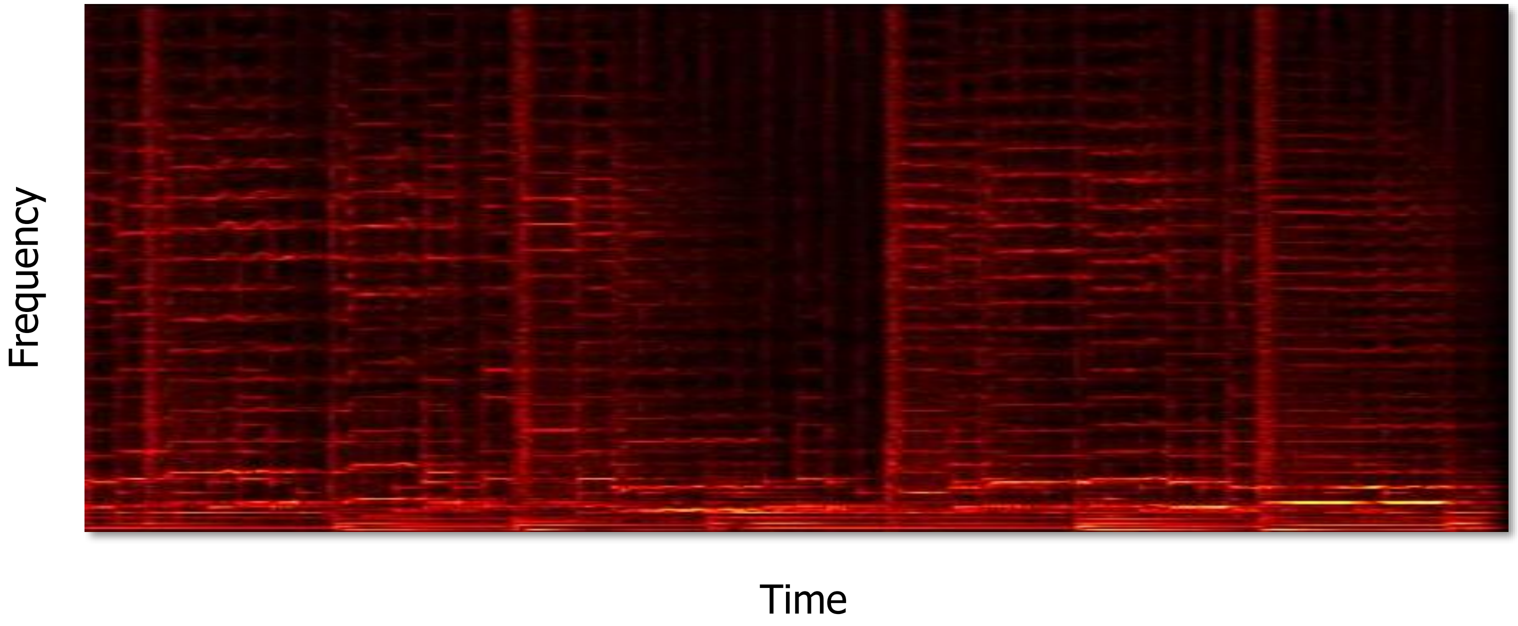
Topic 7

Audio Modeling by Non-negative Matrix Factorization

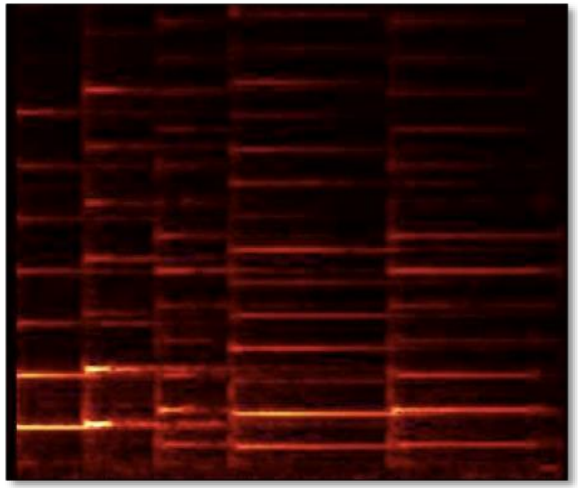
(Some slides are adapted from Gautham J. Mysore's presentation)

Structure in Spectrograms

- Spectral structure
- Temporal structure



Piano Notes

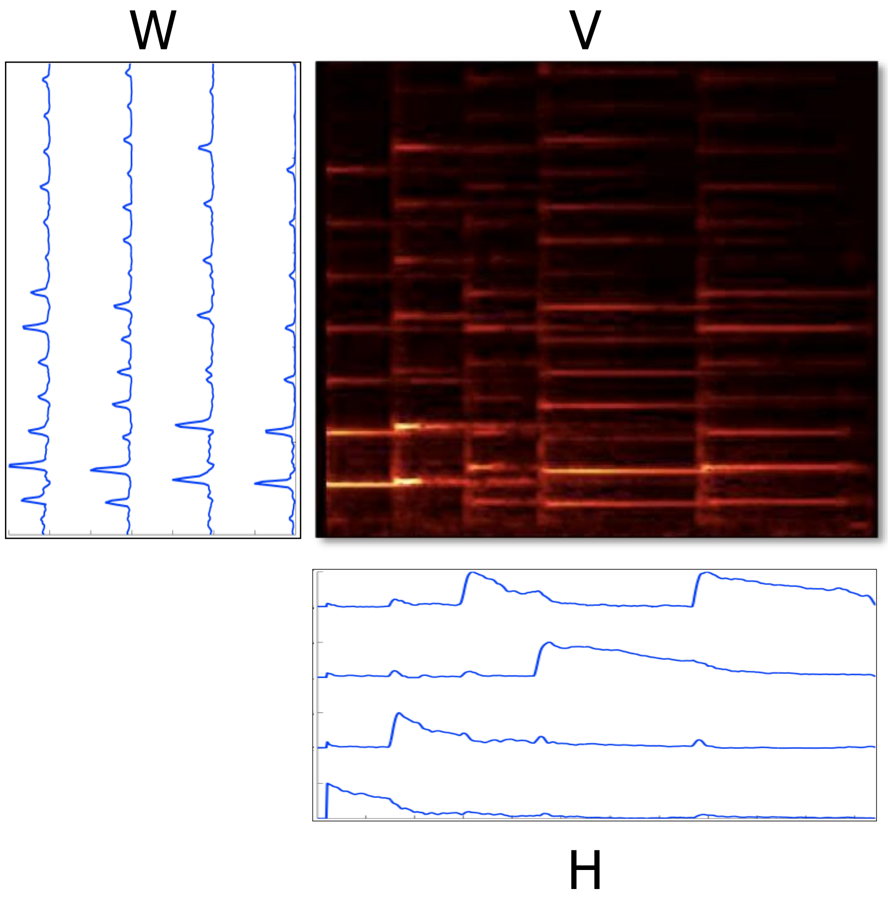


Non-negative Matrix Factorization

$$V \approx WH$$

[Lee, Seung 2001]
[Smaragdis, Brown 2003]

Dictionary Elements (Building Blocks) →



← Activations of Spectral Vectors

How to measure the approximation?

- Euclidean distance (Frobenius norm)

$$D(V \parallel WH) = \|V - WH\|_F^2 = \sum_{i,j} \left(V_{ij} - (WH)_{ij}\right)^2$$

- When $V = WH$, the distance is 0.

How to measure the approximation?

- Kullback-Leibler (KL) divergence

$$D(V \parallel WH) = \sum_{i,j} \left(V_{ij} \ln \frac{V_{ij}}{(WH)_{ij}} - V_{ij} + (WH)_{ij} \right)$$

- KL divergence between two discrete probability distributions

$$D_{KL}(P \parallel Q) = \sum_{i} P(i) \ln \frac{P(i)}{Q(i)}$$

NMF

$$\min_{W,H} D(V \parallel WH)$$

where

$$V \in \mathbb{R}^{\geq 0, m \times n}$$
$$W \in \mathbb{R}^{\geq 0, m \times r}$$
$$\mathrm{H} \in \mathbb{R}^{\geq 0, r \times n}$$
$$r \leq \min\{m, n\}$$

- What is the possible rank of V?
- What is the possible rank of WH?

Singular Value Decomposition (SVD)

$$V = A\Sigma B^T$$

where

$$V \in \mathbb{R}^{m\times n}$$
$$A \in \mathbb{R}^{m\times m}$$
$$\mathrm{B} \in \mathbb{R}^{n\times n}$$

- $\Sigma \in \mathbb{R}^{m\times n}$ is a diagonal matrix with nonnegative elements.
- $\mathrm{rank}(V)$=the number of nonzero diagonal elements of Σ.

Why NMF?

- Nonnegative data
- V is an addition of some "components"

$$V = WH = \sum_{i=1}^{r} \boldsymbol{w}_i \boldsymbol{h}_i^T$$

where $W = [\boldsymbol{w}_1, \dots, \boldsymbol{w}_r]$, $H = [\boldsymbol{h}_1, \dots, \boldsymbol{h}_r]^T$

- Nonnegative components
- Easy to interpret

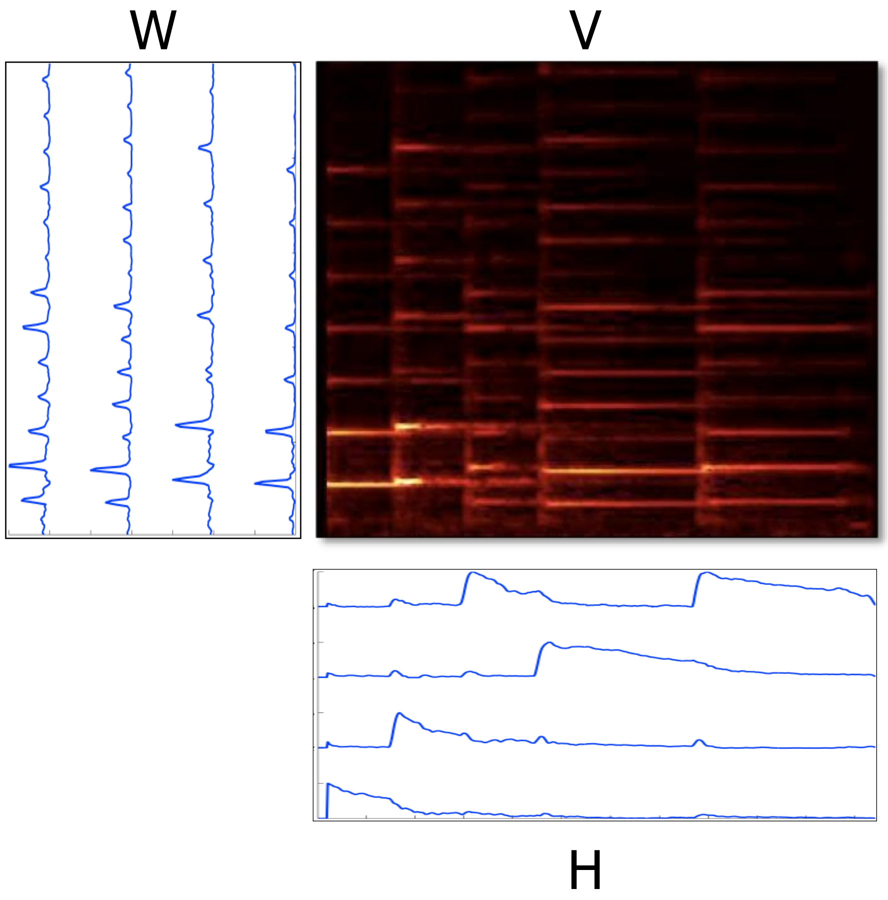
Low-rank Decomposition

- $\mathrm{rank}(V)$ = the number of nonzero diagonal elements of Σ in SVD.
- Let $\mathrm{rank}_+(V)$ be the smallest integer k for which there exists $\widehat{W} \in \mathbb{R}^{\geq 0, m\times k}$ and $\widehat{\mathrm{H}} \in \mathbb{R}^{\geq 0, k\times n}$, such that $V = \widehat{W}\widehat{H}$.
- $\mathrm{rank}(V) \leq \mathrm{rank}_+(V) \leq \min\{m, n\}$

- In NMF, we take $V \approx WH$, where $W \in \mathbb{R}^{\geq 0, m\times r}, H \in \mathbb{R}^{\geq 0, r\times n}$
- $\mathrm{rank}(WH) \leq min\{\mathrm{rank}(W), \mathrm{rank}(H))\} \leq r$
- In NMF, we use $r \ll \mathrm{rank}(V)$.

Low-rank Decomposition

$$V \approx WH$$



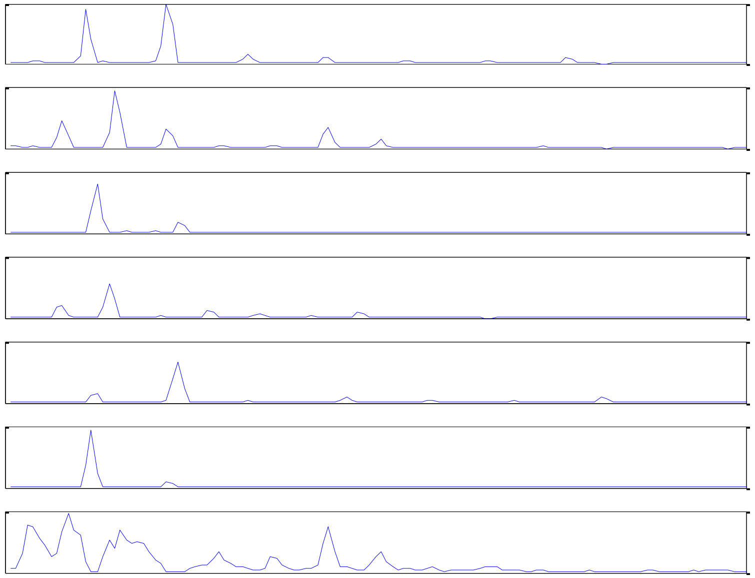
- $rank(V)$ could be pretty large (about the same size as the number of frames), since harmonics do not decay at the same rate.
- $rank(WH) = 4$
- But we get pretty good approximation.

If r is too large

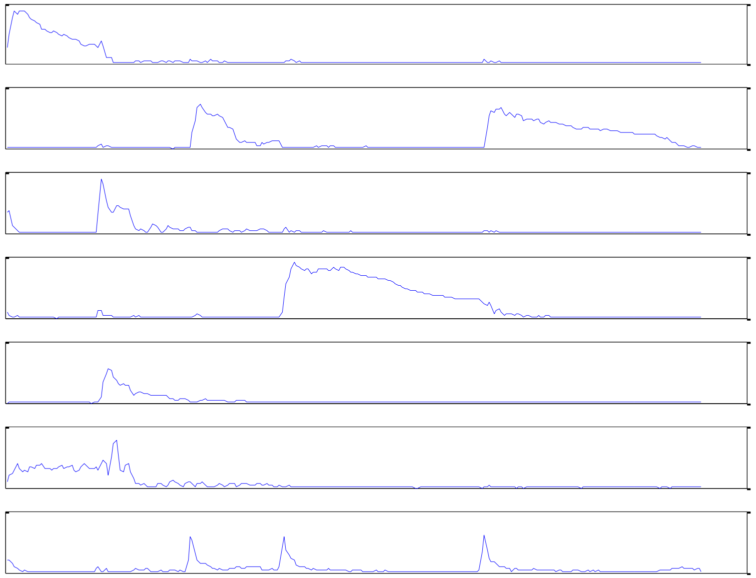
- Let $r = 7$

Original

Reconstructed



W^T



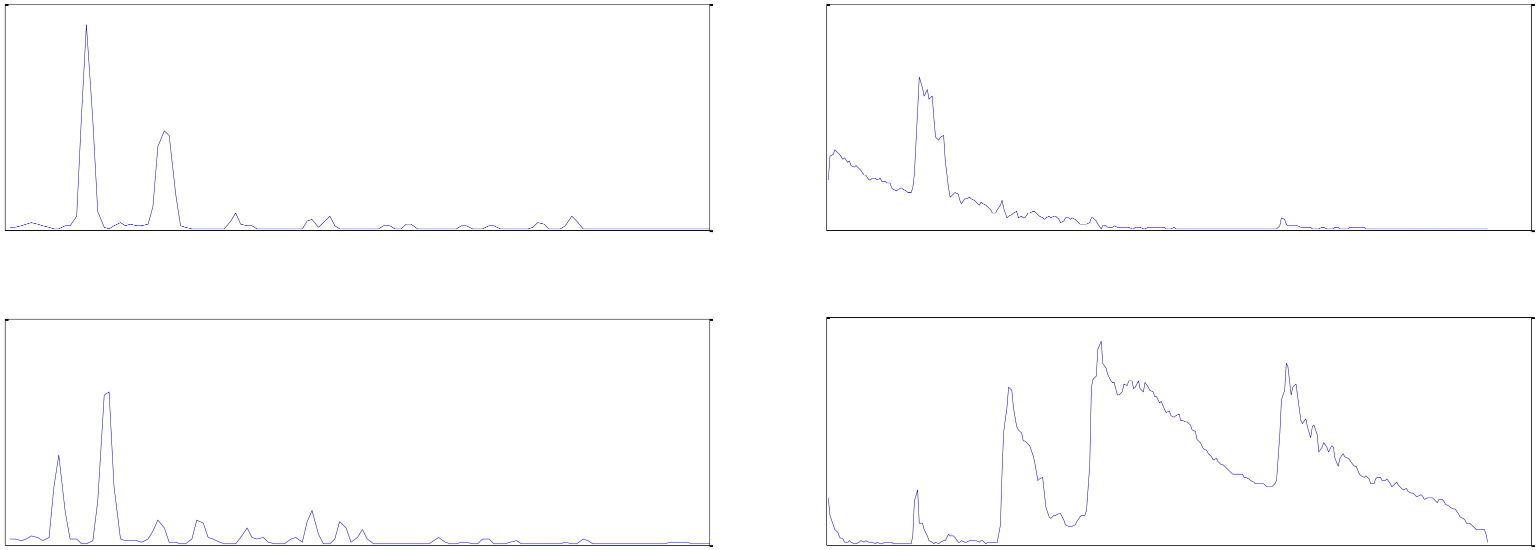
H

If r is too small

- Let $r = 2$

Original

Reconstructed



W^T H

How to determine r?

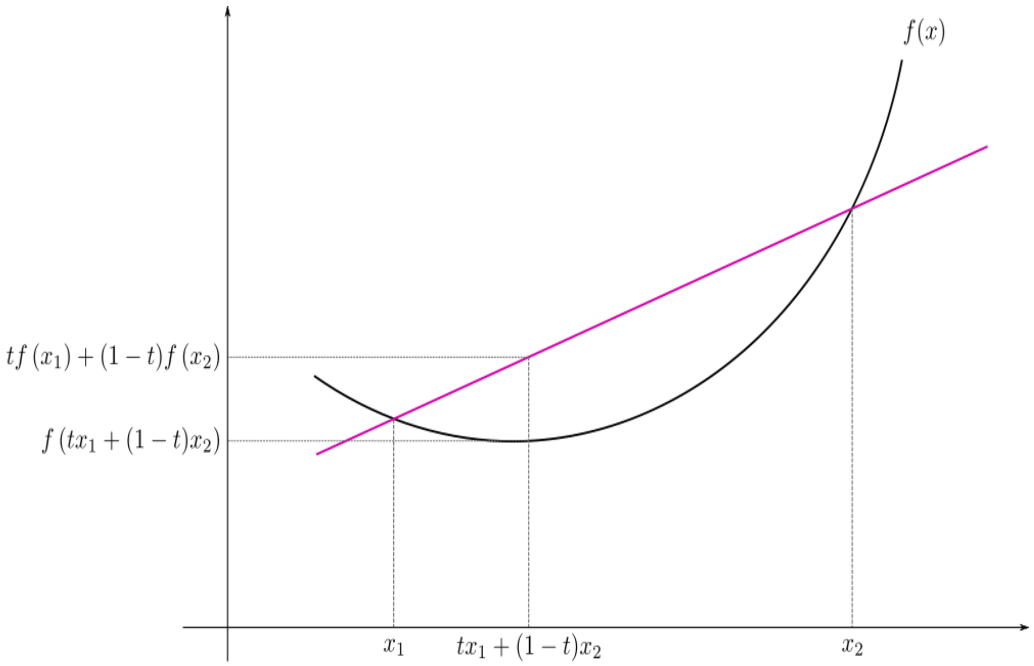
- This is the “secrete” of NMF.
- Look at the data.
- Try different values, and choose the smallest that provides good enough reconstruction.

Convex Functions

- $f(x)$ is convex if $\forall x_1, x_2$ and $\forall \lambda \in [0,1]$, we have
$$f(\lambda x_1 + (1-\lambda)x_2) \leq \lambda f(x_1) + (1-\lambda) f(x_2)$$

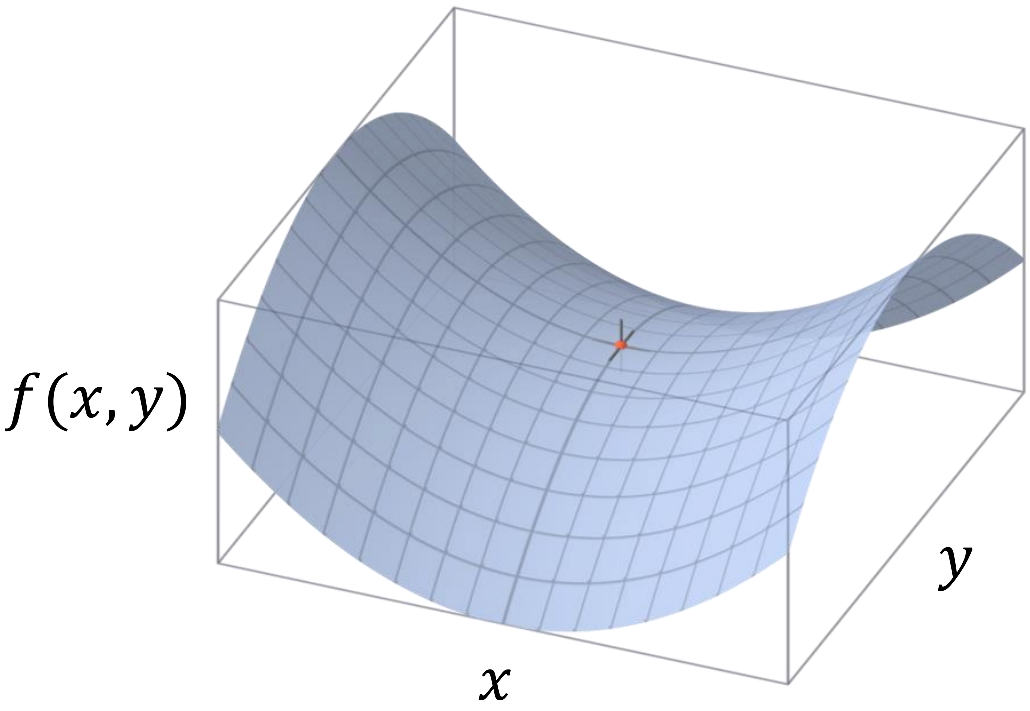
- $f(x) = (x-3)^2$
- $f(x) = \frac{1}{x}, x > 0$

Single local minimum
(if it has a minimum)



Convex?

- $f(x, y) = x^2 - y^2$



Convexity?

$$D(V \parallel WH) = \sum_{i,j} \left(V_{ij} - (WH)_{ij}\right)^2$$

$$D(V \parallel WH) = \sum_{i,j} \left(V_{ij} \ln \frac{V_{ij}}{(WH)_{ij}} - V_{ij} + (WH)_{ij}\right)$$

- Convex functions w.r.t. either W only or H only, but not W and H together
- Lots of local minima

The Algorithms

- Alternating non-negative least squares
- Projected gradient descent
- Active-set method
- Block principal pivoting
- …

- Multiplicative update rule
  - Easy to implement
  - Never get to negative values

Multiplicative Update

- For Euclidean distance [Lee & Seung, 1999]

$$W_{ia} \leftarrow W_{ia} \frac{(VH^T)_{ia}}{(WHH^T)_{ia}}$$

$$H_{a\mu} \leftarrow H_{a\mu} \frac{(W^TV)_{a\mu}}{(W^TWH)_{a\mu}}$$

Multiplicative Update

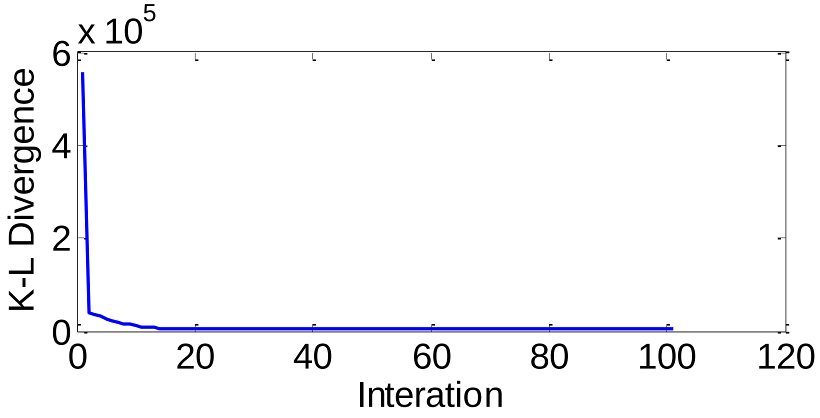
- For K-L divergence [Lee & Seung, 1999]

$$W_{ia} \leftarrow W_{ia} \frac{\sum_\mu H_{a\mu} V_{i\mu}/(WH)_{i\mu}}{\sum_\nu H_{a\nu}}$$

$$H_{a\mu} \leftarrow H_{a\mu} \frac{\sum_i W_{ia} V_{i\mu}/(WH)_{i\mu}}{\sum_k W_{ka}}$$

Convergence

- The multiplicative update rule decreases the cost function in each iteration.



- It converges to some local minimum.

- The convergence is pretty fast.

Problems of Multiplicative Updates

- Non-uniqueness issue

$$WH = (W\Sigma)(\Sigma^{-1}H)$$

  - Solution: normalize W to make each column sum to 1 in each iteration.

- Zero elements won't get updated.
  - Solution: make sure W and H do not have zero elements in initialization.

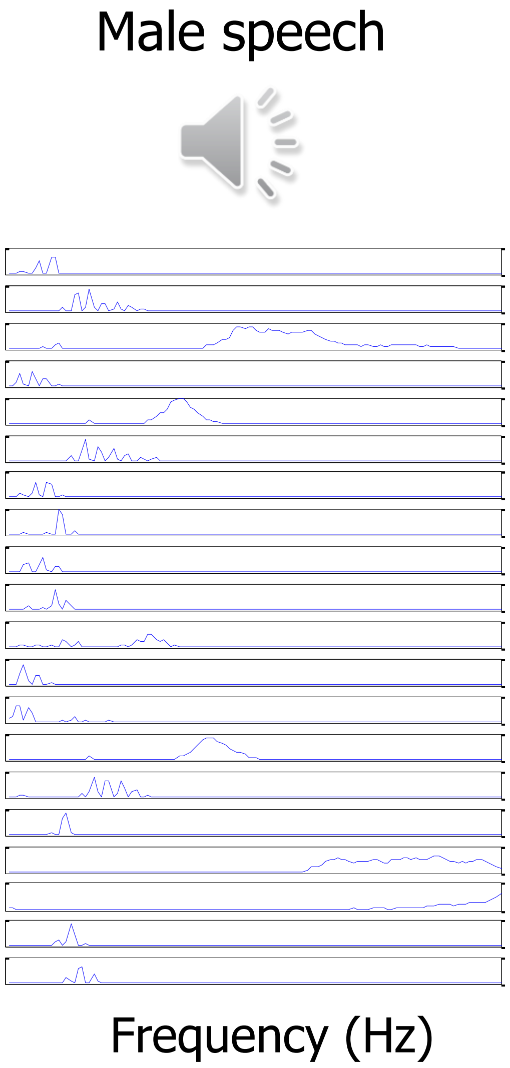
Initialization

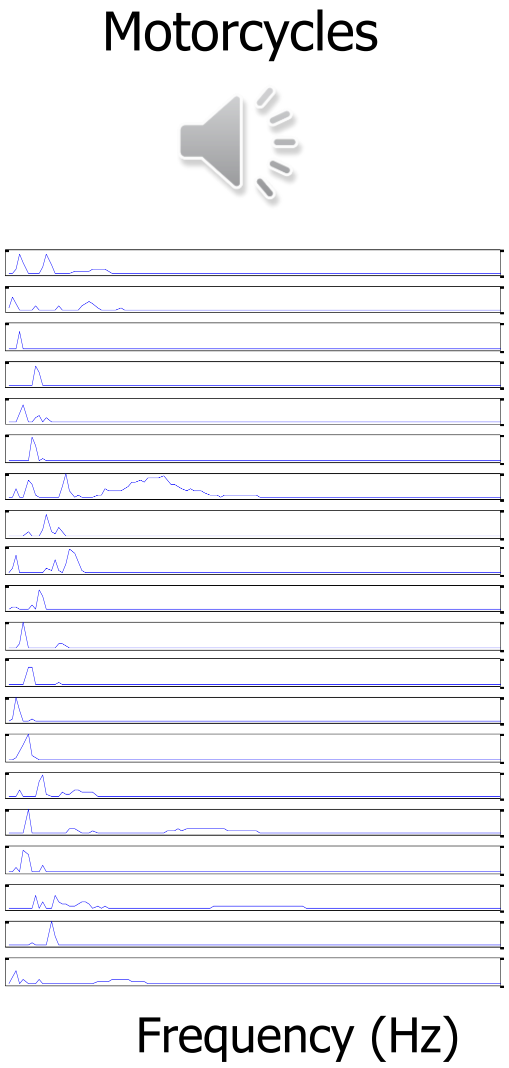
- Initialization affects the final result a lot, because the cost function is not convex.

- For simple data, random initialization is usually ok.

- For more complex data, use domain knowledge to initialize the dictionary.
  - E.g., for music transcription, initialize basis as a bunch of harmonic combs.

The Dictionary Models Source Timbre





Question

- Can we use the source dictionaries to separate sound sources in the mixture signal?

$$V_{mix} \approx V_1 + V_2$$

$$\approx W_1 H_1 + W_2 H_2$$

$$= [W_1, W_2] \begin{bmatrix} H_1 \\ H_2 \end{bmatrix}$$

Mixture spectrogram

Source 1 spectrogram

Source 2 spectrogram

Source 1 dictionary

Source 2 dictionary

Unsupervised Source Separation

- Decompose the mixture spectrogram directly

$$V_{mix} \approx W_{mix} H_{mix}$$

- Figure out what columns of W_{mix} belong to what sources
  - Difficult, could be impossible, if sources have similar spectral profiles
- Extract those columns as W_1;   Extract corresponding rows of H_{mix} as H_1
- Reconstruct the source signal $W_i H_i$

Supervised Source Separation

- Decompose training signals of all sources

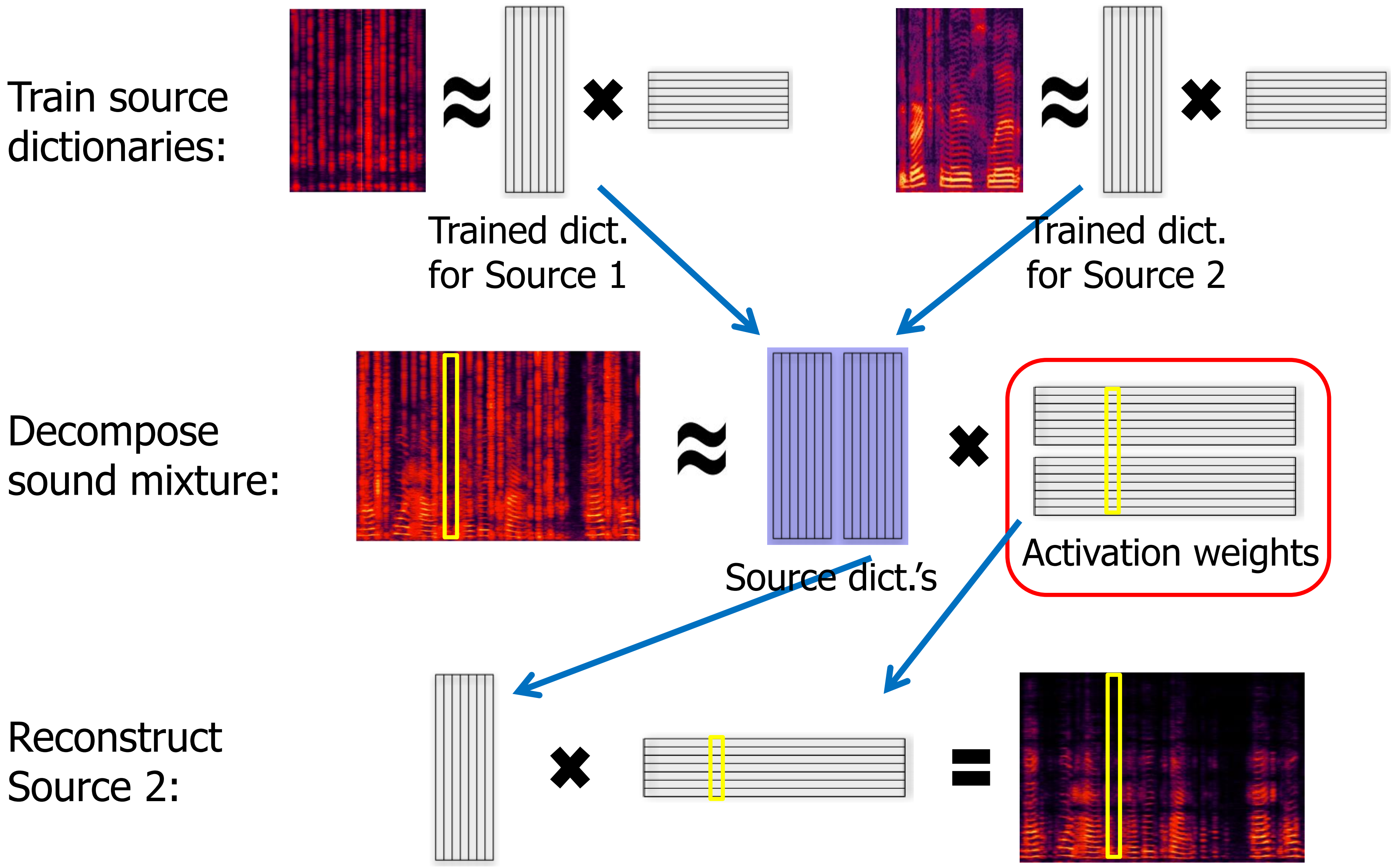
$$V_{\text{train},1} \approx W_1 H_{\text{train},1}, \qquad V_{\text{train},2} \approx W_2 H_{\text{train},2}$$

- Compose a new dictionary $W = [W_1, W_2]$
- Decompose mixture spectrogram using and fixing W, i.e., do not update W, but update H

$$V_{mix} \approx [W_1, W_2] \begin{bmatrix} H_1 \\ H_2 \end{bmatrix}$$

- Reconstruct the source signal $W_i H_i$

Supervised Source Separation illustration



Semi-supervised Source Separation

- Decompose training signals of some source(s)

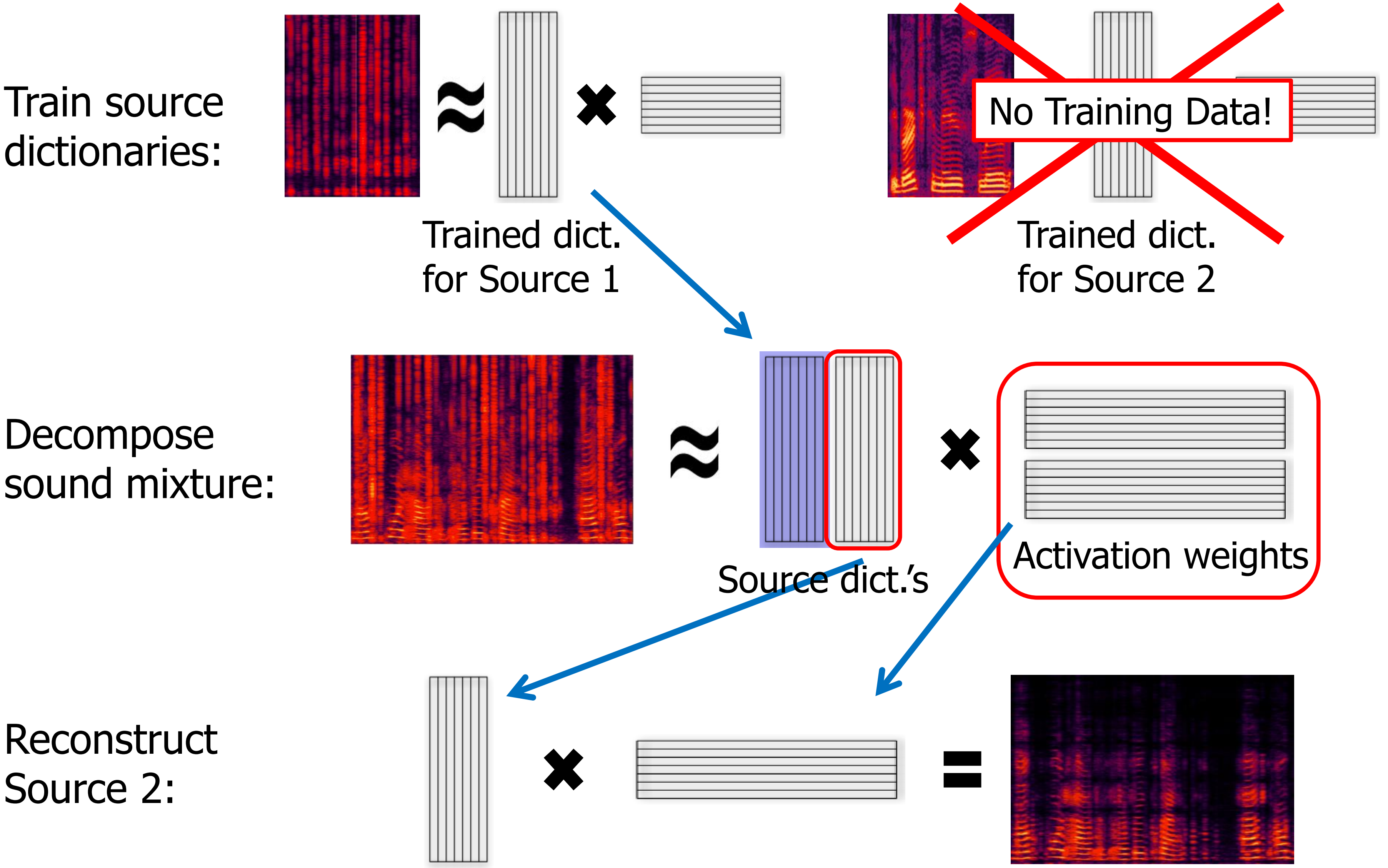
$$V_{\text{train},1} \approx W_1 H_{\text{train},1}$$

- Compose a new dictionary $W = [W_1, W_2]$, where W_2 is randomized.
- Decompose mixture spectrogram fixing W_1, i.e. do not update W_1, but update W_2 and H.

$$V_{mix} \approx [W_1, W_2] \begin{bmatrix} H_1 \\ H_2 \end{bmatrix}$$

- Reconstruct the source signal $W_i H_i$.

Semi-supervised Separation illustration



Look at NMF from another perspective!

- Think about the spectrogram V as a 2-d histogram of sound quanta.

- At each frame t, the sound quanta are distributed along the frequency axis according to $P_t(f)$.
- The number of sound quanta at (t, f) is V_{ft}.
- The number of sound quanta at frame t is $V_t = \sum_f V_{ft}$.

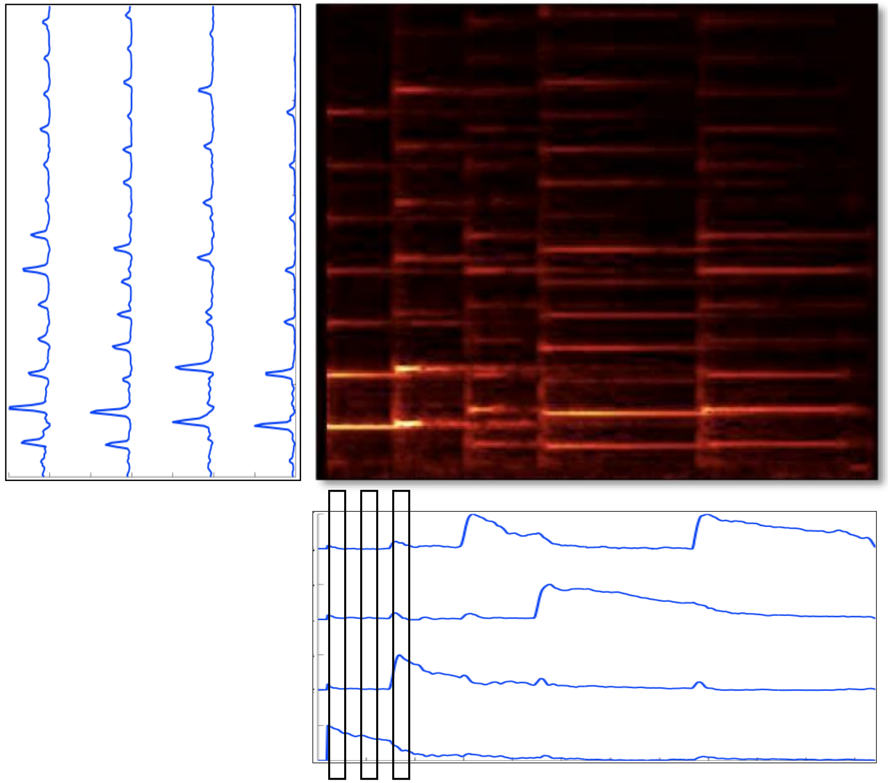
Probabilistic Latent Component Analysis

[Smaragdis, Raj 2006]

Sound quanta distribution at t →

$$P_t(f) \approx \sum_z P(f|z)P_t(z)$$

Dictionary Elements $P(f|z)$ →

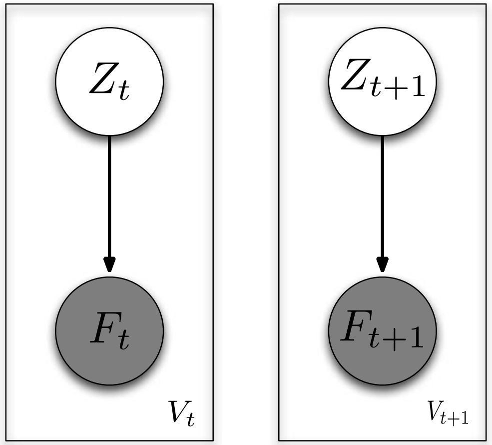


Time-invariant sound quanta distribution for each component

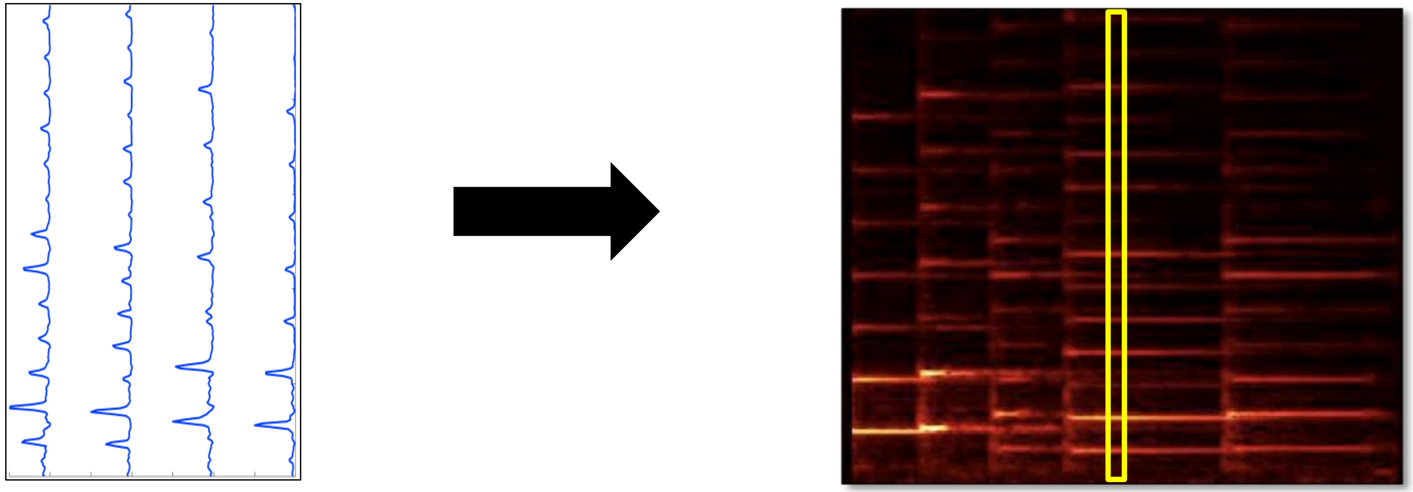
← Activation weights $P_t(z)$

Distribution of components

Generative Process



1. Choose a dictionary element according to $P_t(z)$
2. Choose a frequency from dictionary element z according to the distribution $P(f|z)$
3. Continue the process for V_t draws



How to estimate the parameters?

- Observation: a bunch of sound quanta distributed as V_{ft}

- Model:

$$P_t(f) = \sum_z P_t(f|z)P_t(z) \approx \sum_z P(f|z)P_t(z)$$

- Parameters: $P(f|z)$ and $P_t(z)$

Maximum Likelihood Estimation

- The data likelihood, i.e., the joint probability of all sound quanta

$$\prod_t \prod_f P_t(f)^{V_{ft}}$$

- Log data likelihood

$$\sum_t \sum_f V_{ft} \log P_t(f)$$

Expectation-Maximization

- E step: calculate the posterior distribution of latent components

$$P_t(z|f) = \frac{P(f|z)P_t(z)}{\sum_z P(f|z)P_t(z)}$$

- M step: maximize the expected complete data log-likelihood w.r.t. parameters $P(f|z)$ and $P_t(z)$.

$$\max_{P(f|z),P_t(z)} \mathrm{E}_{P_t(z|f)}\left\{\sum_t \sum_f V_{ft} \log P_t(f,z)\right\}$$

Let's derive the update equations

- See whiteboard.