Source Separation Tutorial Mini-Series III: Extensions and Interpretations to Non-Negative Matrix Factorization

Nicholas Bryan
Dennis Sun

Center for Computer Research in Music and Acoustics,
Stanford University

DSP Seminar
April 9th, 2013

Roadmap of Talk

Roadmap of Talk

Non-Negative Matrix Factorization

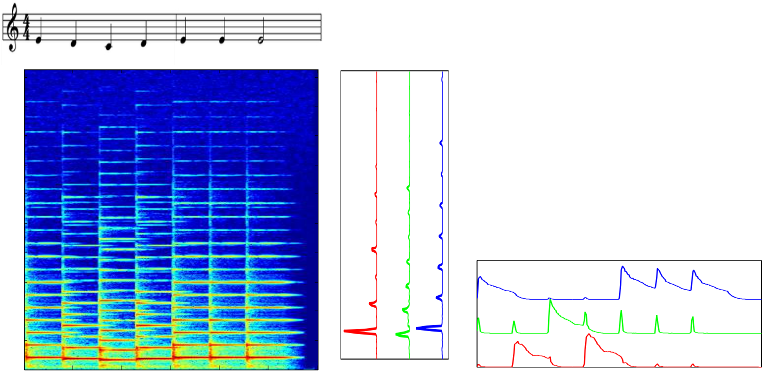
$$\overset{\text{Data}}{\begin{bmatrix} \quad \mathbf{V} \quad \end{bmatrix}} \approx \overset{\text{Basis Vectors}}{\begin{bmatrix} \mathbf{W} \end{bmatrix}} \overset{\text{Weights}}{\begin{bmatrix} \quad \mathbf{H} \quad \end{bmatrix}}$$

Non-Negative Matrix Factorization

$$\begin{array}{ccccc} \text{Data} & & \text{Basis Vectors} & \text{Weights} \\ \begin{bmatrix} & & \\ & \mathbf{V} & \\ & & \end{bmatrix} & \approx & \begin{bmatrix} \\ \mathbf{W} \\ \\ \end{bmatrix} & \begin{bmatrix} & & \\ & \mathbf{H} & \\ & & \end{bmatrix} \end{array}$$

- A matrix factorization where everything is non-negative
- $\mathbf{V} \in \mathrm{R}_+^{F \times T}$ - original non-negative data
- $\mathbf{W} \in \mathrm{R}_+^{F \times K}$ - matrix of basis vectors, dictionary elements
- $\mathbf{H} \in \mathrm{R}_+^{K \times T}$ - matrix of activations, weights, or gains
- $K < F < T$ (typically)
  - A compressed representation of the data
  - A low-rank approximation to $\mathbf{V}$

NMF With Spectrogram Data



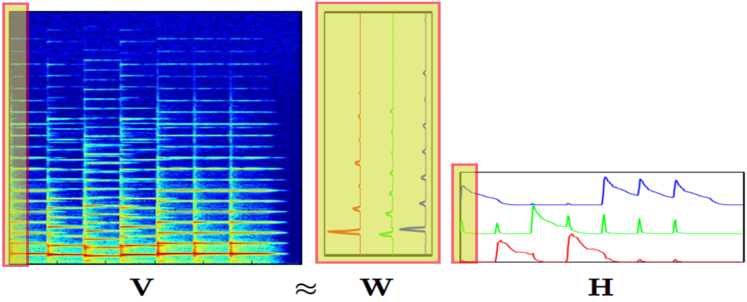
$\mathbf{V} \approx \mathbf{W} \quad \mathbf{H}$

NMF of *Mary Had a Little Lamb* with $K = 3$ play stop

- The basis vectors capture prototypical spectra [SB03]
- The weights capture the gain of the basis vectors

Factorization Interpretation I

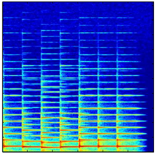
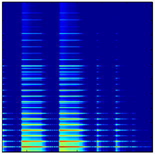
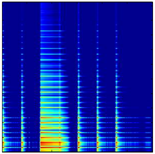
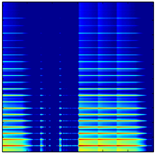
Columns of $\mathbf{V} \approx$ as a weighted sum (mixture) of basis vectors



$$\begin{bmatrix} | & | & & | \\ \mathbf{v}_1 & \mathbf{v}_2 & \dots & \mathbf{v}_T \\ | & | & & | \end{bmatrix} \approx \begin{bmatrix} \sum_{j=1}^{K} \mathbf{H}_{j1}\, \mathbf{w}_j & \sum_{j=1}^{K} \mathbf{H}_{j2}\, \mathbf{w}_j & \dots & \sum_{j=1}^{K} \mathbf{H}_{jT}\, \mathbf{w}_j \end{bmatrix}$$

Factorization Interpretation II

$\mathbf{V}$ is approximated as sum of matrix "layers"

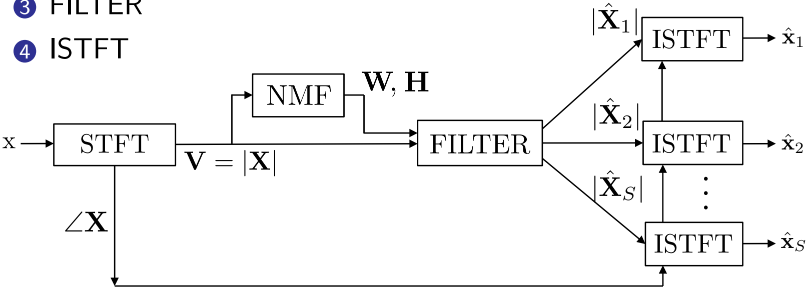
 $=$  $+$  $+$ 

$$\begin{bmatrix} | & | & & | \\ \mathbf{v}_1 & \mathbf{v}_2 & \dots & \mathbf{v}_T \\ | & | & & | \end{bmatrix} \approx \begin{bmatrix} | & | & & | \\ \mathbf{w}_1 & \mathbf{w}_2 & \dots & \mathbf{w}_K \\ | & | & & | \end{bmatrix} \begin{bmatrix} — & \mathbf{h}_1^{\mathrm{T}} & — \\ — & \mathbf{h}_2^{\mathrm{T}} & — \\ & \vdots & \\ — & \mathbf{h}_K^{\mathrm{T}} & — \end{bmatrix}$$

$$\mathbf{V} \approx \mathbf{w}_1\,\mathbf{h}_1^{\mathrm{T}} + \mathbf{w}_2\,\mathbf{h}_2^{\mathrm{T}} + \ldots + \mathbf{w}_K\,\mathbf{h}_K^{\mathrm{T}}$$

General Separation Pipeline

1. STFT
2. NMF
3. FILTER
4. ISTFT



An Algorithm for NMF

Algorithm KL-NMF

initialize $\mathbf{W}, \mathbf{H}$

repeat

$$\mathbf{H} \leftarrow \mathbf{H} \mathbin{.*} \frac{\mathbf{W}^T \frac{\mathbf{V}}{\mathbf{W}\,\mathbf{H}}}{\mathbf{W}^T \mathbf{1}}$$

$$\mathbf{W} \leftarrow \mathbf{W} \mathbin{.*} \frac{\frac{\mathbf{V}}{\mathbf{W}\,\mathbf{H}} \mathbf{H}^T}{\mathbf{1}\,\mathbf{H}^T}$$

until convergence **return** $\mathbf{W}, \mathbf{H}$

Roadmap of Talk

1. Review
2. Further Insight
3. Supervised and Semi-Supervised Separation
4. Probabilistic Interpretation
5. Extensions
6. Evaluation
7. Future Research Directions
8. Matlab

Non-Negativity

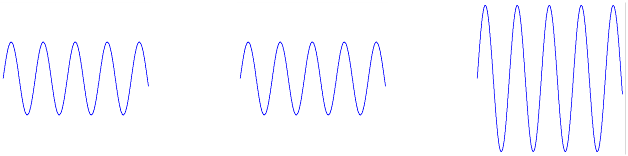
- Question: Why do we get a 'parts-based' representation of sound?

Non-Negativity

- Question: Why do we get a 'parts-based' representation of sound?
- Answer: Non-negativity avoids destructive interference

Constructive and Destructive Interference

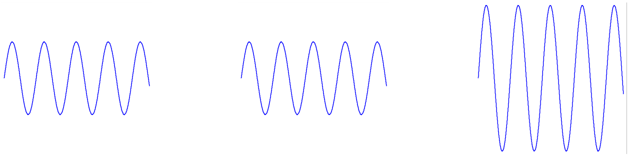
Constructive Interference



$$\mathbf{x} \quad + \quad \mathbf{x} \quad = \quad 2\mathbf{x}$$

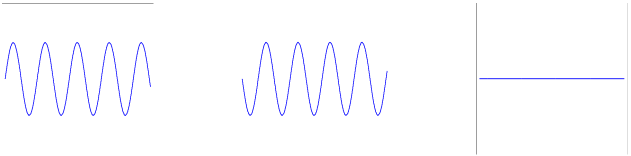
Constructive and Destructive Interference

Constructive Interference



$$\mathbf{x} \quad + \quad \mathbf{x} \quad = \quad 2\mathbf{x}$$

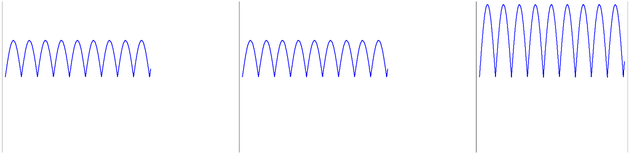
Destructive Interference



$$\mathbf{x} \quad + \quad (-\mathbf{x}) \quad = \quad \mathbf{0}$$

Non-Negative Constructive and Destructive Interference

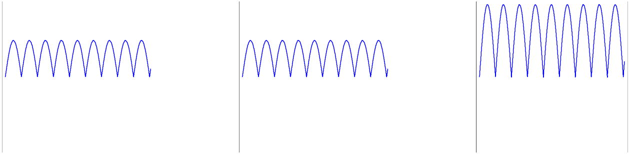
Constructive Interference



$$|\mathbf{x}| \quad + \quad |\mathbf{x}| \quad = \quad 2|\mathbf{x}|$$

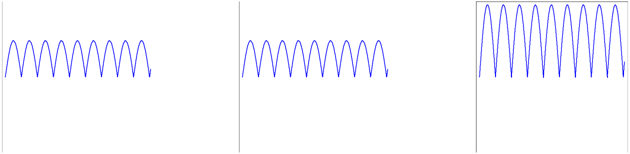
Non-Negative Constructive and Destructive Interference

Constructive Interference



$$|\mathbf{x}| \quad + \quad |\mathbf{x}| \quad = \quad 2|\mathbf{x}|$$

Destructive Interference



$$|\mathbf{x}| \quad + \quad |-\mathbf{x}| \quad = \quad 2|\mathbf{x}|$$

Non-negativity Avoids Destructive Interference

- With non-negativity, destructive interference cannot happen

Non-negativity Avoids Destructive Interference

- With non-negativity, destructive interference cannot happen
- Everything must cumulatively add to explain the original data

Non-negativity Avoids Destructive Interference

- With non-negativity, destructive interference cannot happen
- Everything must cumulatively add to explain the original data
- But . . .

Approximation I

In doing so, we violate the superposition property of sound

$$\mathbf{x} = \mathbf{x}_1 + \mathbf{x}_2 + \ldots + \mathbf{x}_N$$

and actually solve

$$|\mathbf{X}| \approx |\mathbf{X}_1| + |\mathbf{X}_2| + \ldots + |\mathbf{X}_N|$$

Approximation II

Alternatively, we can see this approximation via:

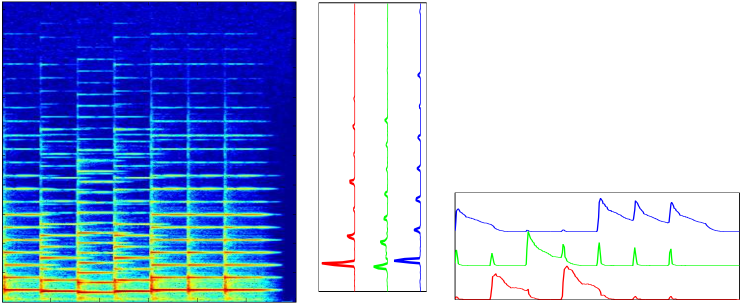
$$\begin{aligned}
\mathbf{x} &= \mathbf{x}_1 + \mathbf{x}_2 + \ldots + \mathbf{x}_N \\
|\mathbf{X}|\, e^{j\phi} &= |\mathbf{X}_1|\, e^{j\phi_1} + |\mathbf{X}_2|\, e^{j\phi_2} + \ldots + |\mathbf{X}_N|\, e^{j\phi_N} \\
|\mathbf{X}|\, e^{j\phi} &\approx \left(|\mathbf{X}_1| + |\mathbf{X}_2| + \ldots + |\mathbf{X}_N|\right) e^{j\phi} \\
|\mathbf{X}| &\approx |\mathbf{X}_1| + |\mathbf{X}_2| + \ldots + |\mathbf{X}_N|
\end{aligned}$$

Roadmap of Talk

1. Review
2. Further Insight
3. Supervised and Semi-Supervised Separation
4. Probabilistic Interpretation
5. Extensions
6. Evaluation
7. Future Research Directions
8. Matlab

Unsupervised Separation I

Single, simultaneously estimation of $\mathbf{W}$ and $\mathbf{H}$ from a mixture $\mathbf{V}$



$$\mathbf{V} \approx \mathbf{W} \quad \mathbf{H}$$

What we've seen so far

Unsupervised Separation II

- Complex sounds need more than one basis vector

Unsupervised Separation II

- Complex sounds need more than one basis vector
- Difficult to control which basis vector explain which source

Unsupervised Separation II

- Complex sounds need more than one basis vector
- Difficult to control which basis vector explain which source
- No way to control the factorization other than F, T, and K

Supervised Separation

General idea:

1. Use isolated training data of each source within a mixture to pre-learn individual models of each source [SRS07]

Supervised Separation

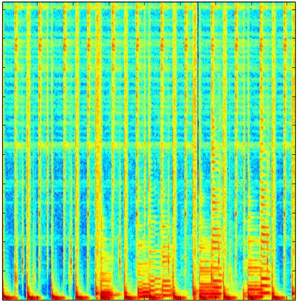
General idea:

1. Use isolated training data of each source within a mixture to pre-learn individual models of each source [SRS07]

2. Given a mixture, use the pre-learned models for separation

Supervised Separation I

Example:

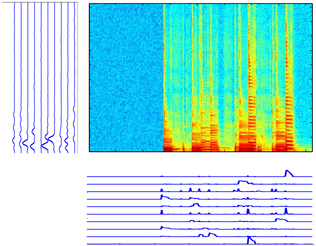


Drum and Bass Loop play stop

Supervised Separation II

Use isolated training data to learn factorization for each source

Bass Loop play stop



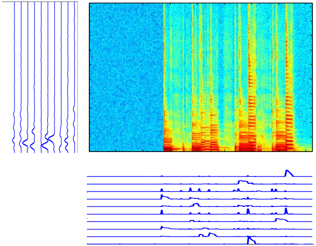
$$\mathbf{V}_1 \approx \mathbf{W}_1 \, \mathbf{H}_1$$

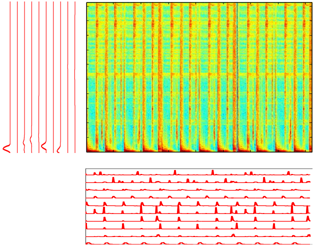
Supervised Separation II

Use isolated training data to learn factorization for each source

Bass Loop play stop

Drum Loop play stop





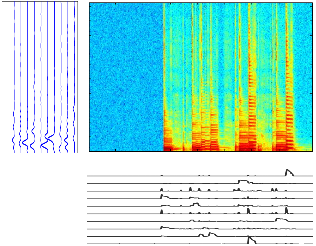
$$\mathbf{V}_1 \approx \mathbf{W}_1 \, \mathbf{H}_1$$

$$\mathbf{V}_2 \approx \mathbf{W}_2 \, \mathbf{H}_2$$

Supervised Separation III

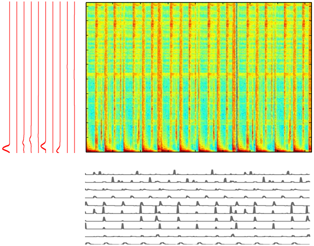
Throw away the activations $\mathbf{H}_1$ and $\mathbf{H}_2$

Bass Loop play stop



$\mathbf{V}_1 \approx \mathbf{W}_1 \, \mathbf{H}_1$

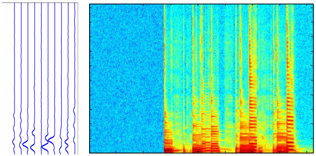
Drum Loop play stop

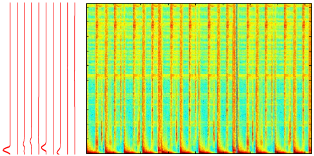


$\mathbf{V}_2 \approx \mathbf{W}_2 \, \mathbf{H}_2$

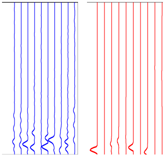
Supervised Separation IV

Concatenate basis vectors of each source for complete dictionary



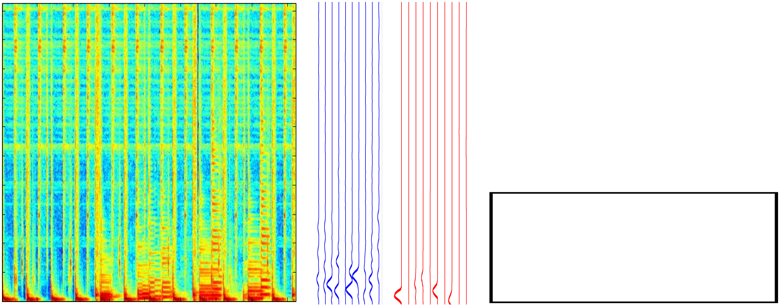


$$\mathbf{W} \approx \begin{bmatrix}\mathbf{W}_1 & \mathbf{W}_2\end{bmatrix} =$$



Supervised Separation V

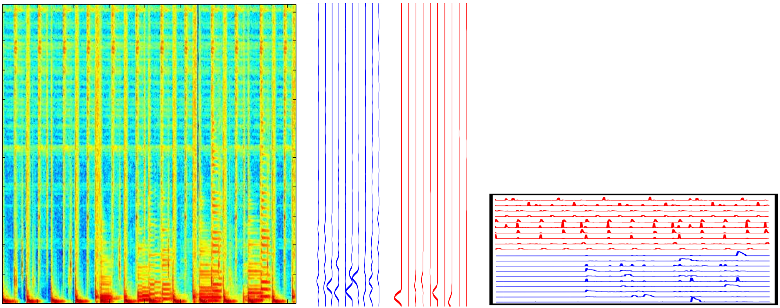
Now, factorize the mixture with $\mathbf{W}$ fixed (only estimate $\mathbf{H}$)



$$
\begin{aligned}
\mathbf{V} \quad &\approx \quad \mathbf{W} \qquad \mathbf{H} \\
&\approx \begin{bmatrix} \mathbf{W}_1 & \mathbf{W}_2 \end{bmatrix} \begin{bmatrix} \mathbf{H}_1^{\mathrm{T}} \\ \mathbf{H}_2^{\mathrm{T}} \end{bmatrix}
\end{aligned}
$$

Supervised Separation V

Now, factorize the mixture with $\mathbf{W}$ fixed (only estimate $\mathbf{H}$)



$$\mathbf{V} \approx \mathbf{W} \mathbf{H}$$

$$\approx \begin{bmatrix} \mathbf{W}_1 & \mathbf{W}_2 \end{bmatrix} \begin{bmatrix} \mathbf{H}_1^{\mathrm{T}} \\ \mathbf{H}_2^{\mathrm{T}} \end{bmatrix}$$

Complete Supervised Process

1. Use isolated training data to learn a factorization $(\mathbf{W}_s \, \mathbf{H}_s)$ for each source s

Complete Supervised Process

1. Use isolated training data to learn a factorization $(\mathbf{W}_s \, \mathbf{H}_s)$ for each source s

2. Throw away activations $\mathbf{H}_s$ for each source s

Complete Supervised Process

1. Use isolated training data to learn a factorization ($\mathbf{W}_s \mathbf{H}_s$) for each source s
2. Throw away activations $\mathbf{H}_s$ for each source s
3. Concatenate basis vectors of each source ($\mathbf{W}_1, \mathbf{W}_2$,...) for complete dictionary $\mathbf{W}$

Complete Supervised Process

1. Use isolated training data to learn a factorization ($\mathbf{W}_s \mathbf{H}_s$) for each source s

2. Throw away activations $\mathbf{H}_s$ for each source s

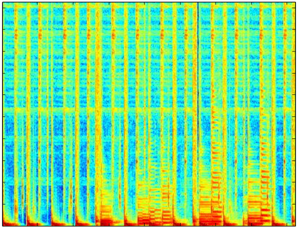
3. Concatenate basis vectors of each source ($\mathbf{W}_1, \mathbf{W}_2$,...) for complete dictionary $\mathbf{W}$

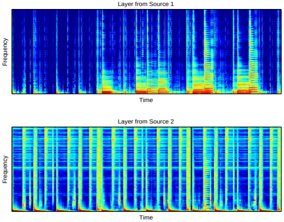
4. Hold $\mathbf{W}$ fixed, and factorize unknown mixture of sources $\mathbf{V}$ (only estimate $\mathbf{H}$)

Complete Supervised Process

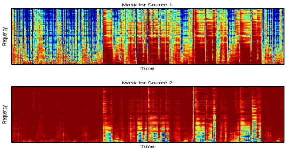
1. Use isolated training data to learn a factorization ($\mathbf{W}_s \mathbf{H}_s$) for each source s
2. Throw away activations $\mathbf{H}_s$ for each source s
3. Concatenate basis vectors of each source ($\mathbf{W}_1, \mathbf{W}_2$,...) for complete dictionary $\mathbf{W}$
4. Hold $\mathbf{W}$ fixed, and factorize unknown mixture of sources $\mathbf{V}$ (only estimate $\mathbf{H}$)
5. Once complete, use $\mathbf{W}$ and $\mathbf{H}$ as before to filter and separate each source

Sound Examples





Mixture sound (left) p s and separated drums p s and bass p s.



Masking filters used to process mixture into the separated sources.

Question

- What if you don't have isolated training data for each source?

Question

- What if you don't have isolated training data for each source?
- And unsupervised separation still doesn't work?

Semi-Supervised Separation

General Idea:

1. Learn supervised dictionaries for as many sources as you can [SRS07]

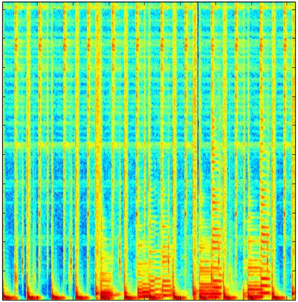
Semi-Supervised Separation

General Idea:

1. Learn supervised dictionaries for as many sources as you can [SRS07]
2. Infer remaining unknown dictionaries from the mixture (only fix certain columns of $\mathbf{W}$)

Semi-Supervised Separation I

Example:

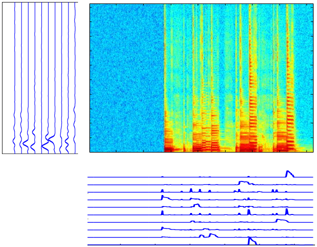


Drum and Bass Loop play stop

Semi-Supervised Separation II

Use isolated training data to learn factorization for as many sources as possible (e.g. one source)

Bass Loop play stop

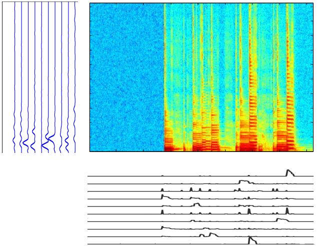


$$\mathbf{V}_1 \approx \mathbf{W}_1 \, \mathbf{H}_1$$

Semi-Supervised Separation III

Throw away the activations $\mathbf{H}_1$

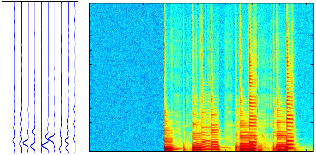
Bass Loop play stop



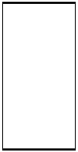
$$\mathbf{V}_1 \approx \mathbf{W}_1 \mathbf{H}_1$$

Semi-Supervised Separation IV

Concatenate *known* basis vectors with *unknown* basis vectors (initialized randomly) for complete dictionary

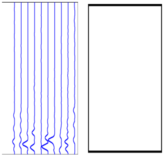


Known bass basis vectors



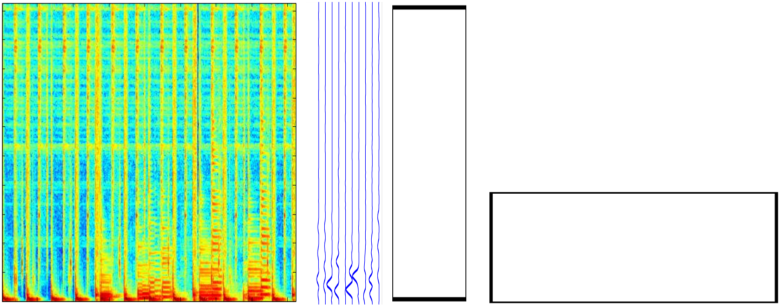
Unknown drum basis vectors (initialized randomly)

$$\mathbf{W} \approx [\mathbf{W}_1 \, \mathbf{W}_2] =$$



Semi-Supervised Separation V

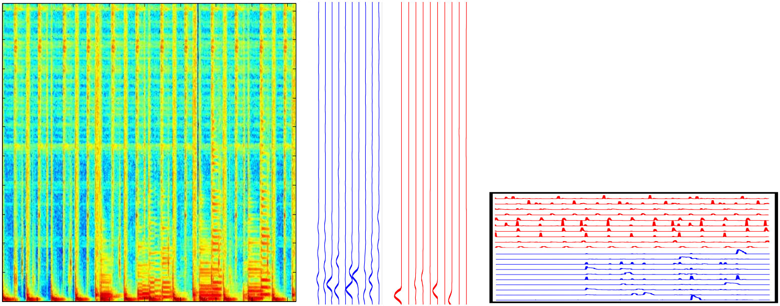
Now, factorize the mixture with $\mathbf{W}_1$ fixed (estimate $\mathbf{W}_2$ and $\mathbf{H}$)



$$\begin{aligned}\mathbf{V} &\approx \mathbf{W} \quad \mathbf{H} \\ &\approx \begin{bmatrix}\mathbf{W}_1 & \mathbf{W}_2\end{bmatrix} \begin{bmatrix}\mathbf{H}_1^{\mathrm{T}} \\ \mathbf{H}_2^{\mathrm{T}}\end{bmatrix}\end{aligned}$$

Semi-Supervised Separation V

Now, factorize the mixture with $\mathbf{W}_1$ fixed (estimate $\mathbf{W}_2$ and $\mathbf{H}$)



$$\begin{aligned}\mathbf{V} &\approx \mathbf{W}\,\mathbf{H} \\ &\approx \begin{bmatrix}\mathbf{W}_1 & \mathbf{W}_2\end{bmatrix} \begin{bmatrix}\mathbf{H}_1^{\mathrm{T}} \\ \mathbf{H}_2^{\mathrm{T}}\end{bmatrix}\end{aligned}$$

Complete Semi-Supervised Process

1. Use isolated training data to learn a factorization ($\mathbf{W}_s\,\mathbf{H}_s$) for as many sources s as possible

Complete Semi-Supervised Process

1. Use isolated training data to learn a factorization $(\mathbf{W}_s\,\mathbf{H}_s)$ for as many sources s as possible

2. Throw away activations $\mathbf{H}_s$ for each known source s

Complete Semi-Supervised Process

1. Use isolated training data to learn a factorization ($\mathbf{W}_s \, \mathbf{H}_s$) for as many sources s as possible

2. Throw away activations $\mathbf{H}_s$ for each known source s

3. Concatenate known basis vectors with random init vectors for unknown sources to construct complete dictionary $\mathbf{W}$

Complete Semi-Supervised Process

1. Use isolated training data to learn a factorization ($\mathbf{W}_s \mathbf{H}_s$) for as many sources s as possible

2. Throw away activations $\mathbf{H}_s$ for each known source s

3. Concatenate known basis vectors with random init vectors for unknown sources to construct complete dictionary $\mathbf{W}$

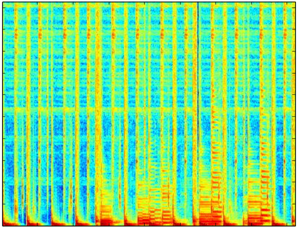
4. Hold the columns of $\mathbf{W}$ fixed which correspond to known sources, and factorize a mixture $\mathbf{V}$ (estimate $\mathbf{H}$ and any known column of $\mathbf{W}$)

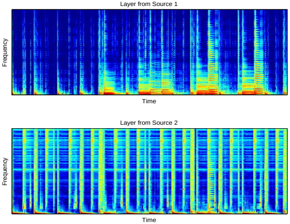
Complete Semi-Supervised Process

1. Use isolated training data to learn a factorization $(\mathbf{W}_s\,\mathbf{H}_s)$ for as many sources s as possible
2. Throw away activations $\mathbf{H}_s$ for each known source s
3. Concatenate known basis vectors with random init vectors for unknown sources to construct complete dictionary $\mathbf{W}$
4. Hold the columns of $\mathbf{W}$ fixed which correspond to known sources, and factorize a mixture $\mathbf{V}$ (estimate $\mathbf{H}$ and any known column of $\mathbf{W}$)
5. Once complete, use $\mathbf{W}$ and $\mathbf{H}$ as before to filter and separate each source

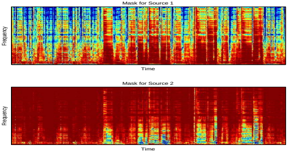
Sound Examples

Supervised the bass.





Mixture sound (left) p s and separated drums p s and bass p s.



Masking filters used to process mixture into the separated sources.

Roadmap of Talk

Probabilistic Interpretation

Some notation:

z indexes basis vectors, f frequency bins, and t time frames.

Probabilistic Interpretation

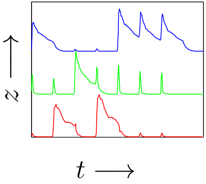
Some notation:

z indexes basis vectors, f frequency bins, and t time frames.

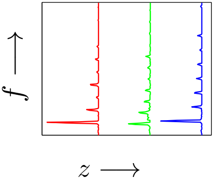
The model:

For each time frame t, repeat the following:

- Choose a component from $p(z|t)$.



- Choose a frequency from $p(f|z)$.



Probabilistic Interpretation

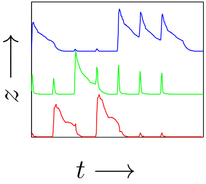
Some notation:

z indexes basis vectors, f frequency bins, and t time frames.

The model:

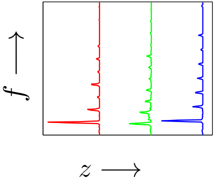
For each time frame t, repeat the following:

- Choose a component from $p(z|t)$.



$= H$

- Choose a frequency from $p(f|z)$.



Probabilistic Interpretation

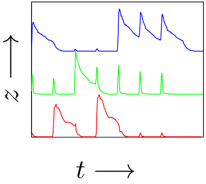
Some notation:

z indexes basis vectors, f frequency bins, and t time frames.

The model:

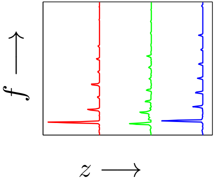
For each time frame t, repeat the following:

- Choose a component from $p(z|t)$.



$= H$

- Choose a frequency from $p(f|z)$.



$= W$

Probabilistic Interpretation

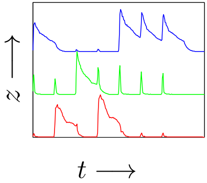
Some notation:

z indexes basis vectors, f frequency bins, and t time frames.

The model:

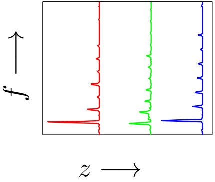
For each time frame t, repeat the following:

- Choose a component from $p(z|t)$.



$= H$

- Choose a frequency from $p(f|z)$.



$= W$

The spectrogram V_{ft} are the counts that we obtain at the end of the day. We want to estimate $p(z|t)$ and $p(f|z)$.

Probabilistic Interpretation

Is this realistic?

- We're assuming the spectrogram contains counts. We sample "quanta" of spectral energy at a time.

Probabilistic Interpretation

Is this realistic?

- We're assuming the spectrogram contains counts. We sample "quanta" of spectral energy at a time.
- This model is popular in topic modeling, where we assume documents are generated from first sampling a topic from $p(z|d)$ and then a word from $p(w|z)$.

Probabilistic Interpretation

Is this realistic?

- We're assuming the spectrogram contains counts. We sample "quanta" of spectral energy at a time.
- This model is popular in topic modeling, where we assume documents are generated from first sampling a topic from $p(z|d)$ and then a word from $p(w|z)$.
  - probabilistic latent semantic indexing, or pLSI [Hof99]

Probabilistic Interpretation

Is this realistic?

- We're assuming the spectrogram contains counts. We sample "quanta" of spectral energy at a time.
- This model is popular in topic modeling, where we assume documents are generated from first sampling a topic from $p(z|d)$ and then a word from $p(w|z)$.
  - probabilistic latent semantic indexing, or pLSI [Hof99]
  - latent Dirichlet allocation, or LDA [BNJ03]

Probabilistic Interpretation

Is this realistic?

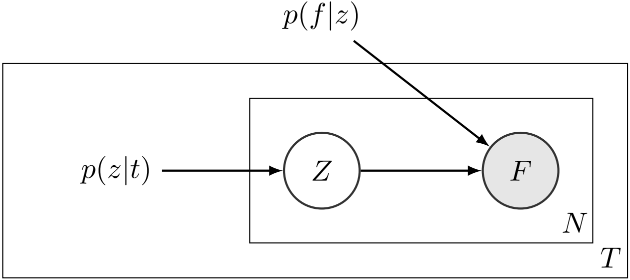
- We're assuming the spectrogram contains counts. We sample "quanta" of spectral energy at a time.
- This model is popular in topic modeling, where we assume documents are generated from first sampling a topic from $p(z|d)$ and then a word from $p(w|z)$.
  - probabilistic latent semantic indexing, or pLSI [Hof99]
  - latent Dirichlet allocation, or LDA [BNJ03]
- In audio, this model is called probabilistic latent component analysis, or PLCA [SRS06]

Latent Variable Model

We only observe the outcomes V_{ft}. But the full model involves unobserved variables Z.

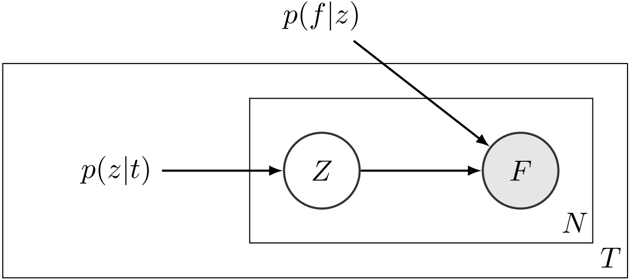
Latent Variable Model

We only observe the outcomes V_{ft}. But the full model involves unobserved variables Z.



Latent Variable Model

We only observe the outcomes V_{ft}. But the full model involves unobserved variables Z.



The **Expectation-Maximization (EM) algorithm** is used to fit latent variable models. It is also used in estimating Hidden Markov Models, Gaussian mixture models, etc.

Maximum Likelihood Estimation

To fit the parameters, we choose the parameters that maximize the likelihood of the data. Let's zoom in on a single time frame:

$$p(v_1, ..., v_F) = \frac{(\sum_f v_f)!}{v_1!...v_F!} \prod_{f=1}^{F} p(f|t)^{v_f}$$

Maximum Likelihood Estimation

To fit the parameters, we choose the parameters that maximize the likelihood of the data. Let's zoom in on a single time frame:

$$p(v_1, ..., v_F) = \frac{(\sum_f v_f)!}{v_1!...v_F!} \prod_{f=1}^{F} p(f|t)^{v_f}$$

According to the model on the previous slide, the frequency could have come from any of the latent components. We don't observe this so we average over all of them.

$$p(f|t) = \sum_z p(z|t)p(f|z)$$

Maximum Likelihood Estimation

To fit the parameters, we choose the parameters that maximize the likelihood of the data. Let's zoom in on a single time frame:

$$p(v_1, ..., v_F) = \frac{(\sum_f v_f)!}{v_1!...v_F!} \prod_{f=1}^{F} p(f|t)^{v_f}$$

According to the model on the previous slide, the frequency could have come from any of the latent components. We don't observe this so we average over all of them.

$$p(f|t) = \sum_z p(z|t)p(f|z)$$

Putting it all together, we obtain:

$$p(v_1, ..., v_F) = \frac{(\sum_f v_f)!}{v_1!...v_F!} \prod_{f=1}^{F} \left(\sum_z p(z|t)p(f|z) \right)^{v_f}$$

Maximum Likelihood Estimation

$$p(v_1, ..., v_F) = \frac{(\sum_f v_f)!}{v_1!...v_F!} \prod_{f=1}^{F} \left(\sum_z p(z|t)p(f|z) \right)^{v_f}$$

- We want to maximize this over $p(z|t)$ and $p(f|z)$.

Maximum Likelihood Estimation

$$p(v_1, ..., v_F) = \frac{(\sum_f v_f)!}{v_1!...v_F!} \prod_{f=1}^{F} \left(\sum_z p(z|t)p(f|z) \right)^{v_f}$$

- We want to maximize this over $p(z|t)$ and $p(f|z)$.
- In general, with probabilities it is easier to maximize the log than the thing itself:

$$\log p(v_1, ..., v_F) = \sum_{f=1}^{F} v_f \log \left(\sum_z p(z|t)p(f|z) \right) + \text{const.}$$

Maximum Likelihood Estimation

$$p(v_1, ..., v_F) = \frac{(\sum_f v_f)!}{v_1!...v_F!} \prod_{f=1}^{F} \left(\sum_z p(z|t)p(f|z) \right)^{v_f}$$

- We want to maximize this over $p(z|t)$ and $p(f|z)$.
- In general, with probabilities it is easier to maximize the log than the thing itself:

$$\log p(v_1, ..., v_F) = \sum_{f=1}^{F} v_f \log \left(\sum_z p(z|t)p(f|z) \right) + \text{const.}$$

- **Remember from last week:** First thing you should always try is differentiate and set equal to zero. Does this work here?

The Connection to NMF

- Last week, we talked about minimizing the KL divergence between V and WH.

$$D(V||WH) = -\sum_{f,t} V_{ft} \log\left(\sum_z W_{fz} H_{zt}\right) + \sum_{f,t}\sum_z W_{fz} H_{zt} + \text{const.}$$

The Connection to NMF

- Last week, we talked about minimizing the KL divergence between V and WH.

$$D(V||WH) = -\sum_{f,t} V_{ft} \log \left(\sum_z W_{fz} H_{zt} \right) + \sum_{f,t} \sum_z W_{fz} H_{zt} + \text{const.}$$

- Compare with maximizing the log-likelihood:

$$\log p(v_1, ..., v_F) = \sum_{f=1}^{F} v_f \log \left(\sum_z p(z|t) p(f|z) \right) + \text{const.}$$

The Connection to NMF

- Last week, we talked about minimizing the KL divergence between V and WH.

$$D(V||WH) = -\sum_{f,t} V_{ft} \log\left(\sum_z W_{fz}H_{zt}\right) + \sum_{f,t}\sum_z W_{fz}H_{zt} + \text{const.}$$

- Compare with maximizing the log-likelihood:

$$\log p(v_1, ..., v_F) = \sum_{f=1}^{F} v_f \log\left(\sum_z p(z|t)p(f|z)\right) + \text{const.}$$

subject to $\sum_z p(z|t) = 1$ and $\sum_f p(f|z) = 1$.

The Connection to NMF

- Last week, we talked about minimizing the KL divergence between V and WH.

$$D(V||WH) = -\sum_{f,t} V_{ft} \log\left(\sum_z W_{fz}H_{zt}\right) + \sum_{f,t}\sum_z W_{fz}H_{zt} + \text{const.}$$

- Compare with maximizing the log-likelihood:

$$\log p(v_1, ..., v_F) = \sum_{f=1}^{F} v_f \log\left(\sum_z p(z|t)p(f|z)\right) + \text{const.}$$

subject to $\sum_z p(z|t) = 1$ and $\sum_f p(f|z) = 1$.

- Last week, we used majorization-minimization on $D(V||WH)$:

$$-\log\left(\sum_z \phi_{ftz}\frac{W_{fz}H_{zt}}{\phi_{ftz}}\right) \leq -\sum_z \phi_{ftz}\log\frac{W_{fz}H_{zt}}{\phi_{ftz}}$$

The Connection to NMF

- Last week, we talked about minimizing the KL divergence between V and WH.

$$D(V||WH) = -\sum_{f,t} V_{ft} \log\left(\sum_z W_{fz} H_{zt}\right) + \sum_{f,t}\sum_z W_{fz} H_{zt} + \text{const.}$$

- Compare with maximizing the log-likelihood:

$$\log p(v_1, \ldots, v_F) = \sum_{f=1}^{F} v_f \log\left(\sum_z p(z|t) p(f|z)\right) + \text{const.}$$

subject to $\sum_z p(z|t) = 1$ and $\sum_f p(f|z) = 1$.

- Last week, we used majorization-minimization on $D(V||WH)$:

$$-\log\left(\sum_z \phi_{ftz} \frac{W_{fz} H_{zt}}{\phi_{ftz}}\right) \leq -\sum_z \phi_{ftz} \log \frac{W_{fz} H_{zt}}{\phi_{ftz}}$$

- Now watch what we do with the log-likelihood....

EM Algorithm

- Suppose we observed the latent component for a frequency quanta. Then we wouldn't need to average over the components; its log-likelihood would be:

$$\log p(z|t)p(f|z)$$

EM Algorithm

- Suppose we observed the latent component for a frequency quanta. Then we wouldn't need to average over the components; its log-likelihood would be:

$$\log p(z|t)p(f|z)$$

- But we don't know the latent component, so let's average this over our best guess of the probability of each component:

$$\sum_z p(z|f,t) \log p(z|t)p(f|z)$$

EM Algorithm

- Suppose we observed the latent component for a frequency quanta. Then we wouldn't need to average over the components; its log-likelihood would be:

$$\log p(z|t)p(f|z)$$

- But we don't know the latent component, so let's average this over our best guess of the probability of each component:

$$\sum_z p(z|f,t) \log p(z|t)p(f|z)$$

- In summary, we've replaced

$$\log \left(\sum_z p(z|t)p(f|z) \right) \quad \text{by} \quad \sum_z p(z|f,t) \log p(z|t)p(f|z)$$

Look familiar?

EM Algorithm

E-step: Calculate

$$p(z|f,t) = \frac{p(z|t)p(f|z)}{\sum_z p(z|t)p(f|z)}$$

M-step: Maximize

$$\sum_{f,t} V_{ft} \sum_z p(z|f,t) \log p(z|t)p(f|z)$$

EM Algorithm

E-step: Calculate

$$p(z|f,t) = \frac{p(z|t)p(f|z)}{\sum_z p(z|t)p(f|z)}$$

M-step: Maximize

$$\sum_{f,t} V_{ft} \sum_z p(z|f,t) \log p(z|t)p(f|z)$$

Majorization: Calculate

$$\phi_{ftz} = \frac{W_{fz}H_{zt}}{\sum_z W_{fz}H_{zt}}$$

Minimization: Minimize

$$-\sum_{f,t} V_{ft} \sum_z \phi_{zft} \log W_{fz}H_{zt} + \sum_{f,t,z} W_{fz}H_{zt}$$

EM Algorithm

E-step: Calculate

$$p(z|f,t) = \frac{p(z|t)p(f|z)}{\sum_z p(z|t)p(f|z)}$$

M-step: Maximize

$$\sum_{f,t} V_{ft} \sum_z p(z|f,t) \log p(z|t)p(f|z)$$

Majorization: Calculate

$$\phi_{ftz} = \frac{W_{fz}H_{zt}}{\sum_z W_{fz}H_{zt}}$$

Minimization: Minimize

$$-\sum_{f,t} V_{ft} \sum_z \phi_{zft} \log W_{fz}H_{zt} + \sum_{f,t,z} W_{fz}H_{zt}$$

The EM updates are exactly the multiplicative updates for NMF, up to normalization!

EM Algorithm

E-step: Calculate

$$p(z|f,t) = \frac{p(z|t)p(f|z)}{\sum_z p(z|t)p(f|z)}$$

M-step: Maximize

$$\sum_{f,t} V_{ft} \sum_z p(z|f,t) \log p(z|t)p(f|z)$$

Majorization: Calculate

$$\phi_{ftz} = \frac{W_{fz}H_{zt}}{\sum_z W_{fz}H_{zt}}$$

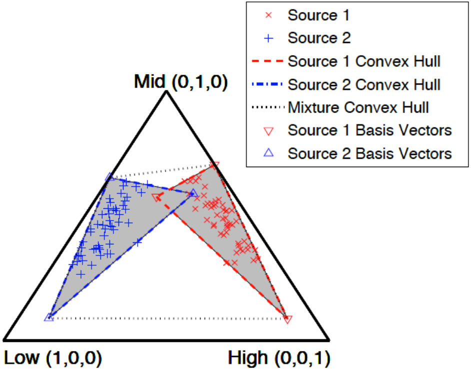
Minimization: Minimize

$$-\sum_{f,t} V_{ft} \sum_z \phi_{zft} \log W_{fz}H_{zt} + \sum_{f,t,z} W_{fz}H_{zt}$$

The EM updates are exactly the multiplicative updates for NMF, up to normalization!

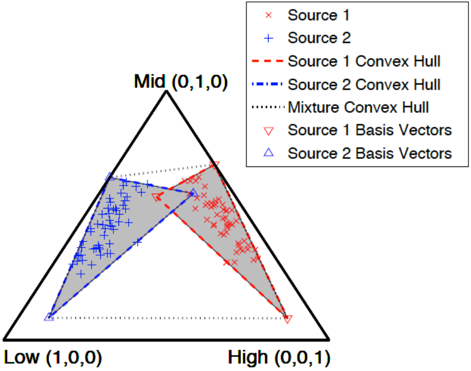
The EM algorithm is a special case of MM, where the minorizing function is the expected conditional log likelihood.

Geometric Interpretation



- We can think of the basis vectors $p(f|z)$ as lying on a probability simplex.

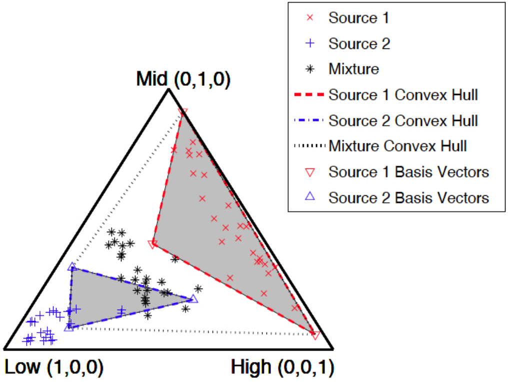
Geometric Interpretation



- We can think of the basis vectors $p(f|z)$ as lying on a probability simplex.
- The possible sounds for a given source is the convex hull of the basis vectors for that source.

Geometric Interpretation

In supervised separation, we try to explain time frames of the mixture signal as combinations of the basis vectors of the different sources.



Roadmap of Talk

1. Review
2. Further Insight
3. Supervised and Semi-Supervised Separation
4. Probabilistic Interpretation
5. Extensions
6. Evaluation
7. Future Research Directions
8. Matlab

Extensions

- The number of parameters that need to be estimated is huge: $FK + KT$.

Extensions

- The number of parameters that need to be estimated is huge: $FK + KT$.
- In high-dimensional settings, it is useful to impose additional structure.

Extensions

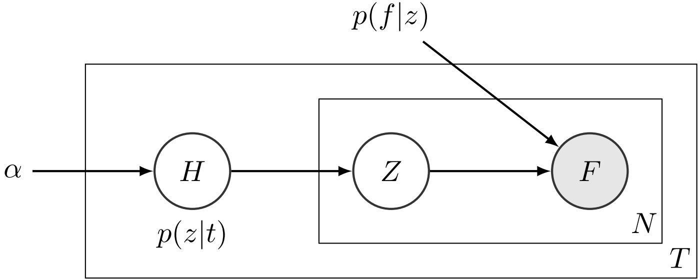
- The number of parameters that need to be estimated is huge: $FK + KT$.
- In high-dimensional settings, it is useful to impose additional structure.
- We will look at two ways to do this: **priors** and **regularization**.

Priors

- Assume the parameters are also random, e.g., $H = p(z|t)$ is generated from $p(H|\alpha)$. This is called a **prior** distribution.

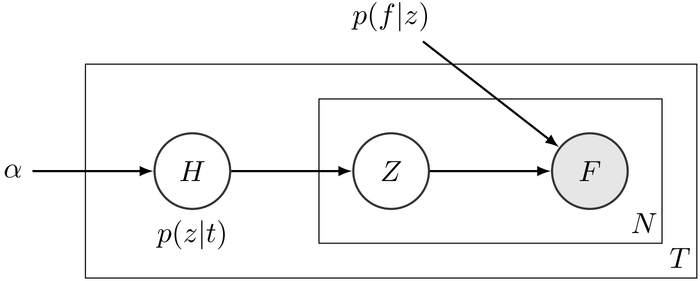
Priors

- Assume the parameters are also random, e.g., $H = p(z|t)$ is generated from $p(H|\alpha)$. This is called a **prior** distribution.



Priors

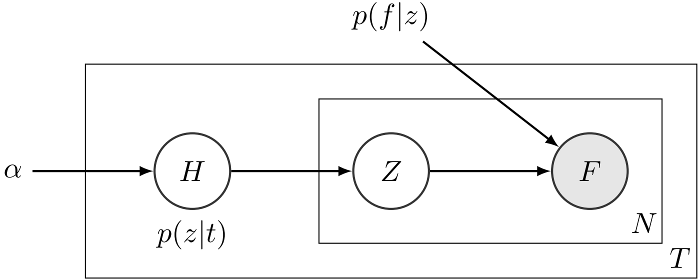
- Assume the parameters are also random, e.g., $H = p(z|t)$ is generated from $p(H|\alpha)$. This is called a **prior** distribution.



- Estimate the **posterior** distribution $p(H|\alpha, V)$.

Priors

- Assume the parameters are also random, e.g., $H = p(z|t)$ is generated from $p(H|\alpha)$. This is called a **prior** distribution.



- Estimate the **posterior** distribution $p(H|\alpha, V)$.
- **Bayes' rule**: $p(H|\alpha, V) = \dfrac{p(H, V|\alpha)}{p(V|\alpha)} = \dfrac{p(H|\alpha)p(V|H)}{p(V|\alpha)}$

Bayesian Inference

- Bayes' rule gives us an entire distribution over $H = p(z|t)$.

Bayesian Inference

- Bayes' rule gives us an entire distribution over $H = p(z|t)$.
- One option is the **posterior mean**: computationally intractable.

Bayesian Inference

- Bayes' rule gives us an entire distribution over $H = p(z|t)$.
- One option is the **posterior mean**: computationally intractable.
- An easier option is the **posterior mode** (MAP):

$$\underset{H}{\text{maximize}}\ \log p(H|\alpha, V) = \log p(H|\alpha) + \log p(V|H) - p(V|\alpha)$$

Bayesian Inference

- Bayes' rule gives us an entire distribution over $H = p(z|t)$.
- One option is the **posterior mean**: computationally intractable.
- An easier option is the **posterior mode** (MAP):

$$\underset{H}{\text{maximize}} \; \log p(H|\alpha, V) = \underbrace{\log p(H|\alpha)}_{\text{log prior}} + \underbrace{\log p(V|H)}_{\text{likelihood}} - \cancel{p(V|\alpha)}$$

Bayesian Inference

- Bayes' rule gives us an entire distribution over $H = p(z|t)$.
- One option is the **posterior mean**: computationally intractable.
- An easier option is the **posterior mode** (MAP):

$$\underset{H}{\text{maximize}} \; \log p(H|\alpha, V) = \underbrace{\log p(H|\alpha)}_{\text{log prior}} + \underbrace{\log p(V|H)}_{\text{likelihood}} - \cancel{p(V|\alpha)}$$

- We can choose priors that encode structural assumptions, like sparsity.

Regularization Viewpoint

- Another way is to add another term to the objective function:

$$\underset{W,H\geq 0}{\text{minimize}}\; D(V||WH) + \lambda\Omega(H)$$

Ω encodes the desired structure, λ controls the strength.

Regularization Viewpoint

- Another way is to add another term to the objective function:

$$\underset{W,H\geq 0}{\text{minimize}}\; D(V||WH) + \lambda\Omega(H)$$

  Ω encodes the desired structure, λ controls the strength.

- We showed earlier that $D(V||WH)$ is the negative log likelihood. So:

$$\lambda\Omega(H) \iff -\log p(H|\alpha)$$

Regularization Viewpoint

- Another way is to add another term to the objective function:

$$\underset{W,H \geq 0}{\text{minimize}}\ D(V||WH) + \lambda\Omega(H)$$

  Ω encodes the desired structure, λ controls the strength.

- We showed earlier that $D(V||WH)$ is the negative log likelihood. So:

$$\lambda\Omega(H) \Longleftrightarrow -\log p(H|\alpha)$$

- Some common choices for $\Omega(H)$:

Regularization Viewpoint

- Another way is to add another term to the objective function:

$$\underset{W,H\geq 0}{\text{minimize}}\; D(V||WH) + \lambda\Omega(H)$$

  Ω encodes the desired structure, λ controls the strength.

- We showed earlier that $D(V||WH)$ is the negative log likelihood. So:

$$\lambda\Omega(H) \Longleftrightarrow -\log p(H|\alpha)$$

- Some common choices for $\Omega(H)$:
  - sparsity: $||H||_1 = \sum_{z,t} |H_{zt}|$

Regularization Viewpoint

- Another way is to add another term to the objective function:

$$\underset{W,H\geq 0}{\text{minimize}}\ D(V||WH) + \lambda\Omega(H)$$

  Ω encodes the desired structure, λ controls the strength.
- We showed earlier that $D(V||WH)$ is the negative log likelihood. So:

$$\lambda\Omega(H) \iff -\log p(H|\alpha)$$

- Some common choices for $\Omega(H)$:
  - sparsity: $||H||_1 = \sum_{z,t} |H_{zt}|$
  - smoothness: $\sum_{z,t}(H_{z,t} - H_{z,t-1})^2$

Roadmap of Talk

1. Review
2. Further Insight
3. Supervised and Semi-Supervised Separation
4. Probabilistic Interpretation
5. Extensions
6. Evaluation
7. Future Research Directions
8. Matlab

Evaluation Measures

- Signal-to-Interference Ratio (SIR)
- Signal-to-Artifact Ratio (SAR)
- Signal-to-Distortion Ratio (SDR)

We want all of these metrics to be as high as possible [VGF06]

Evaluation Measures

To compute these three measures, we must obtain:

- $\mathbf{s} \in R^{T \times N}$ original unmixed signals (ground truth)
- $\hat{\mathbf{s}} \in R^{T \times N}$ estimated separated sources

Then, we decompose these signals into

- s_{target} — actual source estimate
- e_{interf} — interference signal (i.e. the unwanted source)
- e_{artif} — artifacts of the separation algorithm

Evaluation Measures

To compute s_{target}, e_{interf}, and e_{artif}

- $s_{target} = P_{s_j}\hat{s}_j$
- $e_{interf} = P_{\mathbf{s}}\hat{s}_j - P_{s_j}\hat{s}_j$
- $e_{artif} = \hat{s}_j - P_{\mathbf{s}}\hat{s}_j$

where P_{s_j} and $P_{\mathbf{s}}$ are $T \times T$ projection matrices

Signal-to-Interference Ratio (SIR)

A measure of the suppression of the unwanted source

$$\mathsf{SIR} = 10 \log_{10} \frac{||s_{target}||^2}{||e_{interf}||^2}$$

Signal-to-Artifact Ratio (SAR)

A measure of the artifacts that have been introduced by the separation process

$$\mathsf{SAR} = 10 \log_{10} \frac{||s_{target} + e_{interf}||^2}{||e_{artif}||^2}$$

Signal-to-Distortion Ratio (SDR)

An overall measure that takes into account both the SIR and SAR

$$\text{SDR} = 10 \log_{10} \frac{||s_{target}||^2}{||e_{artif} + e_{interf}||^2}$$

Selecting Hyperparameters using BSS Eval Metrics

- One problem with NMF is the need to specify the number of basis vectors K.

Selecting Hyperparameters using BSS Eval Metrics

- One problem with NMF is the need to specify the number of basis vectors K.
- Even more parameters if you include regularization.

Selecting Hyperparameters using BSS Eval Metrics

- One problem with NMF is the need to specify the number of basis vectors K.
- Even more parameters if you include regularization.
- BSS eval metrics give us a way to learn the optimal settings for source separation.

Selecting Hyperparameters using BSS Eval Metrics

- One problem with NMF is the need to specify the number of basis vectors K.
- Even more parameters if you include regularization.
- BSS eval metrics give us a way to learn the optimal settings for source separation.
- Generate synthetic mixtures, try different parameter settings, and choose the parameters that give the best BSS eval metrics.

BSS Eval Toolbox

A Matlab tool box for source separation evaluation [VGF06]:

`http://bass-db.gforge.inria.fr/bss_eval/`

Roadmap of Talk

1. Review
2. Further Insight
3. Supervised and Semi-Supervised Separation
4. Probabilistic Interpretation
5. Extensions
6. Evaluation
7. Future Research Directions
8. Matlab

Research Directions

- Score-informed separation - sheet music
- Interactive separation - user-interaction
- Temporal dynamics - how sounds change over time
- Unsupervised separation - grouping basis vectors, clustering
- Phase estimation - complex NMF, STFT constraints, etc.
- Universal models - big data for general models of sources

Demos

- Universal Speech Models
- Interactive Source Separation
  - Drums + Bass
  - Guitar + Vocals + AutoTune
  - Jackson 5 Remixed

STFT

```
x1  = wavread('bass');
x2  = wavread('drums');
[xm fs] = wavread('drums+bass');
FFTSIZE = 1024;
HOPSIZE = 256;
WINDOWSIZE = 512;

X1 = myspectrogram(x1,FFTSIZE,fs,hann(WINDOWSIZE),-HOPSIZE);
V1 = abs(X1(1:(FFTSIZE/2+1),:));
X2 = myspectrogram(x2,FFTSIZE,fs,hann(WINDOWSIZE),-HOPSIZE);
V2 = abs(X2(1:(FFTSIZE/2+1),:));
Xm = myspectrogram(xm,FFTSIZE,fs,hann(WINDOWSIZE),-HOPSIZE);
Vm = abs(Xm(1:(FFTSIZE/2+1),:)); maxV = max(max(db(Vm)));

F = size(Vm,1);
T = size(Vm,2);
```

- https://ccrma.stanford.edu/~jos/sasp/Matlab_listing_myspectrogram_m.html
- https://ccrma.stanford.edu/~jos/sasp/Matlab_listing_invmyspectrogram_m.html

NMF

```
K = [25 25]; % number of basis vectors
MAXITER = 500; % total number of iterations to run
[W1, H1] = nmf(V1, K(1), [], MAXITER,[]);
[W2, H2] = nmf(V2, K(2), [], MAXITER,[]);
[W, H]   = nmf(Vm, K, [W1 W2], MAXITER, 1:sum(K));

function [W, H] = nmf(V, K, W, MAXITER, fixedInds)
F = size(V,1); T = size(V,2);
rand('seed',0)
if isempty(W)
    W = 1+rand(F, sum(K));
end
H = 1+rand(sum(K), T);
inds = setdiff(1:sum(K),fixedInds);
ONES = ones(F,T);
for i=1:MAXITER
    % update activations
    H = H .* (W'*( V./(W*H+eps))) ./ (W'*ONES);
    % update dictionaries
    W(:,inds) = W(:,inds) .* ((V./(W*H+eps))*H(inds,:)') ./(ONES*H(inds,:)');
end
% normalize W to sum to 1
sumW = sum(W);
W = W*diag(1./sumW);
H = diag(sumW)*H;
```

FILTER & ISTFT

```
% get the mixture phase
phi = angle(Xm);
c = [1 cumsum(K)];
for i=1:length(K)
    % create masking filter
    Mask =  W(:,c(i):c(i+1))*H(c(i):c(i+1),:)./(W*H);
    % filter
    XmagHat = Vm.*Mask;
    % create upper half of frequency before istft
    XmagHat = [XmagHat; conj( XmagHat(end-1:-1:2,:))];
    % Multiply with phase
    XHat = XmagHat.*exp(1i*phi);
    % create upper half of frequency before istft
    xhat(:,i) = real(invmyspectrogram(XmagHat.*exp(1i*phi)
end
```

References I

- David M. Blei, Andrew Y. Ng, and Michael I. Jordan, *Latent dirichlet allocation*, J. Mach. Learn. Res. **3** (2003), 993–1022.
- T. Hofmann, *Probabilistic latent semantic indexing*, Proceedings of the 22nd annual international ACM SIGIR Conference on Research and Development in Information Retrieval (New York, NY, USA), SIGIR '99, ACM, 1999, pp. 50–57.
- P. Smaragdis and J.C. Brown, *Non-negative matrix factorization for polyphonic music transcription*, IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA), oct. 2003, pp. 177 – 180.
- P. Smaragdis, B. Raj, and M. Shashanka, *A Probabilistic Latent Variable Model for Acoustic Modeling*, Advances in Neural Information Processing Systems (NIPS), Workshop on Advances in Modeling for Acoustic Processing, 2006.

References II

——— , *Supervised and semi-supervised separation of sounds from single-channel mixtures*, International Conference on Independent Component Analysis and Signal Separation (Berlin, Heidelberg), Springer-Verlag, 2007, pp. 414–421.

E. Vincent, R. Gribonval, and C. Fevotte, *Performance measurement in blind audio source separation*, IEEE Transactions on Audio, Speech, and Language Processing **14** (2006), no. 4, 1462 –1469.