Normative Networks for Source Separation via Local Plasticity and Dendritic Computation

Bariscan Bozkurt*, Efe Ali Gorguner*,
Francesco Innocenti, Rafal Bogacz

Gatsby Computational Neuroscience Unit, UCL
Department of Computer Science, University of Oxford
Brain Network Dynamics Unit, University of Oxford
MRC Centre of Research Excellence in Restorative Neural Dynamics

Motivation

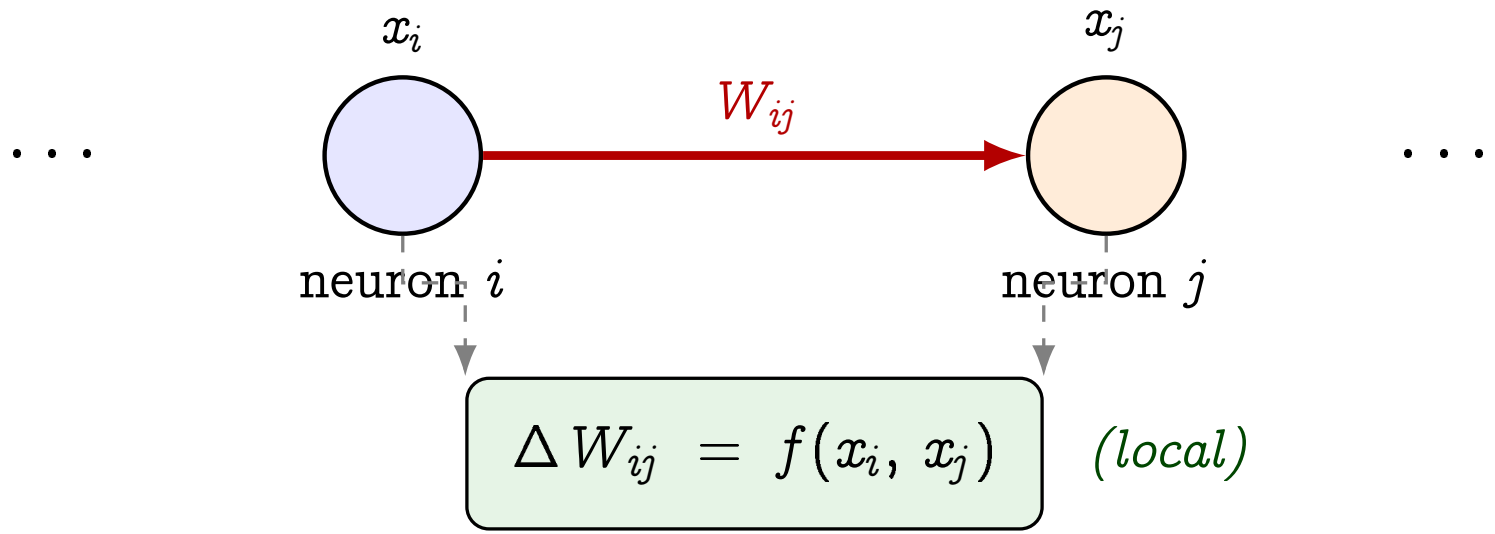
- Sensory inputs are often mixtures of hidden sources.
- Sensory processing and learning rules in the brain.

Motivation

- Sensory inputs are often mixtures of hidden sources.
- Sensory processing and learning rules in the brain.

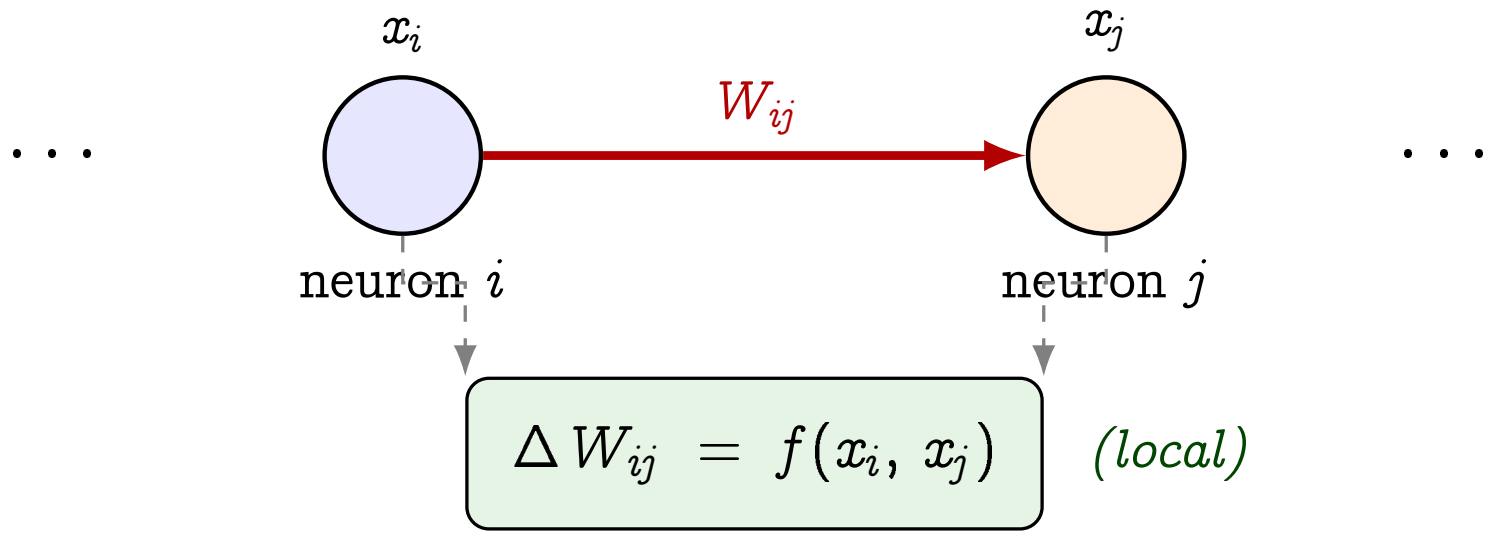
Biological Learning

- Weight updates only depend on the variables encoded in pre- and post-synaptic neurons.
- Online learning



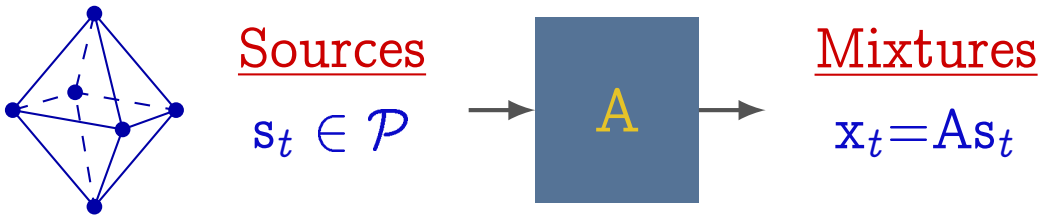
Biological Learning

- Weight updates only depend on the variables encoded in pre- and post-synaptic neurons.
- Online learning



Blind Source Separation: Problem Setup

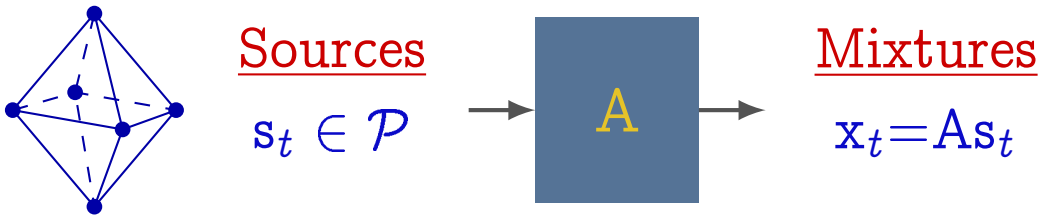
Linear mixing model: $\mathrm{x}(t) = \mathrm{A}\,\mathrm{s}(t)$.



- $\mathrm{s}(t) \in \mathcal{P}$: sources (unknown)
- A: mixing matrix (unknown)
- $\mathrm{x}(t)$: observed mixtures (known)
- Goal: learn separator W with $\mathrm{y}(t) = \mathrm{W}\mathrm{x}(t) \approx \mathrm{s}(t)$

Blind Source Separation: Problem Setup

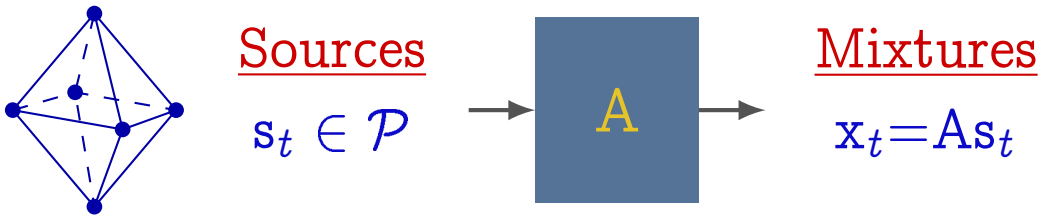
Linear mixing model: $\mathrm{x}(t) = \mathrm{A}\,\mathrm{s}(t)$.



- $\mathrm{s}(t) \in \mathcal{P}$: sources (unknown)
- A: mixing matrix (unknown)
- $\mathrm{x}(t)$: observed mixtures (known)
- Goal: learn separator W with $\mathrm{y}(t) = \mathrm{W}\mathrm{x}(t) \approx \mathrm{s}(t)$

Blind Source Separation: Problem Setup

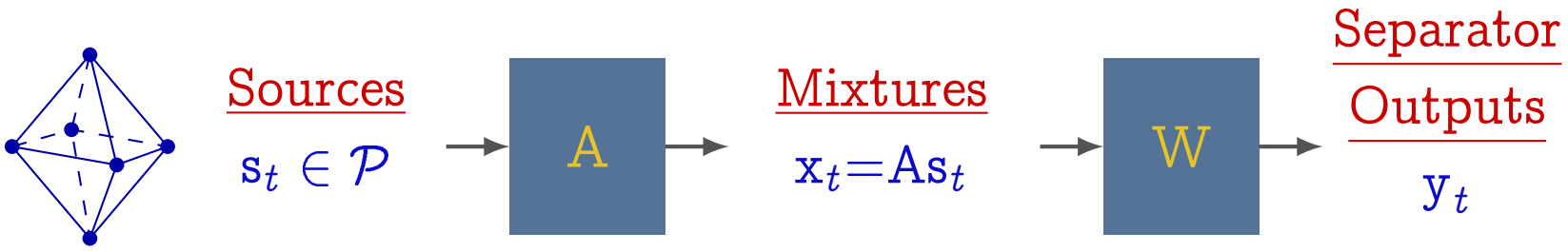
Linear mixing model: $\mathrm{x}(t) = \mathrm{A}\,\mathrm{s}(t)$.



- $\mathrm{s}(t) \in \mathcal{P}$: sources (unknown)
- A: mixing matrix (unknown)
- $\mathrm{x}(t)$: observed mixtures (known)
- Goal: learn separator W with $\mathrm{y}(t) = \mathrm{W}\mathrm{x}(t) \approx \mathrm{s}(t)$

Blind Source Separation: Problem Setup

Linear mixing model: $\mathrm{x}(t) = \mathrm{A}\,\mathrm{s}(t)$.



- $\mathrm{s}(t) \in \mathcal{P}$: sources (unknown)
- A: mixing matrix (unknown)
- $\mathrm{x}(t)$: observed mixtures (known)
- Goal: learn separator W with $\mathrm{y}(t) = \mathrm{W}\mathrm{x}(t) \approx \mathrm{s}(t)$

Determinant/Entropy Maximization

Spread the outputs to fill the source domain:

$$\underset{\mathrm{W},\,\mathrm{Y}}{\text{maximize}} \quad \log\det\left(\hat{C} + \varepsilon I\right)$$

$$\text{subject to} \quad \mathrm{y}_t = \mathrm{W}\mathrm{x}_t, \quad \mathrm{y}_t \in \mathcal{P}$$

Directly optimizing the log determinant objective $\rightarrow$ dependence on $\hat{C}^{-1}$!

Key idea - 2nd-order Taylor expansion:

$$\log\det(\hat{C} + \varepsilon I) \approx \underbrace{\sum_i \log(\hat{v}_i + \varepsilon)}_{\text{variance expansion}} - \underbrace{\frac{1}{2}\sum_{i \neq j} \frac{\hat{c}_{ij}^2}{(\hat{v}_i + \varepsilon)(\hat{v}_j + \varepsilon)}}_{\text{normalised covariance penalty}}$$

Determinant/Entropy Maximization

Spread the outputs to fill the source domain:

$$\underset{\mathrm{W},\,\mathrm{Y}}{\text{maximize}} \quad \log\det\left(\hat{C} + \varepsilon I\right)$$

$$\text{subject to} \quad \mathrm{y}_t = \mathrm{W}\mathrm{x}_t, \quad \mathrm{y}_t \in \mathcal{P}$$

Directly optimizing the log determinant objective $\rightarrow$ dependence on $\hat{C}^{-1}$!

Key idea - 2nd-order Taylor expansion:

$$\log\det(\hat{C} + \varepsilon I) \approx \underbrace{\sum_i \log(\hat{v}_i + \varepsilon)}_{\text{variance expansion}} - \underbrace{\frac{1}{2}\sum_{i \neq j} \frac{\hat{c}_{ij}^2}{(\hat{v}_i + \varepsilon)(\hat{v}_j + \varepsilon)}}_{\text{normalised covariance penalty}}$$

Determinant/Entropy Maximization

Spread the outputs to fill the source domain:

$$\underset{\mathrm{W},\,\mathrm{Y}}{\text{maximize}} \quad \log\det\left(\hat{C}+\varepsilon I\right)$$

$$\text{subject to} \quad \mathrm{y}_t = \mathrm{W}\mathrm{x}_t, \quad \mathrm{y}_t \in \mathcal{P}$$

Directly optimizing the log determinant objective $\rightarrow$ dependence on $\hat{C}^{-1}$!

Key idea - 2nd-order Taylor expansion:

$$\log\det(\hat{C}+\varepsilon I) \approx \underbrace{\sum_i \log(\hat{v}_i+\varepsilon)}_{\text{variance expansion}} - \underbrace{\frac{1}{2}\sum_{i\neq j}\frac{\hat{c}_{ij}^2}{(\hat{v}_i+\varepsilon)(\hat{v}_j+\varepsilon)}}_{\text{normalised covariance penalty}}$$

Algorithm

Now, our cost function becomes

$$\mathcal{J} = -(\text{log-det approximation}) + \gamma \left\| \mathrm{y} - \mathrm{W}\mathrm{x} \right\|^2$$

For each incoming mixture x_t,

1. Fast updates:
   - $\Delta \mathrm{y} \propto -\frac{\partial \mathcal{J}}{\partial \mathrm{y}}$
2. Slow updates:
   - $\Delta \mathrm{W} \propto -\frac{\partial \mathcal{J}}{\partial \mathrm{W}}$
   - update $\hat{v}_i$ and $\hat{c}_{ij}$ to track variance/covariance

Algorithm

Now, our cost function becomes

$$\mathcal{J} = -(\text{log-det approximation}) + \gamma \left\| \mathrm{y} - \mathrm{W}\mathrm{x} \right\|^2$$

For each incoming mixture x_t,

1. Fast updates:
   - $\Delta \mathrm{y} \propto -\frac{\partial \mathcal{J}}{\partial \mathrm{y}}$
2. Slow updates:
   - $\Delta \mathrm{W} \propto -\frac{\partial \mathcal{J}}{\partial \mathrm{W}}$
   - update $\hat{v}_i$ and $\hat{c}_{ij}$ to track variance/covariance

Algorithm

Now, our cost function becomes

$$\mathcal{J} = -(\text{log-det approximation}) + \gamma \left\| \mathrm{y} - \mathrm{W}\mathrm{x} \right\|^2$$

For each incoming mixture x_t,

1. Fast updates:
   - $\Delta \mathrm{y} \propto -\frac{\partial \mathcal{J}}{\partial \mathrm{y}}$
2. Slow updates:
   - $\Delta \mathrm{W} \propto -\frac{\partial \mathcal{J}}{\partial \mathrm{W}}$
   - update $\hat{v}_i$ and $\hat{c}_{ij}$ to track variance/covariance

Algorithm

Now, our cost function becomes

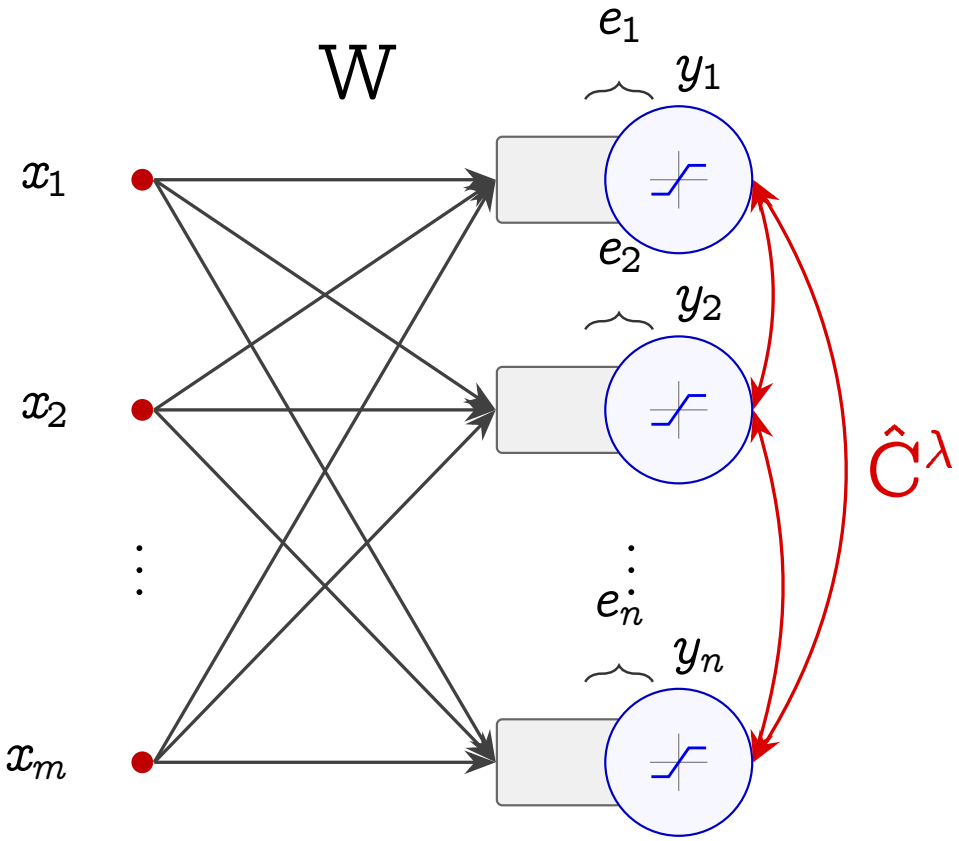
$$\mathcal{J} = -(\text{log-det approximation}) + \gamma \left\| \mathrm{y} - \mathrm{W}\mathrm{x} \right\|^2$$

For each incoming mixture x_t,

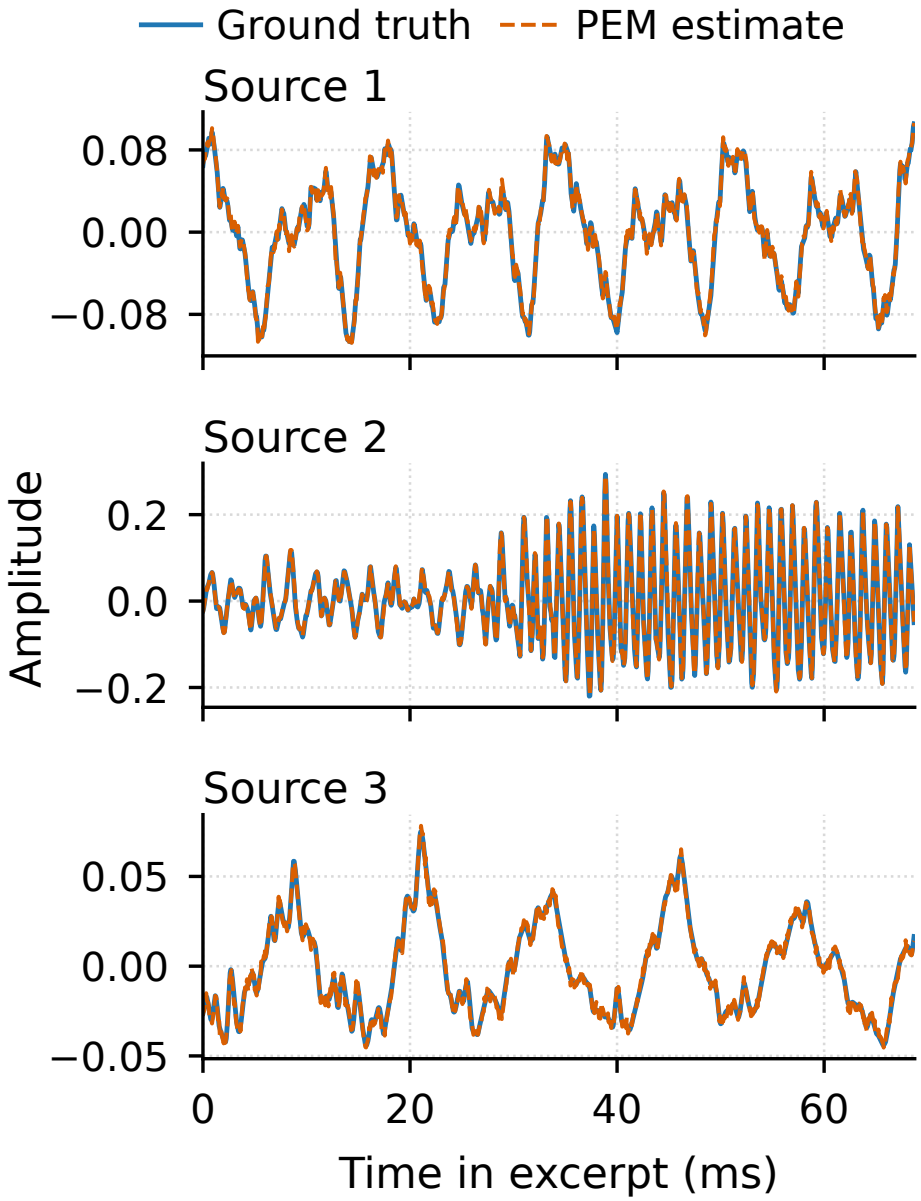
1. Fast updates:
   - $\Delta \mathrm{y} \propto -\frac{\partial \mathcal{J}}{\partial \mathrm{y}}$
2. Slow updates:
   - $\Delta \mathrm{W} \propto -\frac{\partial \mathcal{J}}{\partial \mathrm{W}}$
   - update $\hat{v}_i$ and $\hat{c}_{ij}$ to track variance/covariance

Neural Implementation

1. Fast updates:
   - $\Delta \mathrm{y} \propto -\frac{\partial \mathcal{J}}{\partial \mathrm{y}}$ ⟶ Neural dynamics
2. Slow updates:
   - $\Delta \mathrm{W} \propto -\frac{\partial \mathcal{J}}{\partial \mathrm{W}}$ and update $\hat{v}_i$ and $\hat{c}_{ij}$ ⟶ Local plasticity

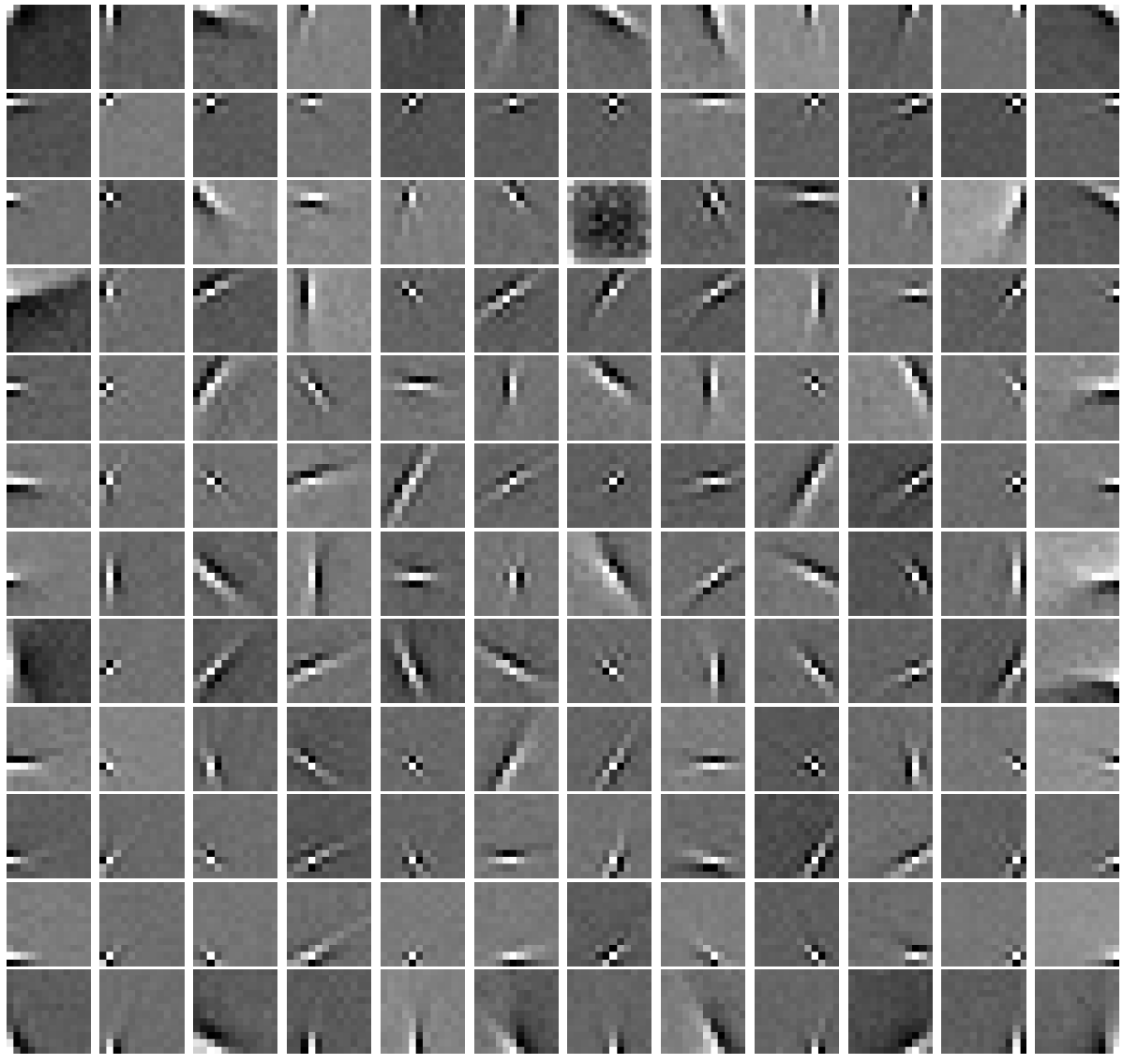


Auditory Sources

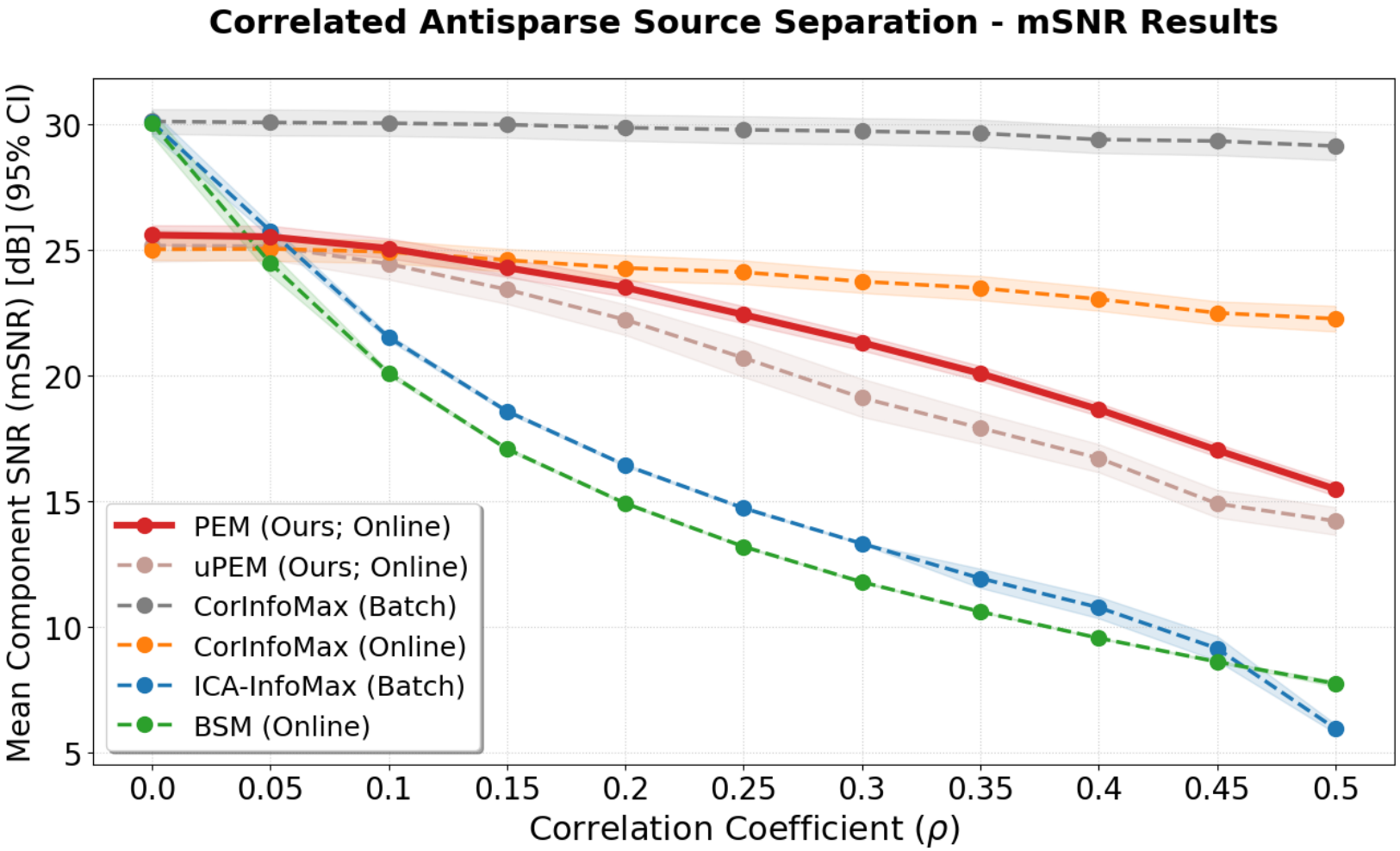


Receptive Fields

Receptive Filters Learned from Natural Scenes



Correlated Sources



Summary

Predictive Entropy Maximization (PEM)

Online, biologically plausible algorithm for blind source separation.

- Taylor approximation of the entropy maximization objective eliminates $\hat{C}^{-1}$.
- Algorithm maps directly onto neural implementation.
- Strong empirical performance.

Summary

Predictive Entropy Maximization (PEM)

Online, biologically plausible algorithm for blind source separation.

- Taylor approximation of the entropy maximization objective eliminates $\hat{C}^{-1}$.
- Algorithm maps directly onto neural implementation.
- Strong empirical performance.

Summary

Predictive Entropy Maximization (PEM)

Online, biologically plausible algorithm for blind source separation.

- Taylor approximation of the entropy maximization objective eliminates $\hat{C}^{-1}$.
- Algorithm maps directly onto neural implementation.
- Strong empirical performance.