$V = WH$

Non-negative Matrix Factorization - II

EE698V - Machine Learning for Signal Processing

Vipul Arora

NMF – seminal paper

- Lee, Daniel D., and H. Sebastian Seung, 'Learning the Parts of Objects by Non-Negative Matrix Factorization', Nature, 401 (1999), 788–91

Learning the parts of objects by non-negative matrix factorization

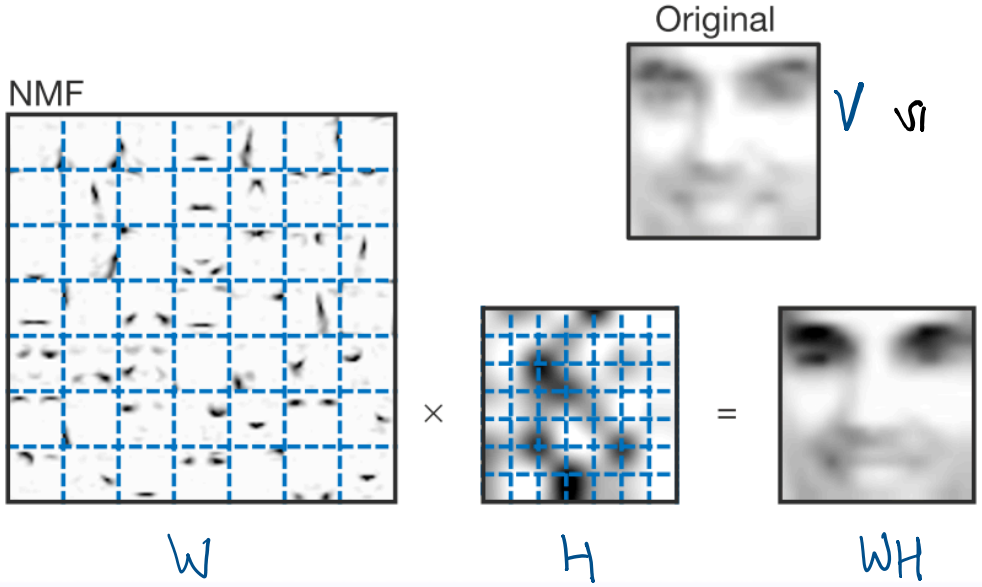
Daniel D. Lee* **& H. Sebastian Seung*†**

** Bell Laboratories, Lucent Technologies, Murray Hill, New Jersey 07974, USA*
† Department of Brain and Cognitive Sciences, Massachusetts Institute of Technology, Cambridge, Massachusetts 02139, USA

Is perception of the whole based on perception of its parts? There is psychological[1] and physiological[2,3] evidence for parts-based representations in the brain, and certain computational theories of object recognition rely on such representations[4,5]. But little is known about how brains or computers might learn the parts of objects. Here we demonstrate an algorithm for non-negative matrix factorization that is able to learn parts of faces and semantic features of text. This is in contrast to other methods, such as principal components analysis and vector quantization, that learn holistic, not parts-based, representations. Non-negative matrix factorization is distinguished from the other methods by its use of non-negativity constraints. These constraints lead to a parts-based representation because they allow only additive, not subtractive, combinations. When non-negative matrix factorization is implemented as a neural network, parts-based representations emerge by virtue of two properties: the firing rates of neurons are never negative and synaptic strengths do not change sign.

We have applied non-negative matrix factorization (NMF), together with principal components analysis (PCA) and vector quantization (VQ), to a database of facial images. As shown in

Recap: NMF



Goal: estimate W and H, s.t.

$V \approx WH$ for a given V.

$v_i = h_0 w_0 + h_1 w_1 + \ldots$

vec = scalar vec + ...

basis vectors

coeff or activations

$$\boldsymbol{V} \geq 0, \boldsymbol{W} \geq 0, \boldsymbol{H} \geq 0$$

Recap: PLCA

$$V_{ft} \sim P_t(f) = \sum_z P(f|z) P_t(z)$$

V W H

- For images, f may denote pixel position
- For audio spectra, f may denote frequency bin

Recap: PLCA optimization

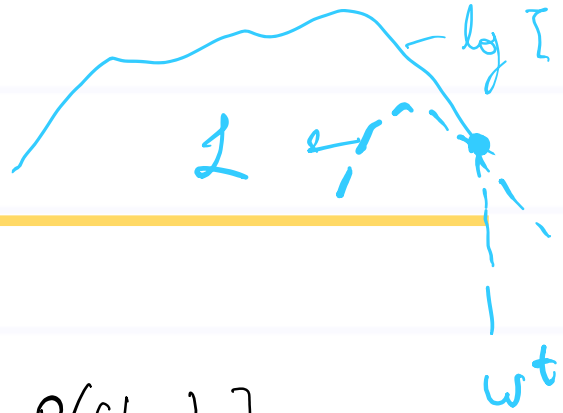
$$\log \text{likelihood} = \sum_{f,t} V_{ft} \log P_t(f)$$

$$= \sum_{f,t} V_{ft} \log \left[\sum_z P_t(z) P(f|z) \right]$$

given t

- We used EM algorithm to estimate $P_t(z)$ and $P(f|z)$

Recap: PLCA optimization



$$\text{Aux. fn., } \mathcal{L}(q,\theta) = \sum_{f,t} V_{ft} \sum_{z} q_t(z|f) \log\left[\frac{P_t(z)\, P(f|z)}{q_t(z|f)}\right]$$

- Iterate between E and M steps

$$\text{E-step: } q_t(z|f) = \frac{P_t(z)\, P(f|z)}{\sum_z P_t(z)\, P(f|z)}$$

$$\text{M-step: } P_t(z) = \frac{\sum_f V_{ft}\, q_t(z|f)}{\sum_{z'}\sum_f V_{ft}\, q_t(z'|f)}$$

$$P(f|z) = \frac{\sum_t V_{ft}\, q_t(z|f)}{\sum_f \sum_t V_{ft}\, q_t(z|f)}$$

NMF: another method

- Lee, D D, and H S Seung, 'Algorithms for Non-Negative Matrix Factorization', in *Proc. Conf. Adv. Neural Info. Process. Syst. (NIPS)* (MIT Press, 2001), XIII, 556–62

Algorithms for Non-negative Matrix Factorization

Daniel D. Lee*
*Bell Laboratories
Lucent Technologies
Murray Hill, NJ 07974

H. Sebastian Seung*†
†Dept. of Brain and Cog. Sci.
Massachusetts Institute of Technology
Cambridge, MA 02138

Abstract

Non-negative matrix factorization (NMF) has previously been shown to be a useful decomposition for multivariate data. Two different multiplicative algorithms for NMF are analyzed. They differ only slightly in the multiplicative factor used in the update rules. One algorithm can be shown to minimize the conventional least squares error while the other minimizes the generalized Kullback-Leibler divergence. The monotonic convergence of both algorithms can be proven using an auxiliary function analogous to that used for proving convergence of the Expectation-Maximization algorithm. The algorithms can also be interpreted as diagonally rescaled gradient descent, where the rescaling factor is optimally chosen to ensure convergence.

1 Introduction

Unsupervised learning algorithms such as principal components analysis and vector quantization can be understood as factorizing a data matrix subject to different constraints. Depending upon the constraints utilized, the resulting factors can be shown to have very different representational properties. Principal components analysis enforces only a weak orthogonality constraint, resulting in a very distributed representation that uses cancellations to generate variability [1, 2]. On the other hand, vector quantization uses a hard winner-take-all constraint that results in clustering the data into mutually exclusive prototypes [3].

1. Data as vectors
2. Cost fn.

} space

fix W

$$F(h) = \sum_i (v_i - W h_i)^2$$

T.P. $\quad G(h, h^t) - F(h) \geq 0$

$$\underbrace{(h - h^t)}_{v}{}^T \left(K(h^t) - W^T W\right) \underbrace{(h - h^t)}_{v} \geq 0$$

$$\text{LHS} = \sum_{ab} v_a \underbrace{\left(K - W^T W\right)}_{h_a} v_b$$

$$\frac{(W^T W h^t)_a}{h^t_a} \delta_{ab}$$

To prove :

Lemma 2 *If $K(h^t)$ is the diagonal matrix*

$$K_{ab}(h^t) = \delta_{ab}(W^T W h^t)_a / h_a^t \tag{13}$$

then

$$G(h, h^t) = F(h^t) + (h - h^t)^T \nabla F(h^t) + \frac{1}{2}(h - h^t)^T K(h^t)(h - h^t) \tag{14}$$

is an auxiliary function for

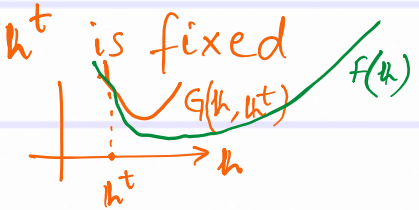
$$F(h) = \frac{1}{2}\sum_i (v_i - \sum_a W_{ia} h_a)^2 \tag{15}$$

$\frac{1}{2} \| v - W h \|^2$

i.e., $G(h, h^t) \geq F(h) \quad \forall h$, equality holds at $h = h^t$

i.e., $G(h, h^t) - F(h) \geq 0$ " "

- h is a vector i.e., h_v
- h^t is fixed



Proof :- Expand $F(h)$ with Taylor's expansion:

$$F(h) = F(h^t) + (h - h^t)^T \underbrace{\frac{\partial F(h^t)}{\partial h}}_{\nabla F(h^t)} + \frac{1}{2}(h - h^t)^T \underbrace{\frac{\partial^2 F(h^t)}{\partial h^2}}_{\frac{\partial F}{\partial h_{a'}} = \frac{1}{2}\sum_i \left(v_i - \sum_a w_{ia} h_a\right)(-w_{ia'})} (h - h^t)$$

$\frac{\partial F}{\partial h_{a'}}$ ↑ scalar

$$\frac{\partial^2 F}{\partial h_{a''} \partial h_{a'}} = \sum_i (-w_{ia''})(-w_{ia'}) = (W^T W)_{a''a'}$$

$$\therefore \underbrace{G(h, h^t) - F(h)}_{\text{call it LHS.}} = \frac{1}{2}(h - h^t)^T \underbrace{\left(K(h^t) - W^T W\right)}_{= \delta_{ab} \frac{(W^T W h^t)_a}{(h^t)_b}} (h - h^t)$$

a and b are indices of vectors or matrices

Let $(h - h^t)_a = v_a h^t_a$, i.e., element wise product of v and h^t

$$\therefore \text{LHS} = \frac{1}{2} \sum_{a,b} v_a h^t_a \underbrace{\left(K(h^t) - W^T W\right)_{ab}}_{\delta_{ab} \frac{(W^T W h^t)_a}{(h^t)_a}} v_b h^t_b$$

$$= \frac{1}{2} \sum_{a,b} v_a h^t_a \delta_{ab} \frac{(W^T W h^t)_a}{(h^t)_a} v_b h^t_b - \frac{1}{2} \sum_{a,b} v_a h^t_a (W^T W)_{ab} v_b h^t_b$$

$$\sum_{b'} (W^T W)_{ab'} h^t_{b'}$$

$$\sum_a v_a^2 h^t_a \sum_{b'} (W^T W)_{ab'} h^t_{b'}$$

$$\therefore \text{LHS} = \frac{1}{2} \sum_{a,b} v_a^2 h_a^t h_b^t (W^T W)_{ab} - \sum_{a,b} v_a v_b h_a^t h_b^t (W^T W)_{ab}$$

$$= \frac{1}{2} \sum_{a,b} h_a^t h_b^t (W^T W)_{ab} \left[v_a^2 - v_a v_b \right]$$

how to show LHS ≥ 0 ?

Note that LHS is symmetric in a, b

$$\text{LHS} = \frac{1}{2} \sum_{b,a} h_b^t h_a^t \underbrace{(W^T W)_{ba}}_{= (W^T W)_{ab}} \left[v_b^2 - v_b v_a \right]$$

Sum and divide by 2,

$$\text{LHS} = \frac{1}{2} \sum_{a,b} h_a^t h_b^t (W^T W)_{ab} \underbrace{\left[\frac{v_a^2}{2} + \frac{v_b^2}{2} - \frac{2 v_a v_b}{2} \right]}_{\frac{(v_a - v_b)^2}{2} \geq 0}$$

≥ 0 ≥ 0 ≥ 0

≥ 0

Hence proved!

Now, how to optimize $G(h, h^t)$?

$$h^{t+1} \leftarrow \underset{h}{\operatorname{argmax}}\, G(h, h^t)$$

$$G(h, h^t) = F(h^t) + (h - h^t)^T \nabla F(h^t) + \frac{1}{2}(h - h^t)^T K(h^t)(h - h^t)$$

$\sum_i v_i w_{ia'}$

$= W^T v$

or $v^T W$

$\sum_{i,a} w_{ia} h_a w_{ia'}$

$= \sum_{i,a} h_a w_{ai}^T w_{ia'}$

$= h^T W^T W$

or $= \sum_{i,a} w_{a'i}^T w_{ia} h_a$

$= W^T W h$

$$\frac{\partial F}{\partial h_{a'}} = -\sum_i \left(v_i - \sum_a w_{ia} h_a\right) w_{ia'}$$, derived before

$$\therefore \frac{\partial F}{\partial h} = -\left(W^T v - W^T W h\right)$$ ← vector indexed by a'

$$\therefore \nabla F(h^t) = \left.\frac{\partial F}{\partial h}\right|_{h = h^t} = -W^T v + W^T W h^t$$

$$\frac{\partial}{\partial h} G(h, h^t) = 0 + \nabla F(h^t) + K(h^t)(h - h^t) = 0$$

$$\therefore h = h^t - K(h^t)^{-1} \nabla F(h^t)$$

$$\frac{\partial}{\partial x}(u^T A v) = \frac{\partial u}{\partial x} A v + \frac{\partial v}{\partial x} A^T u$$

$K(h^t)$ is a diagonal matrix, so $\left(K(h^t)^{-1}\right)_{aa} = \frac{h_a^t}{\left(W^T W h^t\right)_a}$

$$\therefore h_a = h_a^t - \frac{h_a^t}{\left(W^T W h^t\right)_a} \left(-W^T v + W^T W h^t\right)_a$$

$$= h_a^t + \frac{h_a^t}{\left(W^T W h^t\right)_a} \left(W^T v\right)_a - \frac{h_a^t}{\left(W^T W h^t\right)_a} \left(W^T W h^t\right)_a$$

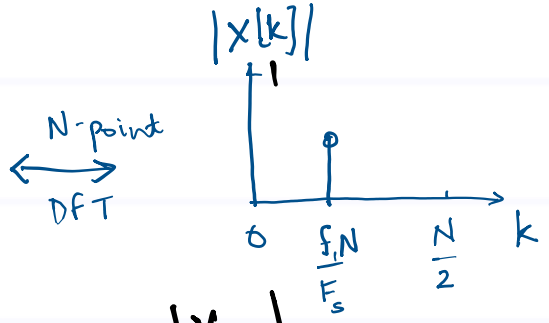
$$\therefore \boxed{h_a^{t+1} = h_a^t \frac{\left(W^T v\right)_a}{\left(W^T W h^t\right)_a}}$$

$v = Wh$

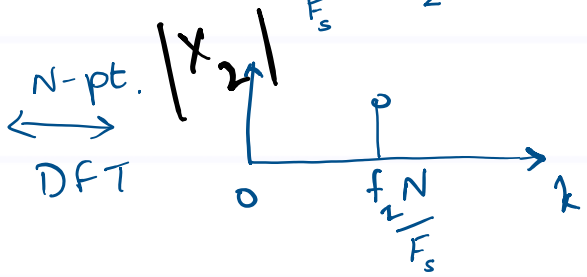
Similarly one can derive W

Fourier Transform

$$x_1[n] = \sin\left(2\pi f_1 \frac{n}{F_s}\right)$$

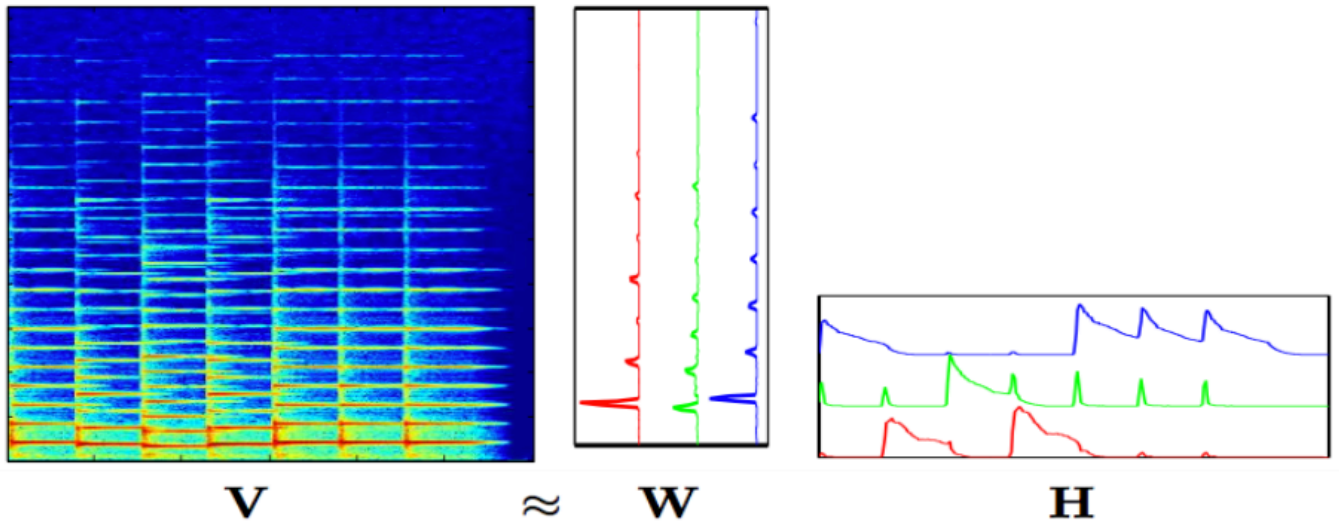


$$x_2[n] = \sin\left(2\pi f_2 \frac{n}{F_s}\right)$$



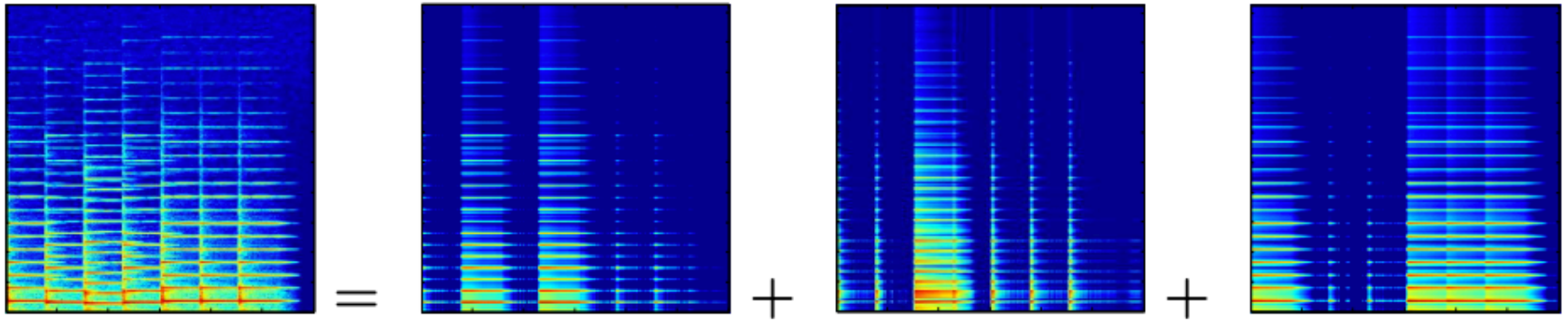
$x = x_1 + x_2$ $\leftrightarrow$

NMF on STFT: |X[k]| as a feature vector



Courtesy: https://ccrma.stanford.edu/~njb/teaching/sstutorial/part2.pdf

Reconstruct each pitch separately



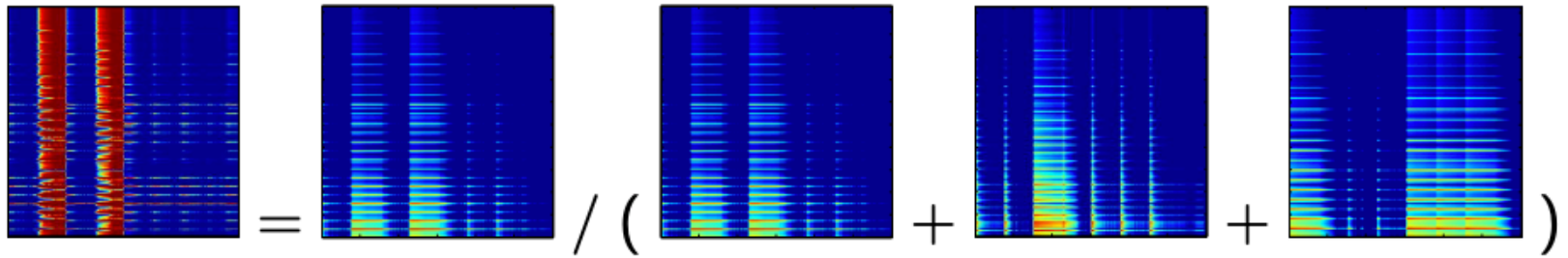
$$\mathbf{V} \approx \mathbf{w}_1 \mathbf{h}_1^{\mathrm{T}} + \mathbf{w}_2 \mathbf{h}_2^{\mathrm{T}} + \mathbf{w}_3 \mathbf{h}_3^{\mathrm{T}}$$

$$|\hat{\mathbf{X}}_s| \approx \mathbf{w}_1 \mathbf{h}_1^{\mathrm{T}}$$

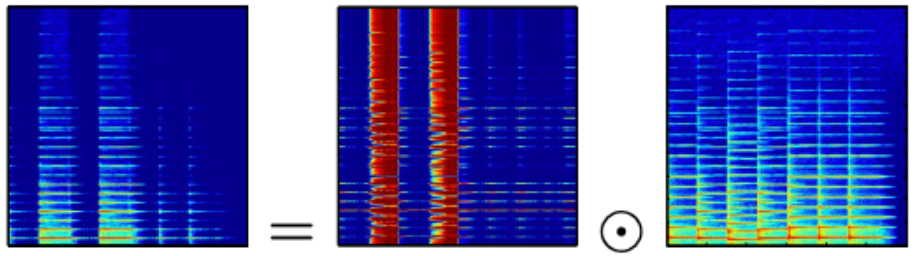
Reconstruction with masking

1. Compute filter $\mathbf{M}_s = \dfrac{\mathbf{w}_1 \mathbf{h}_1^{\mathrm{T}}}{\sum_{i=1}^{K} \mathbf{w}_i \mathbf{h}_i^{\mathrm{T}}}$, with $\alpha = 1$

$$\mathbf{M}_s = \frac{(\mathbf{W}_s \mathbf{H}_s)^{\alpha}}{\sum_{i=1}^{K} (\mathbf{W}_i \mathbf{H}_i)^{\alpha}} = \frac{|\hat{\mathbf{X}}_s|^{\alpha}}{\sum_{i=1}^{K} |\hat{\mathbf{X}}_i|^{\alpha}}$$



2. Multiply with $|\hat{\mathbf{X}}_s| = \mathbf{M}_s \odot |\mathbf{X}|$



Spectrum to x[n]

- Inverse DFT to get $x[n]$
  - Use $|\hat{X}_s[k]|$ and $\angle X[k]$

Source Separation

- Train W_1 from source 1
- Train W_2 from source 2
- Decompose mixture as $V = [W_1\ W_2]\begin{bmatrix} H_1 \\ H_2 \end{bmatrix}$