasing](image)
Many more things ...

- Sensors might have non-uniform response
  - In direction or frequency

- Adaptive filters instead of delays
  - Suppress certain areas, boost others

- Rich literature in radio, sonar and audio
Extreme beamforming demo
Localization

- Determine delays that align with input
  - If you know the delays you can determine the angle of most likely incidence
Localization of a single source

• Cross-correlate sensor inputs

\[ c_{i,j}(t) = x_i(t) * x_j(-t) \]

• Peak denotes relative delay between inputs
  • Convert delays to angle

\[ \phi_{i,j} = \arcsin \frac{\tau_{i,j} C}{d_{i,j}} \]

• But there’s a problem
Localization of multiple sources

- If we have multiple sources estimation of one delay is not feasible

- Instead we can measure the energy from each location in the space we operate
  - Steer the array beam and measure signal

- This provides a spatial energy map
Multi-source demo

Situation video

Energy over angle and time
Arrays are everywhere
One more

USS San Francisco – after hitting an underwater mountain in 2005
Using more machine learning

- Most array literature is DSP-based
  - Beamforming has its limitations

- Source separation and blind methods
  - Machine learning applications on arrays
A “simple” audio problem

foo!

bar!
Formalizing the problem

- Each mic receives a mix of both sounds
  - Sound waves superimpose linearly
  - Ignoring propagation delays for now

- The simplified mixing model is:
  \[ x(t) = A \cdot s(t) \]

- How do we solve this system and find the original signals \( s(t) \)?

\[
\begin{align*}
  x_1(t) &= a_{11}s_1(t) + a_{21}s_2(t) \\
  x_2(t) &= a_{21}s_1(t) + a_{22}s_2(t)
\end{align*}
\]
When can we solve this?

- The mixing equation is:
  \[ x(t) = A \cdot s(t) \]

- Our estimates of \( s(t) \) will be:
  \[ \hat{s}(t) = A^{-1} \cdot x(t) \]

- To recover \( s(t) \), \( A \) must be invertible:
  - We need as many mics as sources
  - The mics/sources must not coincide
  - All sources must be audible

- Otherwise this is a different story ...
A simple example

- A simple invertible problem

\[
x(t) = A \cdot s(t)
\]

- \(s(t)\) contains two structured waveforms
- \(A\) is invertible (but we don’t know it’s values)
- \(x(t)\) looks messy, doesn’t reveal \(s(t)\) clearly

- Known as the Blind Source Separation problem (BSS)
  - Blind because we don’t see a lot of the information
What to look for

• We can only use $x(t)$

$$x(t) = A \cdot s(t)$$

• Is there a property we can take advantage of?
  • Yes! We know that different sounds are “different”

• The plan: Find a solution that enforces “differentness”
A first try

- Find $s(t)$ by minimizing cross-correlation
- Our estimate of $s(t)$ is computed by: $\hat{s}(t) = W \cdot x(t)$
  - If $W \approx A^{-1}$ then we have a good solution
- The goal is that the output becomes uncorrelated:
  $$\langle \hat{s}_i(t) \cdot \hat{s}_j(t) \rangle = 0, \forall i \neq j$$
  - We assume here that our signals are zero mean
- So the problem to solve is: $\arg \min_W \left\{ \sum_k w_{ik} x_k(t) \cdot \sum_k w_{jk} x_k(t) \right\}, \forall i \neq j$
• This is PCA!

\[
\begin{bmatrix}
\hat{s}(1), & \cdots, & \hat{s}(T)
\end{bmatrix} = W \cdot \begin{bmatrix}
x(1), & \cdots, & x(T)
\end{bmatrix} \Rightarrow
\]

\[
\Rightarrow \hat{S} = W \cdot X
\]

• Easy to compute with a simple decomposition:

\[
W = \text{eig} \left( X \cdot X^T \right)
\]
So how well does this work?

Mixing process:

\[ \mathbf{x}(t) \]

Unmixing process:

\[ \hat{\mathbf{s}}(t) \]

\[ \mathbf{A} = \begin{bmatrix} 2 & 1 \\ 1 & 1 \end{bmatrix}. \]

\[ \mathbf{W} = \begin{bmatrix} -0.6 & -0.4 \\ -2.8 & 3.7 \end{bmatrix}. \]

Well, that was a waste of time ...
What went wrong?

\[
\begin{align*}
\mathbf{x}(t) &= \begin{bmatrix}
2 & 1 \\
1 & 1
\end{bmatrix}.
\end{align*}
\]
PCA does the wrong thing

What are the mixture eigenvectors?

Scaling/rotating to get decorrelation
What’s wrong?

- Our data is not Gaussian
  - Been there before, right?

- We can’t trust decorrelation, we need “statistical independence”
  - We should use independent component analysis!
ICA for array mixtures

• Given the input:
  \[ \hat{s}(t) = W \cdot x(t) \]

• Find \( W \) so that the transformed data has maximally statistically independent elements

• So far we cared about features \( (W) \), now we care about the transformed data itself \( (s) \)
Trying this on our dataset

- We can now separated the mixture!

\[ \begin{align*}
\mathbf{x}(t) & = \mathbf{A} \cdot \hat{\mathbf{s}}(t) \\
& = \begin{bmatrix} 2 & 1 \\ 1 & 1 \end{bmatrix} \cdot \begin{bmatrix} -1.3 & 2.7 \\ 2.5 & -2.5 \end{bmatrix} \\
\mathbf{s}(t) & = -1.3 \begin{bmatrix} 2.7 \\ -2.5 \end{bmatrix}
\end{align*} \]
A couple of issues (not bad ones yet)

- Permutation invariance
  - Order of outputs is random

- Scaling invariance
  - Scale is also random

- ICA will actually recover:
  \[ \hat{s}(t) = D \cdot P \cdot s(t) \]
  - Where \( D \) is diagonal and \( P \) is a permutation matrix
Instantaneous mixing limitations

• Sounds don’t mix instantaneously
• There are multiple effects
  • Room reflections
  • Sensor response
  • Propagation delays
  • Propagation and reflection filtering
• Most can be seen as filters
• We must model a convolutive mixture
Convolutive mixing

- Instead of *instantaneous mixing*:
  \[ x_i(t) = \sum_{j=1}^n a_{ij} s_j(t) \]
- We now have *convolutive mixing*:
  \[ x_i(t) = \sum_j \sum_k a_{ij}(k) s_j(t - k) \]
- The mixing filters \( a_{ij}(k) \) capture all the mixing effects in this model
- How do we do ICA now?
**FIR matrix algebra**

- Matrices with FIR filters as elements

\[ \mathbf{A} = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} \]

\[ a_{ij} = \begin{bmatrix} a_{ij}(0) & \cdots & a_{ij}(k-1) \end{bmatrix} \]

- FIR matrix multiplication performs convolution and accumulation

\[ \mathbf{A} \cdot \mathbf{b} = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} \cdot \begin{bmatrix} b_1 \\ b_2 \end{bmatrix} = \begin{bmatrix} a_{11} * b_1 + a_{12} * b_2 \\ a_{21} * b_1 + a_{22} * b_2 \end{bmatrix} \]
Back to convolutive mixing

- Rewrite convolutive mixing as:

$$x_i(t) = \sum_j \sum_k a_{i j}(k)s_j(t - k) \Rightarrow$$

$$\Rightarrow x(t) = A \cdot s(t) = \begin{bmatrix}
a_{11} * s_1(t) + a_{12} * s_2(t) \\
a_{21} * s_1(t) + a_{22} * s_2(t)
\end{bmatrix}$$

- Tidier formulation!

- We can use the FIR matrix abstraction to solve this problem
The easy solution

• Straightforward translation of instantaneous learning rules using FIR matrices, e.g.:

\[
\Delta W \propto \left( I - f(W \cdot x) \cdot (W \cdot x)^\top \right) \cdot W
\]

• Not so easy with algebraic approaches
  • FIR tensors, FIR eigendecompositions, etc ...

• Multiple other (and more rigorous/better behaved) approaches have been developed
Complications with this approach

- Required convolutions are expensive
  - Real-room filters are long
  - Their FIR inverses are very long
  - FIR products can become very time consuming

- Convergence is hard to achieve
  - Huge parameter space
  - Tightly interwoven parameter relationships

- A slow optimization nightmare!
FIR matrix algebra, part II

- FIR matrices have frequency domain versions:

\[
\mathbf{A} = \begin{bmatrix}
a_{11} & a_{12} \\
a_{21} & a_{22}
\end{bmatrix} \xrightarrow{\text{frequency domain}} \hat{\mathbf{A}} = \begin{bmatrix}
\hat{a}_{11} & \hat{a}_{12} \\
\hat{a}_{21} & \hat{a}_{22}
\end{bmatrix}
\]

\[
\hat{a}_{ij} = \text{DFT} [a_{ij}]
\]

- And their products are simpler:

\[
\hat{\mathbf{A}} \cdot \hat{\mathbf{b}} = \begin{bmatrix}
\hat{a}_{11} \circ \hat{b}_1 + \hat{a}_{12} \circ \hat{b}_2 \\
\hat{a}_{21} \circ \hat{b}_1 + \hat{a}_{22} \circ \hat{b}_2
\end{bmatrix}
\]

\[
\hat{a} \circ \hat{b} = \begin{bmatrix}
a(0) \cdot b(0) & \cdots & a(k-1) \cdot b(k-1)
\end{bmatrix}
\]
Yet another formulation

- We can now model the mixing process in the frequency domain:

\[ \hat{X} = \hat{A} \cdot \hat{S} \]

- For every frequency slice we have:

\[
\begin{bmatrix}
\hat{X}_{1}(f,t) \\
\hat{X}_{2}(f,t)
\end{bmatrix} =
\begin{bmatrix}
\hat{a}_{1,1}^{(f)} & \hat{a}_{1,2}^{(f)} \\
\hat{a}_{2,1}^{(f)} & \hat{a}_{2,2}^{(f)}
\end{bmatrix} \cdot
\begin{bmatrix}
\hat{s}_{1}^{(f,t)} \\
\hat{s}_{2}^{(f,t)}
\end{bmatrix}
\]

- Looks familiar?
Overall flowgraph

Convolved Mixtures

\( x_1 \)
\( x_2 \)
\( \ldots \)
\( x_N \)

Input mixture

DFT

Frequency Transform

\( W_1 \)
\( W_2 \)
\( \ldots \)
\( W_M \)

Instantaneous ICA unmixers

Mixed Frequency Bins

Unmixed Frequency Bins

Time Transform

Recovered Sources

\( s_1 \)
\( s_2 \)
\( \ldots \)
\( s_N \)

Source 1

Source 2

Source 1 in normal speed
Some complications ...

- **Permutation issues**
  - We don’t know which source will end up in each narrowband output ...
  - Resulting output can have separated narrowband elements from both sounds!

- **Scaling issues**
  - Narrowband outputs can be scaled arbitrarily
  - This results in spectrally colored outputs
Scaling issue

• One simple fix is to normalize the separating matrices:

\[ \mathbf{W}_{f}^{\text{norm}} = \mathbf{W}_{f}^{\text{orig}} \cdot \left| \mathbf{W}_{f}^{\text{orig}} \right|^{1/N} \]

• Results into more reasonable scaling

• More sophisticated approaches exist but this is not a major problem

• Some spectral coloration is unavoidable
The permutation problem

- Continuity of unmixing matrices
  - Adjacent unmixing matrices tend to be similar
  - We can permute/bias them accordingly (doesn’t work that great)

- Smoothness of spectral output
  - Narrowband components from each source tend to modulate the same way
  - Permute unmixing matrices to ensure adjacent narrowband output are similarly modulated (works ok)

- The above can fail miserably for more than two sources!
  - Combinatorial explosion!
Beamforming and ICA

• If we know the placement of the sensors we can obtain the spatial response of the ICA solution

• ICA places nulls to cancel out interfering sources
  • Just as in the instantaneous case we cancel out sources

• We can visualize the permutation problem now
  • Out of place bands
Beamforming to resolve permutations

- Spatial information can be used to resolve permutations
  - Find permutations that preserve zeros or smooth out the responses
- Works fine, although can be flaky if the array response is not clean
The N-input N-output problem

• ICA, in either formulation inverts a square mixing matrix
  • Implies that we have the same number of input as outputs
  • E.g. in a street with 30 sources we need at least 30 mics!

• Solutions exist for $M$ ins – $N$ outs where $M > N$

• If $N > M$ we can only beamform
  • In some cases extra sources can be treated as noise
  • This can be very restrictive
    • But we’ll see ways out of it in the next lecture
Recap

• Sensor arrays

• Beamforming and localization

• Array source separation
  • FIR Matrix algebra
Next lecture

• Underconstrained source separation

• Single-channel source separation
Reading

- Beamforming and localization

- Blind source separation and ICA
Blind-date afternoon!!

- What better event after Blind Source Separation! :)

- If you don’t have a project-mate stick around after class
  - Meet eligible project mates!
  - Mingle and find cool projects