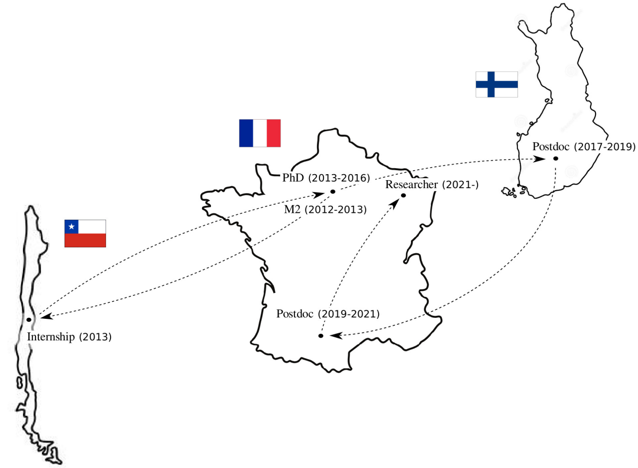
Towards deep phase recovery for audio source separation

Seminar at Audio Research Group, Tampere University, Finland
August 30, 2023

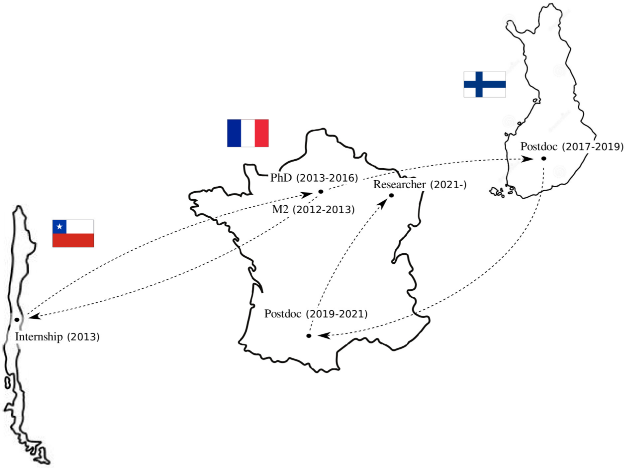
Paul Magron

Université de Lorraine, CNRS, Inria, LORIA, Nancy, France

A brief history of me



A brief history of me



MULTISPEECH

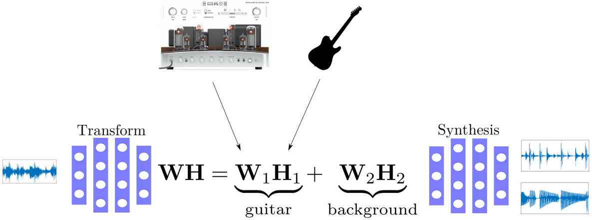
Speech Modeling for Facilitating Oral-Based Communication

Research themes

- Speech enhancement for auditory neuropathy (with N. Monir, R. Serizel).
- Audio inpainting / restoration (with L. Bahrman, M. Krémé, A. Deleforge).

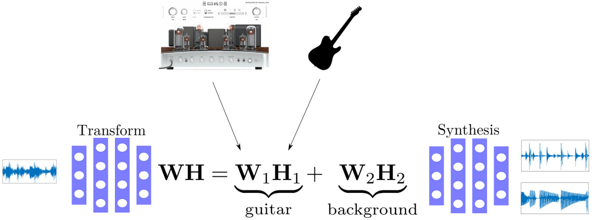
Research themes

- Speech enhancement for auditory neuropathy (with N. Monir, R. Serizel).
- Audio inpainting / restoration (with L. Bahrman, M. Krémé, A. Deleforge).
- Combining dictionary models and deep learning (with L. Lalay, M. Sadeghi).
- Joint synthesis / source separation.



Research themes

- ▷ Speech enhancement for auditory neuropathy (with N. Monir, R. Serizel).
- ▷ Audio inpainting / restoration (with L. Bahrman, M. Krémé, A. Deleforge).
- ▷ Combining dictionary models and deep learning (with L. Lalay, M. Sadeghi).
- ▷ Joint synthesis / source separation.



- ▷ **Source separation** (with so many people).

Audio source separation

Audio source separation

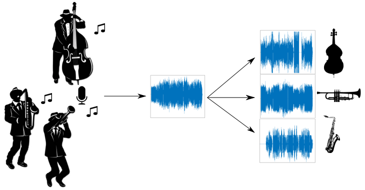
- Audio signals are composed of several constitutive sounds: multiple speakers, background noise, domestic sounds, musical instruments...

Audio source separation

- Audio signals are composed of several constitutive sounds: multiple speakers, background noise, domestic sounds, musical instruments...

Source separation = recovering the sources from the mixture.

- Augmented mixing (from mono to stereo).
- An important preprocessing for many analysis tasks (speech recognition, melody extraction...).

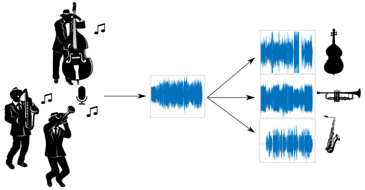


Audio source separation

- ▷ Audio signals are composed of several constitutive sounds: multiple speakers, background noise, domestic sounds, musical instruments...

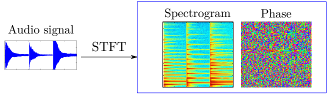
Source separation = recovering the sources from the mixture.

- ▷ Augmented mixing (from mono to stereo).
- ▷ An important preprocessing for many analysis tasks (speech recognition, melody extraction...).



Framework

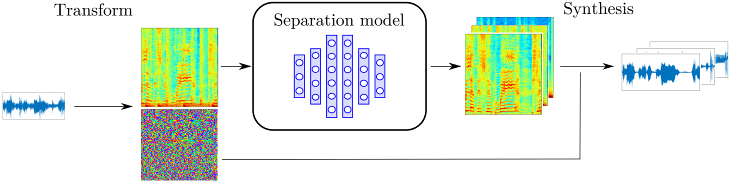
- ▷ Monaural signals.
- ▷ Short-time Fourier transform (STFT)-domain separation.
- ▷ Mixture model: $\mathbf{X} = \sum_{j=1}^{J} \mathbf{S}_j$.



$$\mathbf{x} \in \mathbb{R}^N \xrightarrow{\text{STFT}} \mathbf{X} \in \mathbb{C}^{F \times T}$$

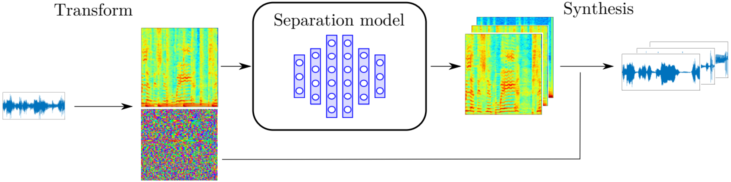
Typical separation pipeline

Nonnegative time-frequency (TF) masking:



Typical separation pipeline

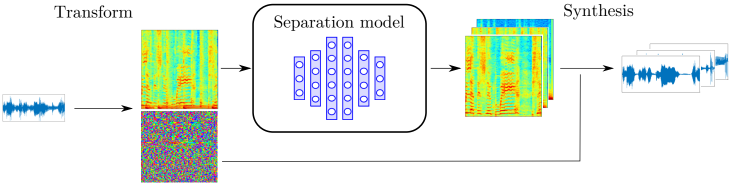
Nonnegative time-frequency (TF) masking:



- ▷ A nonnegative representation is processed (e.g., magnitude or power spectrogram).

Typical separation pipeline

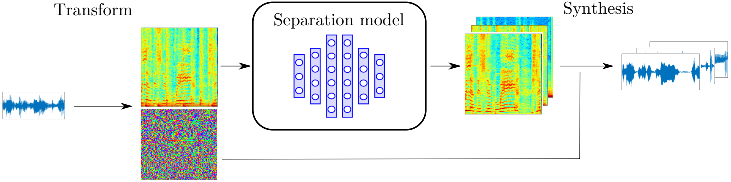
Nonnegative time-frequency (TF) masking:



- ▷ A nonnegative representation is processed (e.g., magnitude or power spectrogram).
- ▷ The separator is a deep neural network, trained using a (large) dataset with isolated sources.

Typical separation pipeline

Nonnegative time-frequency (TF) masking:



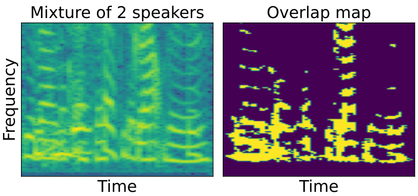
- A nonnegative representation is processed (e.g., magnitude or power spectrogram).
- The separator is a deep neural network, trained using a (large) dataset with isolated sources.
- The mixture's phase is assigned to each source using a Wiener-like filter or masking process.

The phase problem

✗ Nonnegative masking: Issues in sound quality when sources *overlap* in the TF domain.

$$|X| \neq |S_1| + |S_2|$$

$$\angle X \neq \angle S_1 \text{ or } \angle S_2$$

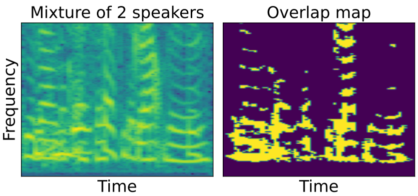


The phase problem

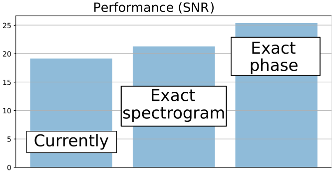
✗ Nonnegative masking: Issues in sound quality when sources *overlap* in the TF domain.

$$|X| \neq |S_1| + |S_2|$$

$$\angle X \neq \angle S_1 \text{ or } \angle S_2$$

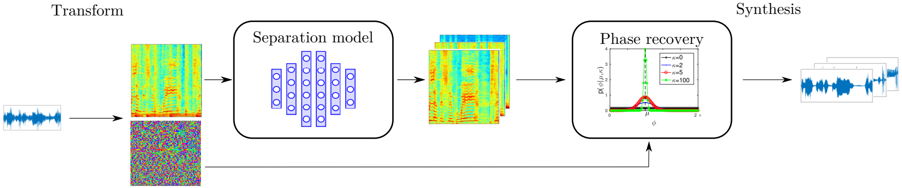


The potential of phase recovery

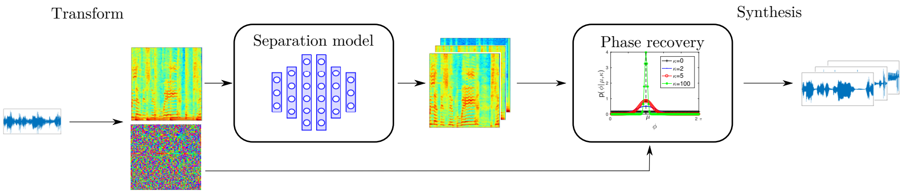


Given the current state-of-the-art, more potential gain in phase recovery than in magnitude estimation.

Phase recovery for source separation

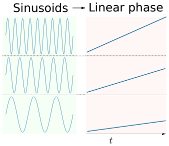


Phase recovery for source separation

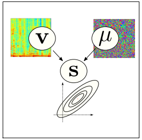


Main contributions

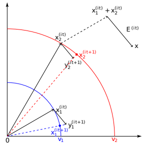
Phase models



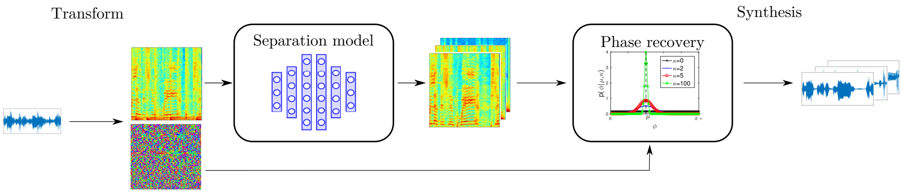
Statistical framework



Iterative algorithms

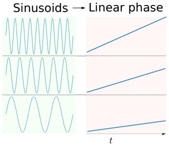


Phase recovery for source separation

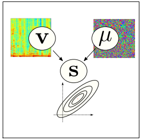


Main contributions

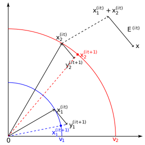
Phase models



Statistical framework



Iterative algorithms



- ▷ So far using “old-school” signal processing.
- ▷ Perspective: leveraging deep learning for phase recovery.

Phase models

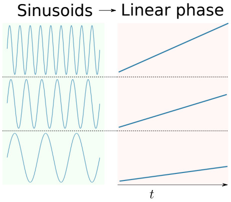
Sinusoidal phase model

Consider a mixture of sinusoids: $x(n) = \sum_{p=1}^{P} A_p \sin(2\pi \underbrace{\nu_p}_{\text{normalized frequency}} n + \phi_{0,p})$.

Sinusoidal phase model

Consider a mixture of sinusoids: $x(n) = \sum_{p=1}^{P} A_p \sin(2\pi \underbrace{\nu_p}_{\text{normalized frequency}} n + \phi_{0,p})$.

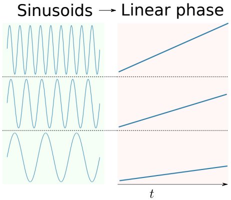
The STFT phase follows: $\mu_{f,t} = \mu_{f,t-1} + l\nu_{f,t}$

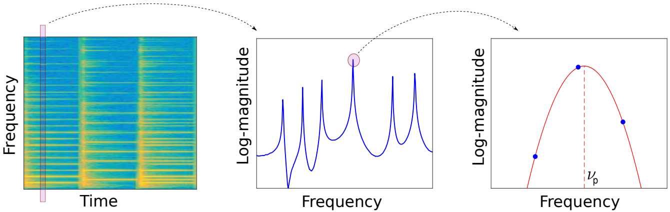


Sinusoidal phase model

Consider a mixture of sinusoids: $x(n) = \sum_{p=1}^{P} A_p \sin(2\pi \underbrace{\nu_p}_{\text{normalized frequency}} n + \phi_{0,p})$.

The STFT phase follows: $\mu_{f,t} = \mu_{f,t-1} + l\nu_{f,t}$

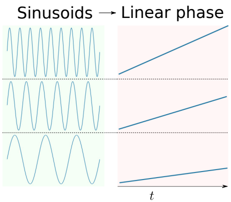


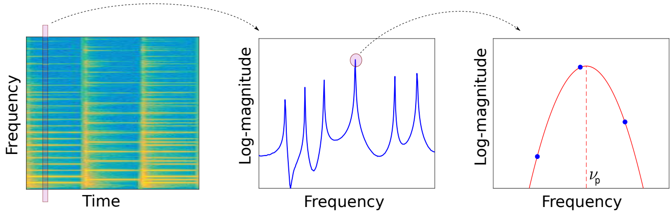


Sinusoidal phase model

Consider a mixture of sinusoids: $x(n) = \sum_{p=1}^{P} A_p \sin(2\pi \underbrace{\nu_p}_{\text{normalized frequency}} n + \phi_{0,p})$.

The STFT phase follows: $\mu_{f,t} = \mu_{f,t-1} + l\nu_{f,t}$





✓ Useful for source separation (and audio inpainting) applications.

✗ The performance is limited due to the simplicity of the model.

Perspective: towards deep phase models

Recently: Some attempts at predicting the phase using DNNs.

- ✗ Generic architectures which do not account for the particular phase structure.
- ✗ Cumbersome two-stage approaches to resolve some ambiguities.

Perspective: towards deep phase models

Recently: Some attempts at predicting the phase using DNNs.

- ✗ Generic architectures which do not account for the particular phase structure.
- ✗ Cumbersome two-stage approaches to resolve some ambiguities.

Proposal: Generalize phase models from signal analysis using deep learning.

$$\boldsymbol{\mu}_t = \boldsymbol{\mu}_{t-1} + l\boldsymbol{\nu}_t \quad \rightarrow \quad \boldsymbol{\mu}_t = \underbrace{\mathcal{R}(\boldsymbol{\nu}_t, \boldsymbol{\mu}_{t-1}, \dots, \boldsymbol{\mu}_{t-\tau})}_{\text{temporal dynamics}} \quad \text{with} \quad \boldsymbol{\nu}_t = \underbrace{\mathcal{C}(|\mathbf{x}|_t)}_{\text{frequency extraction}}$$

Perspective: towards deep phase models

Recently: Some attempts at predicting the phase using DNNs.

- ✗ Generic architectures which do not account for the particular phase structure.
- ✗ Cumbersome two-stage approaches to resolve some ambiguities.

Proposal: Generalize phase models from signal analysis using deep learning.

$$\boldsymbol{\mu}_t = \boldsymbol{\mu}_{t-1} + l\boldsymbol{\nu}_t \quad \rightarrow \quad \boldsymbol{\mu}_t = \underbrace{\mathcal{R}(\boldsymbol{\nu}_t, \boldsymbol{\mu}_{t-1}, \ldots, \boldsymbol{\mu}_{t-\tau})}_{\text{temporal dynamics}} \quad \text{with} \quad \boldsymbol{\nu}_t = \underbrace{\mathcal{C}(|\mathbf{x}|_t)}_{\text{frequency extraction}}$$

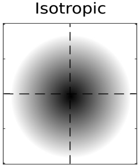
- ▷ Architectural choices (non-linearities, loss functions) adapted to the phase (periodicity).
- ▷ Identify and exploit perceptual phase invariants.

Probabilistic phase modeling

Phase-aware Gaussian models

The ubiquitous isotropic Gaussian model:

$$s \sim \mathcal{N}_{\mathbb{C}}(m, \Gamma) \text{ with } \Gamma = \begin{pmatrix} \gamma & 0 \\ 0 & \gamma \end{pmatrix}$$



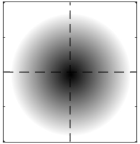
Equivalent to assuming a uniform phase $\angle s \sim \mathcal{U}_{[0,2\pi[}$.

- ✗ Impossible to promote any phase structure / prior.

Phase-aware Gaussian models

The ubiquitous isotropic Gaussian model:

$$s \sim \mathcal{N}_{\mathbb{C}}(m, \Gamma) \text{ with } \Gamma = \begin{pmatrix} \gamma & 0 \\ 0 & \gamma \end{pmatrix}$$

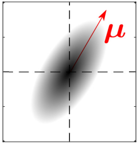


Equivalent to assuming a uniform phase $\angle s \sim \mathcal{U}_{[0,2\pi[}$.

✗ Impossible to promote any phase structure / prior.

Anisotropic Gaussian model

$$s \sim \mathcal{N}_{\mathbb{C}}(m, \Gamma) \text{ with } \Gamma = \begin{pmatrix} \gamma & c \\ \bar{c} & \gamma \end{pmatrix}$$



c is the *relation* term, defined as a function of the phase parameter μ.

✓ Allows to incorporate phase priors; nice performance boost for source separation applications (e.g., phase-aware Wiener filter).

Perspective: anisotropic deep learning

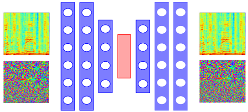
✗ Bayesian deep learning / variational autoencoders (VAE) are limited to isotropic distributions.

Perspective: anisotropic deep learning

✗ Bayesian deep learning / variational autoencoders (VAE) are limited to isotropic distributions.

Proposal: Combine deep learning and anisotropic modeling, e.g., via anisotropic VAEs.

$$\underbrace{\mathbf{z}|\mathbf{x} \sim \mathcal{N}_{\mathbb{C}}(\psi_{\text{enc}}(\mathbf{x}), \mathbf{\Gamma}_{\text{enc}})}_{\text{encoder}} \quad \text{and} \quad \underbrace{\mathbf{s}|\mathbf{z} \sim \mathcal{N}_{\mathbb{C}}(\psi_{\text{dec}}(\mathbf{z}), \mathbf{\Gamma}_{\text{dec}})}_{\text{decoder}}$$

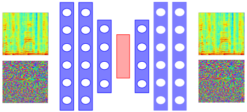


Perspective: anisotropic deep learning

✗ Bayesian deep learning / variational autoencoders (VAE) are limited to isotropic distributions.

Proposal: Combine deep learning and anisotropic modeling, e.g., via anisotropic VAEs.

$$\underbrace{\mathbf{z}|\mathbf{x} \sim \mathcal{N}_{\mathbb{C}}(\psi_{\text{enc}}(\mathbf{x}), \mathbf{\Gamma}_{\text{enc}})}_{\text{encoder}} \quad \text{and} \quad \underbrace{\mathbf{s}|\mathbf{z} \sim \mathcal{N}_{\mathbb{C}}(\psi_{\text{dec}}(\mathbf{z}), \mathbf{\Gamma}_{\text{dec}})}_{\text{decoder}}$$



▷ A strong effort in modeling and optimization is needed for deriving appropriate estimation techniques.

Spectrogram inversion algorithms

Spectrogram inversion

Goal: retrieve (complex-valued) STFTs from (non-negative) spectrograms.

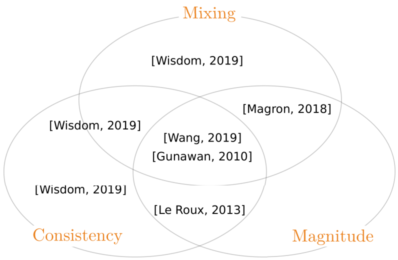
- Identify important properties in the STFT domain.
- Promote them by defining an optimization problem.
- Solve it using some optimization strategy.

Spectrogram inversion

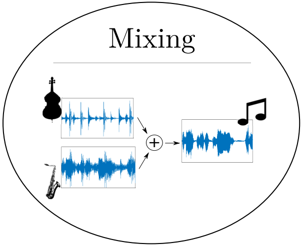
Goal: retrieve (complex-valued) STFTs from (non-negative) spectrograms.

- ▷ Identify important properties in the STFT domain.
- ▷ Promote them by defining an optimization problem.
- ▷ Solve it using some optimization strategy.

- ▷ Many algorithms in the literature!
- ▷ Which problem formulation is the most appropriate in practice?
- ▷ Proposal: let's define a general spectrogram inversion framework.

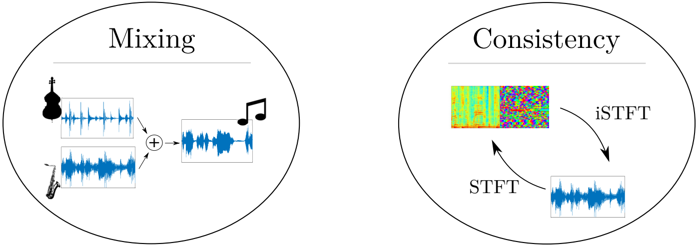


STFT-domain constraints



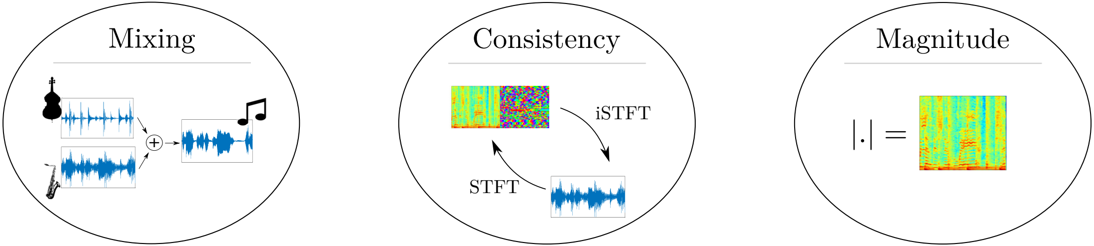
- Mixing: the estimates should be *conservative* = sum up to the mixture, such that there is no creation/destruction of energy.

STFT-domain constraints



- Mixing: the estimates should be *conservative* = sum up to the mixture, such that there is no creation/destruction of energy.
- Consistency: the estimates (=complex-valued matrices) should be the STFT of time-domain signals.

STFT-domain constraints



- Mixing: the estimates should be *conservative* = sum up to the mixture, such that there is no creation/destruction of energy.
- Consistency: the estimates (=complex-valued matrices) should be the STFT of time-domain signals.
- Magnitude match: the estimates' magnitude should remain close to the output of the DNN computed beforehand.

Overview

Proposal: A general framework for deriving spectrogram inversion algorithms

- For each property/objective/constraint, define a loss function (and an auxiliary function).
- Combine them (soft penalties / hard constraints) to formulate optimization problems.
- Derive algorithms that alternate projections on the corresponding constraints subspaces.

Overview

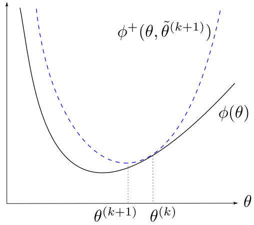
Proposal: A general framework for deriving spectrogram inversion algorithms

- For each property/objective/constraint, define a loss function (and an auxiliary function).
- Combine them (soft penalties / hard constraints) to formulate optimization problems.
- Derive algorithms that alternate projections on the corresponding constraints subspaces.

Auxiliary function method

- Considering minimization of ϕ, construct ϕ^+ such that: $\phi(\theta) = \min_{\tilde{\theta}} \phi^+(\theta, \tilde{\theta})$.
- ϕ is non-increasing when minimizing ϕ^+ with respect to θ and $\tilde{\theta}$ alternately.

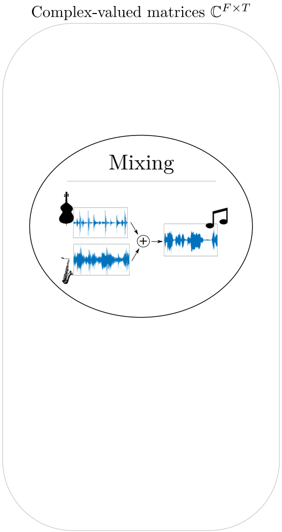
✓ Convergence, successfully used in audio, no hyperparameter to tune.



Mixing constraint

Loss function that promotes conservative estimates:

$$h(\mathbf{S}) = ||\mathbf{X} - \sum_j \mathbf{S}_j||^2$$



Mixing constraint

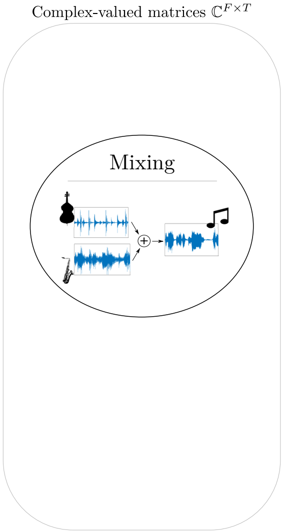
Loss function that promotes conservative estimates:

$$h(\mathbf{S}) = ||\mathbf{X} - \sum_j \mathbf{S}_j||^2$$

Auxiliary function

- Auxiliary parameters $\mathbf{Y}$ such that $\sum_j \mathbf{Y}_j = \mathbf{X}$.
- Positive weights $\mathbf{\Lambda}_j$ such that $\sum_j \lambda_{j,f,t} = 1$.
- Then the following is an auxiliary function for h:

$$h^+(\mathbf{S}, \mathbf{Y}) = \sum_{j,f,t} \frac{|y_{j,f,t} - s_{j,f,t}|^2}{\lambda_{j,f,t}}$$



Mixing constraint

Loss function that promotes conservative estimates:

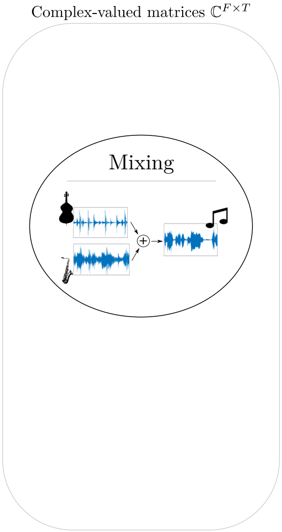
$$h(\mathbf{S}) = ||\mathbf{X} - \sum_j \mathbf{S}_j||^2$$

Auxiliary function

- ▷ Auxiliary parameters $\mathbf{Y}$ such that $\sum_j \mathbf{Y}_j = \mathbf{X}$.
- ▷ Positive weights $\mathbf{\Lambda}_j$ such that $\sum_j \lambda_{j,f,t} = 1$.
- ▷ Then the following is an auxiliary function for h:

$$h^+(\mathbf{S}, \mathbf{Y}) = \sum_{j,f,t} \frac{|y_{j,f,t} - s_{j,f,t}|^2}{\lambda_{j,f,t}}$$

Auxiliary parameters update: $\mathbf{Y}_j = \mathbf{S}_j + \mathbf{\Lambda}_j \odot (\mathbf{X} - \sum_k \mathbf{S}_k)$



Mixing constraint

Loss function that promotes conservative estimates:

$$h(\mathbf{S}) = ||\mathbf{X} - \sum_j \mathbf{S}_j||^2$$

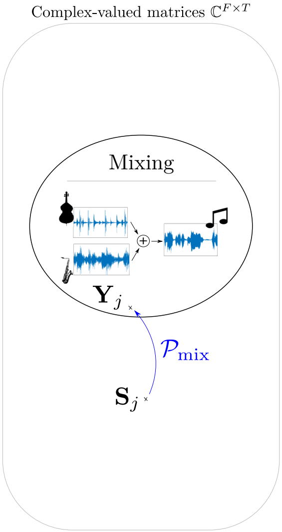
Auxiliary function

- ▷ Auxiliary parameters $\mathbf{Y}$ such that $\sum_j \mathbf{Y}_j = \mathbf{X}$.
- ▷ Positive weights $\mathbf{\Lambda}_j$ such that $\sum_j \lambda_{j,f,t} = 1$.
- ▷ Then the following is an auxiliary function for h:

$$h^+(\mathbf{S}, \mathbf{Y}) = \sum_{j,f,t} \frac{|y_{j,f,t} - s_{j,f,t}|^2}{\lambda_{j,f,t}}$$

Auxiliary parameters update: $\mathbf{Y}_j = \mathbf{S}_j + \mathbf{\Lambda}_j \odot (\mathbf{X} - \sum_k \mathbf{S}_k)$

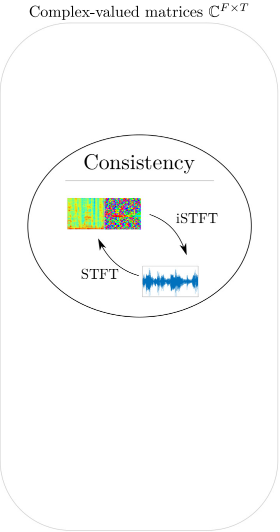
- ▷ Defines a projector $\mathcal{P}_{\text{mix}}$ onto the subspace of matrices complying with the mixing constraint.



Consistency constraint

Loss function that promotes consistent estimates:

$$i(\mathbf{S}) = \sum_j ||\mathbf{S}_j - \mathcal{G}(\mathbf{S}_j)||^2 \text{ with } \mathcal{G} = \text{STFT} \circ \text{iSTFT}$$



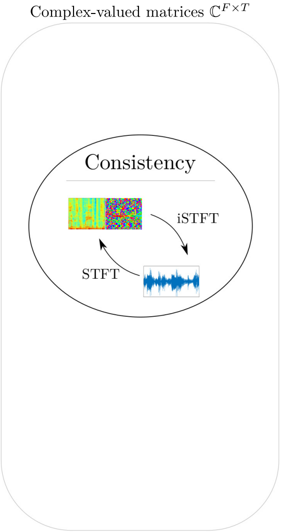
Consistency constraint

Loss function that promotes consistent estimates:

$$i(\mathbf{S}) = \sum_j ||\mathbf{S}_j - \mathcal{G}(\mathbf{S}_j)||^2 \text{ with } \mathcal{G} = \mathsf{STFT} \circ \mathsf{iSTFT}$$

Auxiliary function

- $\mathcal{G}(\mathbf{S}_j)$ is the closest consistent matrix to $\mathbf{S}_j$.
- Then $i^+(\mathbf{S}, \mathbf{Z}) = \sum_j ||\mathbf{S}_j - \mathbf{Z}_j||^2$ (where $\mathbf{Z}_j \in \mathrm{Im}(\mathsf{STFT})$) is an auxiliary function for i.



Consistency constraint

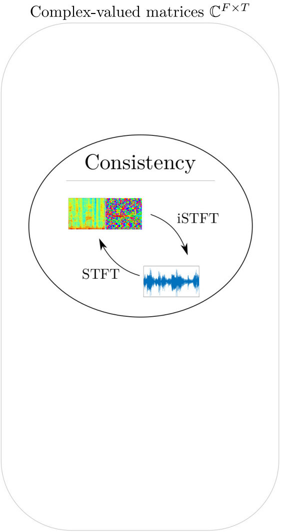
Loss function that promotes consistent estimates:

$$i(\mathbf{S}) = \sum_j ||\mathbf{S}_j - \mathcal{G}(\mathbf{S}_j)||^2 \text{ with } \mathcal{G} = \mathsf{STFT} \circ \mathsf{iSTFT}$$

Auxiliary function

- $\mathcal{G}(\mathbf{S}_j)$ is the closest consistent matrix to $\mathbf{S}_j$.
- Then $i^+(\mathbf{S}, \mathbf{Z}) = \sum_j ||\mathbf{S}_j - \mathbf{Z}_j||^2$ (where $\mathbf{Z}_j \in \mathrm{Im}(\mathsf{STFT})$) is an auxiliary function for i.

Auxiliary parameters update: $\mathbf{Z}_j = \mathcal{G}(\mathbf{S}_j)$



Consistency constraint

Loss function that promotes consistent estimates:

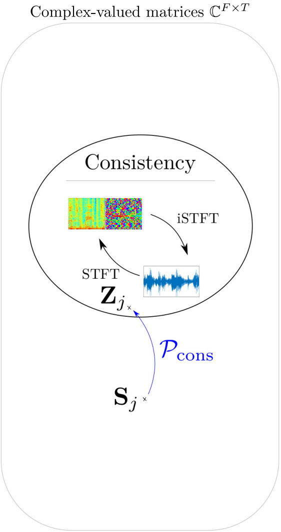
$$i(\mathbf{S}) = \sum_j ||\mathbf{S}_j - \mathcal{G}(\mathbf{S}_j)||^2 \text{ with } \mathcal{G} = \text{STFT} \circ \text{iSTFT}$$

Auxiliary function

- $\mathcal{G}(\mathbf{S}_j)$ is the closest consistent matrix to $\mathbf{S}_j$.
- Then $i^+(\mathbf{S}, \mathbf{Z}) = \sum_j ||\mathbf{S}_j - \mathbf{Z}_j||^2$ (where $\mathbf{Z}_j \in \text{Im(STFT)}$) is an auxiliary function for i.

Auxiliary parameters update: $\mathbf{Z}_j = \mathcal{G}(\mathbf{S}_j)$

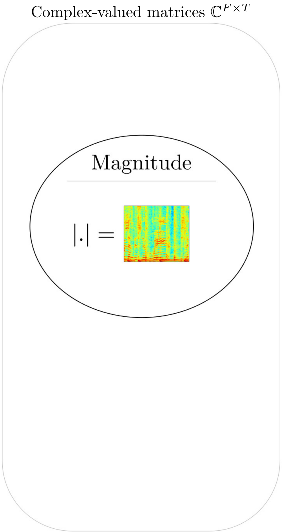
- Defines a projector $\mathcal{P}_{\text{cons}}$ onto the subspace of consistent matrices.



Magnitude constraint

Loss function that ensures the estimates' magnitudes remain close to the target value $\mathbf{V}_j$ estimated beforehand (e.g., using a DNN):

$$m(\mathbf{S}) = \sum_j |||\mathbf{S}_j| - \mathbf{V}_j||^2$$



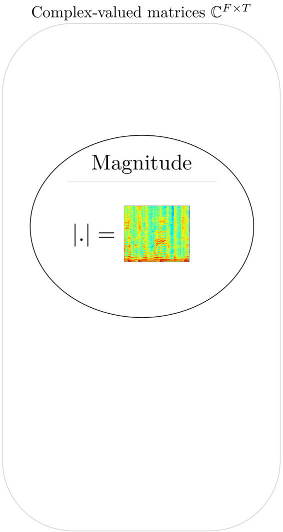
Magnitude constraint

Loss function that ensures the estimates' magnitudes remain close to the target value $\mathbf{V}_j$ estimated beforehand (e.g., using a DNN):

$$m(\mathbf{S}) = \sum_j |||\mathbf{S}_j| - \mathbf{V}_j||^2$$

Auxiliary function

- Auxiliary parameters $\mathbf{U}$ such that $|\mathbf{U}_j| = \mathbf{V}_j$.
- $m^+(\mathbf{S}, \mathbf{Z}) = \sum_j ||\mathbf{S}_j - \mathbf{U}_j||^2$ is an auxiliary function for m.



Magnitude constraint

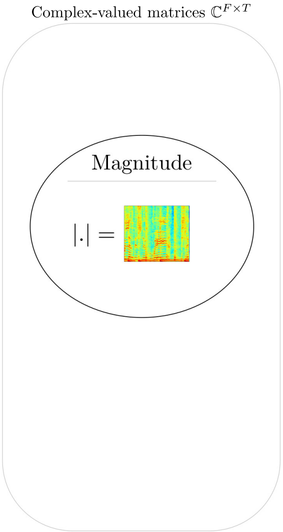
Loss function that ensures the estimates' magnitudes remain close to the target value $\mathbf{V}_j$ estimated beforehand (e.g., using a DNN):

$$m(\mathbf{S}) = \sum_j |||\mathbf{S}_j| - \mathbf{V}_j||^2$$

Auxiliary function

- Auxiliary parameters $\mathbf{U}$ such that $|\mathbf{U}_j| = \mathbf{V}_j$.
- $m^+(\mathbf{S}, \mathbf{Z}) = \sum_j ||\mathbf{S}_j - \mathbf{U}_j||^2$ is an auxiliary function for m.

Auxiliary parameters update: $\mathbf{U}_j = \dfrac{\mathbf{S}_j}{|\mathbf{S}_j|} \odot \mathbf{V}_j$



Magnitude constraint

Loss function that ensures the estimates' magnitudes remain close to the target value $\mathbf{V}_j$ estimated beforehand (e.g., using a DNN):

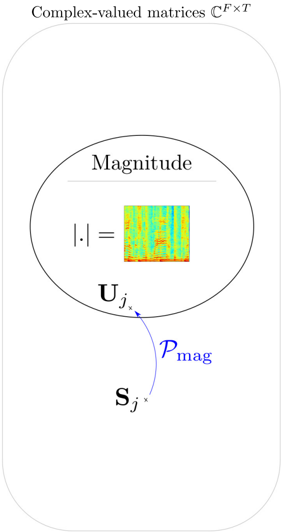
$$m(\mathbf{S}) = \sum_j |||\mathbf{S}_j| - \mathbf{V}_j||^2$$

Auxiliary function

- Auxiliary parameters $\mathbf{U}$ such that $|\mathbf{U}_j| = \mathbf{V}_j$.
- $m^+(\mathbf{S}, \mathbf{Z}) = \sum_j ||\mathbf{S}_j - \mathbf{U}_j||^2$ is an auxiliary function for m.

Auxiliary parameters update: $\mathbf{U}_j = \dfrac{\mathbf{S}_j}{|\mathbf{S}_j|} \odot \mathbf{V}_j$

- Defines a projector $\mathcal{P}_{\mathsf{mag}}$ onto the subspace of matrices whose magnitude equals the target value.



Algorithm derivation example: problem setting

Main problem: optimize the mixing objective + soft consistency penalty + hard magnitude constraint.

$$\min_{\mathbf{S}} h(\mathbf{S}) + \sigma i(\mathbf{S}) \text{ such that } |\mathbf{S}_j| = \mathbf{V}_j$$

Algorithm derivation example: problem setting

Main problem: optimize the mixing objective + soft consistency penalty + hard magnitude constraint.

$$\min_{\mathbf{S}} h(\mathbf{S}) + \sigma i(\mathbf{S}) \text{ such that } |\mathbf{S}_j| = \mathbf{V}_j$$

Using our auxiliary function framework, this rewrites:

$$\min_{\mathbf{S},\mathbf{Y},\mathbf{Z}} h^+(\mathbf{S},\mathbf{Y}) + \sigma i^+(\mathbf{S},\mathbf{Z}) \quad \text{such that} \quad \begin{cases} |\mathbf{S}_j| = \mathbf{V}_j \\ \sum_j \mathbf{Y}_j = \mathbf{X} \\ \mathbf{Z}_j \in \mathrm{Im}(\mathsf{STFT}) \end{cases}$$

Algorithm derivation example: problem setting

Main problem: optimize the mixing objective + soft consistency penalty + hard magnitude constraint.

$$\min_{\mathbf{S}} h(\mathbf{S}) + \sigma i(\mathbf{S}) \text{ such that } |\mathbf{S}_j| = \mathbf{V}_j$$

Using our auxiliary function framework, this rewrites:

$$\min_{\mathbf{S},\mathbf{Y},\mathbf{Z}} h^+(\mathbf{S}, \mathbf{Y}) + \sigma i^+(\mathbf{S}, \mathbf{Z}) \quad \text{such that} \quad \begin{cases} |\mathbf{S}_j| = \mathbf{V}_j \\ \sum_j \mathbf{Y}_j = \mathbf{X} \\ \mathbf{Z}_j \in \text{Im(STFT)} \end{cases}$$

- ▷ Auxiliary parameters updates ($\mathbf{Y}$ and $\mathbf{Z}$) are already known.

Algorithm derivation example: problem setting

Main problem: optimize the mixing objective + soft consistency penalty + hard magnitude constraint.

$$\min_{\mathbf{S}} h(\mathbf{S}) + \sigma i(\mathbf{S}) \text{ such that } |\mathbf{S}_j| = \mathbf{V}_j$$

Using our auxiliary function framework, this rewrites:

$$\min_{\mathbf{S},\mathbf{Y},\mathbf{Z}} h^+(\mathbf{S},\mathbf{Y}) + \sigma i^+(\mathbf{S},\mathbf{Z}) \quad \text{such that} \quad \begin{cases} |\mathbf{S}_j| = \mathbf{V}_j \\ \sum_j \mathbf{Y}_j = \mathbf{X} \\ \mathbf{Z}_j \in \mathrm{Im}(\mathrm{STFT}) \end{cases}$$

- ▷ Auxiliary parameters updates ($\mathbf{Y}$ and $\mathbf{Z}$) are already known.
- ▷ So let's focus on the update on $\mathbf{S}$.

Algorithm derivation example: update

New problem

- Incorporate the hard constraint using the method of Lagrange multipliers.
- Find a critical point for:

$$h^+(\mathbf{S}, \mathbf{Y}) + \sigma i^+(\mathbf{S}, \mathbf{Z}) + \sum_{j,f,t} \delta_{j,f,t}(|s_{j,f,t}|^2 - v_{j,f,t}^2)$$

Algorithm derivation example: update

New problem

- Incorporate the hard constraint using the method of Lagrange multipliers.
- Find a critical point for:

$$h^+(\mathbf{S}, \mathbf{Y}) + \sigma i^+(\mathbf{S}, \mathbf{Z}) + \sum_{j,f,t} \delta_{j,f,t}(|s_{j,f,t}|^2 - v_{j,f,t}^2)$$

Update

- Set the partial derivative with respect to $\mathbf{S}$ at 0 and solve:

$$\mathbf{S}_j = \frac{\mathbf{Y}_j + \sigma \mathbf{\Lambda}_j \odot \mathbf{Z}_j}{|\mathbf{Y}_j + \sigma \mathbf{\Lambda}_j \odot \mathbf{Z}_j|} \odot \mathbf{V}_j$$

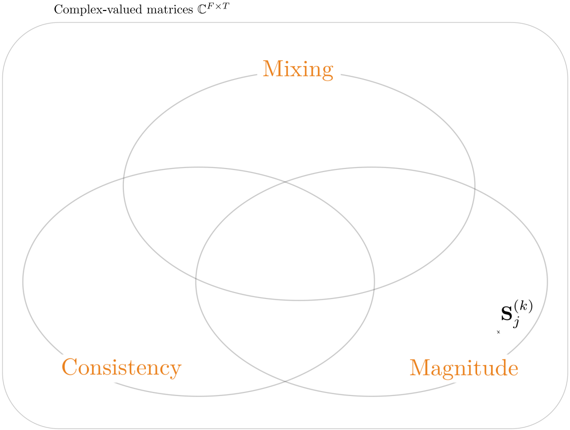
- Generalizes particular cases from the literature ($\sigma = 0$ and $\sigma = +\infty$).

Algorithm derivation example: illustration

Compact update rule using the projectors: $\mathcal{P}_{\mathsf{mag}}\left(\mathcal{P}_{\mathsf{mix}}(\mathbf{S}) + \sigma \mathbf{\Lambda} \odot \mathcal{P}_{\mathsf{cons}}(\mathbf{S})\right)$

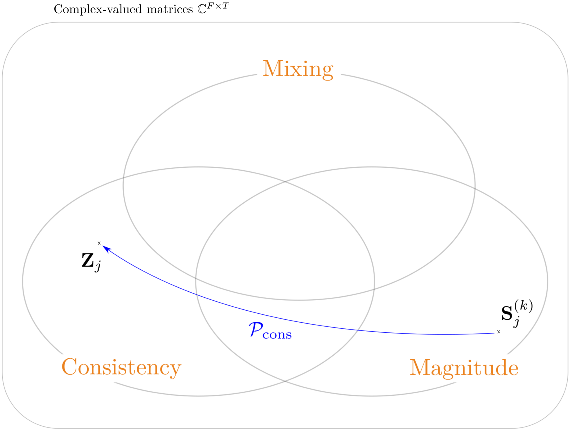
Algorithm derivation example: illustration

Compact update rule using the projectors: $\mathcal{P}_{\text{mag}}\left(\mathcal{P}_{\text{mix}}(\mathbf{S}) + \sigma\boldsymbol{\Lambda} \odot \mathcal{P}_{\text{cons}}(\mathbf{S})\right)$



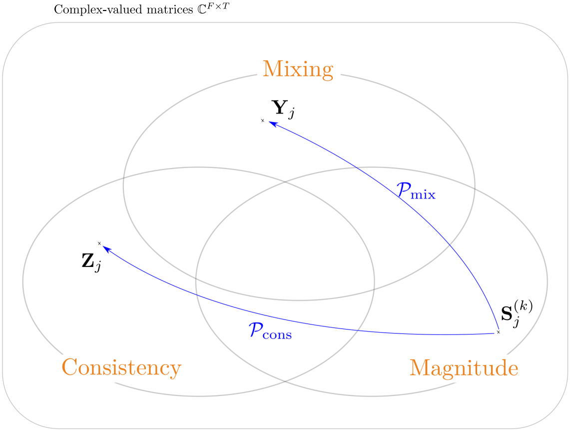
Algorithm derivation example: illustration

Compact update rule using the projectors: $\mathcal{P}_{\mathsf{mag}}\left(\mathcal{P}_{\mathsf{mix}}(\mathbf{S}) + \sigma\mathbf{\Lambda} \odot \mathcal{P}_{\mathsf{cons}}(\mathbf{S})\right)$



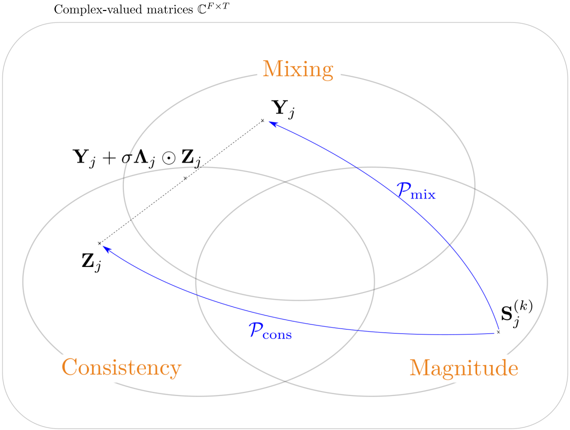
Algorithm derivation example: illustration

Compact update rule using the projectors: $\mathcal{P}_{\mathsf{mag}}\left(\mathcal{P}_{\mathsf{mix}}(\mathbf{S}) + \sigma\boldsymbol{\Lambda} \odot \mathcal{P}_{\mathsf{cons}}(\mathbf{S})\right)$



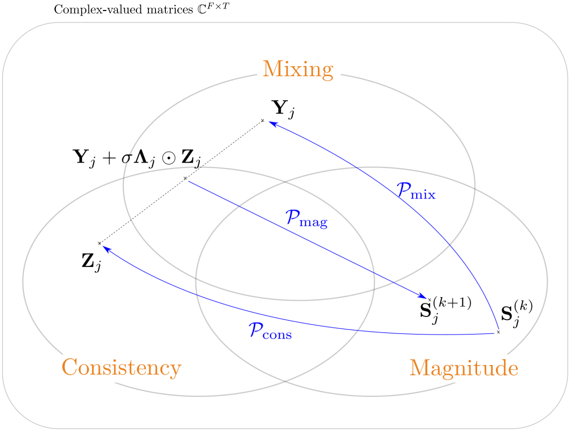
Algorithm derivation example: illustration

Compact update rule using the projectors: $\mathcal{P}_{\mathsf{mag}}\left(\mathcal{P}_{\mathsf{mix}}(\mathbf{S}) + \sigma\mathbf{\Lambda} \odot \mathcal{P}_{\mathsf{cons}}(\mathbf{S})\right)$



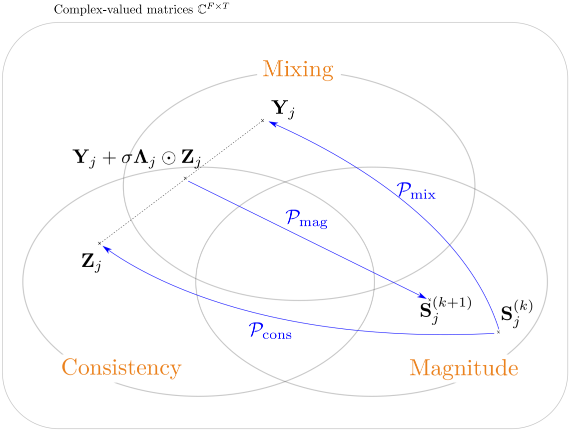
Algorithm derivation example: illustration

Compact update rule using the projectors: $\mathcal{P}_{\mathsf{mag}}\left(\mathcal{P}_{\mathsf{mix}}(\mathbf{S}) + \sigma\boldsymbol{\Lambda} \odot \mathcal{P}_{\mathsf{cons}}(\mathbf{S})\right)$



Algorithm derivation example: illustration

Compact update rule using the projectors: $\mathcal{P}_{\mathsf{mag}}\left(\mathcal{P}_{\mathsf{mix}}(\mathbf{S}) + \sigma \mathbf{\Lambda} \odot \mathcal{P}_{\mathsf{cons}}(\mathbf{S})\right)$



Check our EUSIPCO paper for all problem formulations / update schemes.

Experiments

Task: speech enhancement

- ▷ Clean speech (VoiceBank) + noise (DEMAND: living room, bus, and public square noises).
- ▷ Magnitudes are estimated beforehand using Open-Unmix.

Experiments

Task: speech enhancement

- ▷ Clean speech (VoiceBank) + noise (DEMAND: living room, bus, and public square noises).
- ▷ Magnitudes are estimated beforehand using Open-Unmix.

Separation quality (signal-to-distortion ratio):

	Accurate magnitudes	Less accurate magnitudes
Baseline (`MISI`)	**19.6**	7.7
Proposed (1)	**19.6**	7.7
Proposed (2)	**19.6**	7.5
Proposed (3)	19.3	**8.1**

Experiments

Task: speech enhancement

- ▷ Clean speech (VoiceBank) + noise (DEMAND: living room, bus, and public square noises).
- ▷ Magnitudes are estimated beforehand using Open-Unmix.

Separation quality (signal-to-distortion ratio):

	Accurate magnitudes	Less accurate magnitudes
Baseline (`MISI`)	**19.6**	7.7
Proposed (1)	**19.6**	7.7
Proposed (2)	**19.6**	7.5
Proposed (3)	19.3	**8.1**

- ▷ Some novel algorithms are interesting alternatives.
- ▷ Perspectives: unfold these into neural networks for time-domain training.

Conclusion

Current trends

From nonnegative to time-domain deep learning.

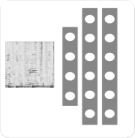
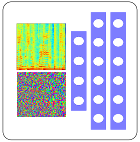
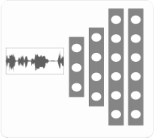
✓ Performance in controlled conditions, no more phase problem.

Current trends

From nonnegative to time-domain deep learning.

- ✓ Performance in controlled conditions, no more phase problem.
- ✗ Greediness in (annotated) training data.
- ✗ Lacks interpretability and flexibility.

Current trends

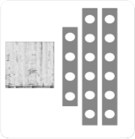
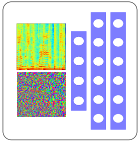
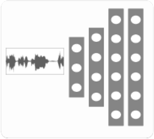
From nonnegative to time-domain deep learning.

- ✓ Performance in controlled conditions, no more phase problem.
- ✗ Greediness in (annotated) training data.
- ✗ Lacks interpretability and flexibility.

... and then back to STFT-domain deep learning.

- ✓ Robustness/flexibility of time-frequency processing.
- ✓ Performance of processing all the data exhaustively.

Current trends

From nonnegative to time-domain deep learning.

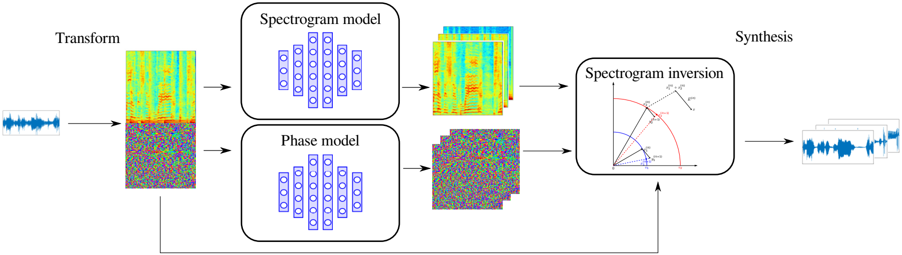
- ✓ Performance in controlled conditions, no more phase problem.
- ✗ Greediness in (annotated) training data.
- ✗ Lacks interpretability and flexibility.

... and then back to STFT-domain deep learning.

- ✓ Robustness/flexibility of time-frequency processing.
- ✓ Performance of processing all the data exhaustively.
- ✗ Using a real/imaginary part decomposition of the STFT is sub-optimal.

The proposed alternative

- ▷ The room for improvement of phase recovery: more potential gain than with magnitudes.
- ▷ Move towards **deep phase recovery** for increased performance.



Work in progress:

- ▷ Design deep phase prior models.
- ▷ Unfold iterative algorithms into neural networks for time-domain separation.

Thanks!

https://magronp.github.io/

https://github.com/magronp/