Adverserial Non-Negative Matrix Factorization for Single Channel Source Separation

DNA Seminar 16. November 2022

Martin Ludvigsen

Department of Mathematical Sciences, NTNU.

Norwegian University of Science and Technology

Prelude

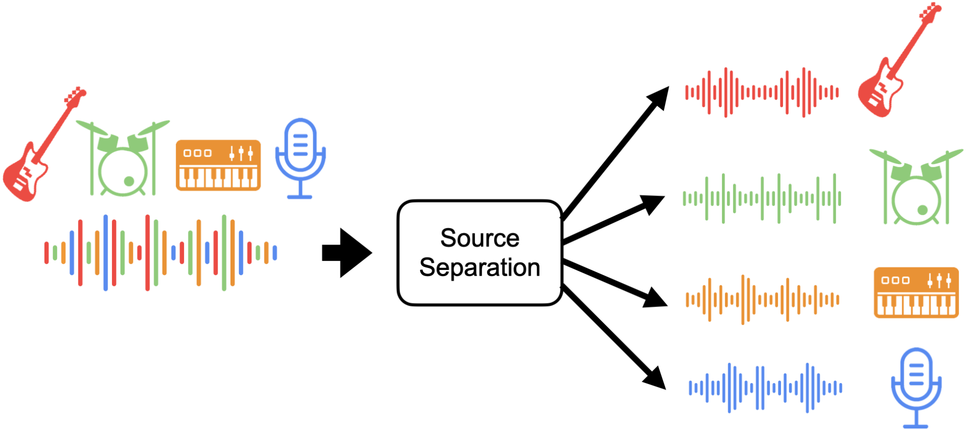


Figure: "De-mixing" music. Measure single channel. Source:
https://source-separation.github.io/tutorial/landing.html

Single Channel Source Separation (SCSS)

Problem formulation

$$v = \sum_{i=1}^{S} u_i = Au,$$

$$A = \begin{bmatrix} I & \cdots & I \end{bmatrix}, \quad u = \begin{bmatrix} u_1^T & \cdots & u_S^T \end{bmatrix}^T.$$

Given measured mixed signal $v \in \mathbb{R}^m$, want to recover up to S individual source signals $u_i \in \mathbb{R}^m$, $i = 1, ..., S$.

Single Channel Source Separation (SCSS)

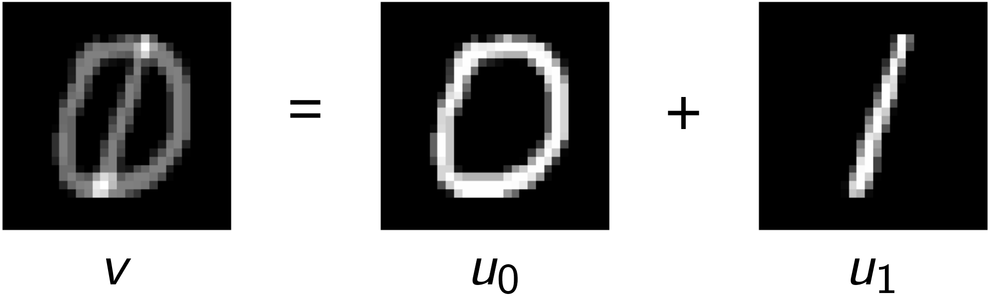
Problem formulation

$$v = \sum_{i=1}^{S} u_i = Au,$$

$$A = \begin{bmatrix} I & \cdots & I \end{bmatrix}, \quad u = \begin{bmatrix} u_1^T & \cdots & u_S^T \end{bmatrix}^T.$$

Given measured mixed signal $v \in \mathbb{R}^m$, want to recover up to S individual source signals $u_i \in \mathbb{R}^m$, $i = 1, \dots, S$.

- Linear inverse problem.
- Underdetermined $\rightarrow$ need prior information about the source signals.
- Data-driven approach is most reasonable for many problems.



Given a mixed image $v = u_0 + u_1$, can we recover the individual images u_0 and u_1?

Structure of talk

- ~~Introduction~~
- Non-Negative Matrix Factorization (NMF)
  - NMF for SCSS
  - NMF as projection onto convex cones
- Data setting for inverse problems and SCSS
- Adverserial regularization functions
  - Adverserial regularization functions for SCSS
  - Adverserial NMF (ANMF)
  - Numerical algorithm for ANMF
- Numerical experiments

Non-Negative Matrix Factorization (NMF)

- Assume non-negative $M \times N$ matrix $V \approx WH$.
- W is non-negative $M \times d$ matrix.
- H is non-negative $d \times N$ matrix.
- $d \ll N, M$ is the rank of the decomposition chosen a priori.

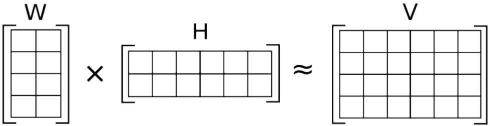


Figure: Source:
https://en.wikipedia.org/wiki/Non-negative_matrix_factorization

- Also called sparse (non-negative) dictionary learning.

$$\min_{W,H \geq 0} \|V - WH\|_F^2 + \mu_H |H|_1 + \mu_W |W|_1 .$$

- Non-convex, non-unique solutions.

NMF for source separation

- Assume that we have data from each individual source, stored columnwise in matrices U_i.
- During training, fit NMF for each matrix $U_i \to$ learn S non-negative bases W_i.
- During testing, want to separate v:

$$\min_{\substack{h_i \geq 0 \\ i=1,\dots,S}} \|v - \sum_{i=1}^{S} W_i h_i\|^2 + \mu_H \sum_{i=1}^{S} \|h_i\|_1.$$

- After solving $W_i h_i$ should approximate the i-th source.
- Define $W = [W_1 \cdots W_S]$, $h = [h_1^T \cdots h_S^T]^T$, write problem as

$$\min_{h \geq 0} \|v - Wh\|^2 + \mu_H \|h\|_1.$$

- Post-process with Wiener filter to ensure that the sources sum to v:

$$\text{Separated signals } u_i = v \odot \frac{W_i h_i}{\sum_{j=1}^{S} W_j h_j}.$$

MNIST data

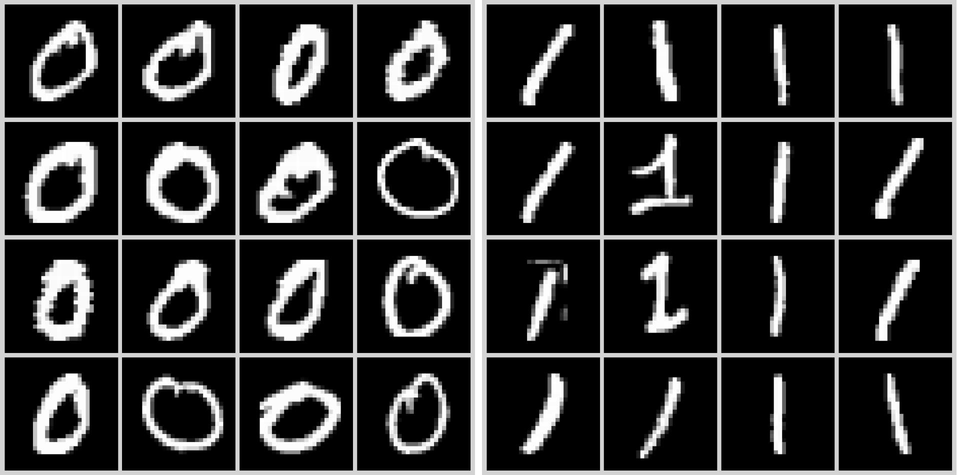


Figure: Zero digits and one digits from the MNIST dataset. For each source (each digit) we have $N = 2500$ grayscale images with resolution 28×28, which can be stored in a matrix $U_i \in \mathbb{R}_+^{784 \times 2500}$

NMF basis vectors

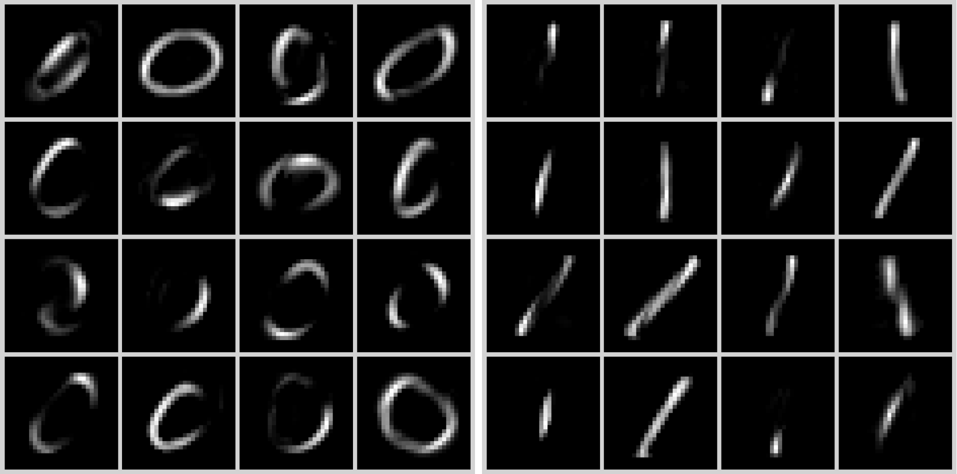


Figure: NMF basis vectors (columns of W_0 and W_1) for zero digits and one digits using $N = 2500$ images each and $d = 16$ basis vectors.

NMF projections

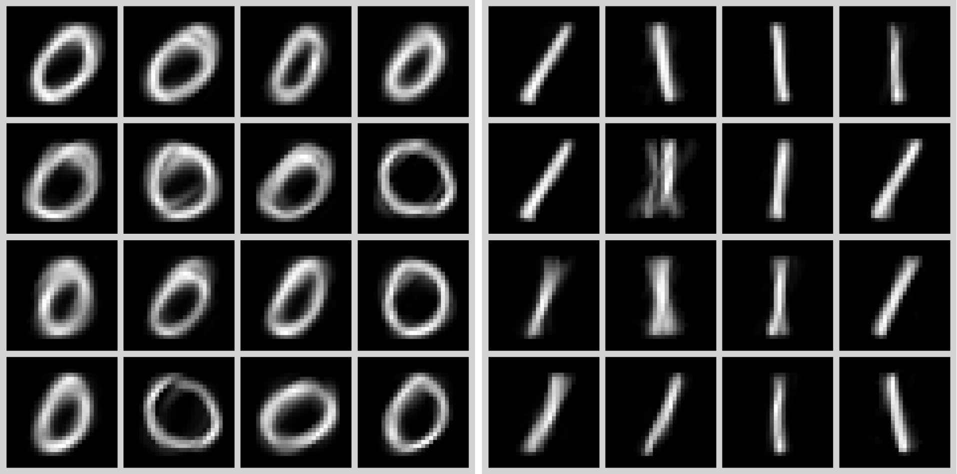
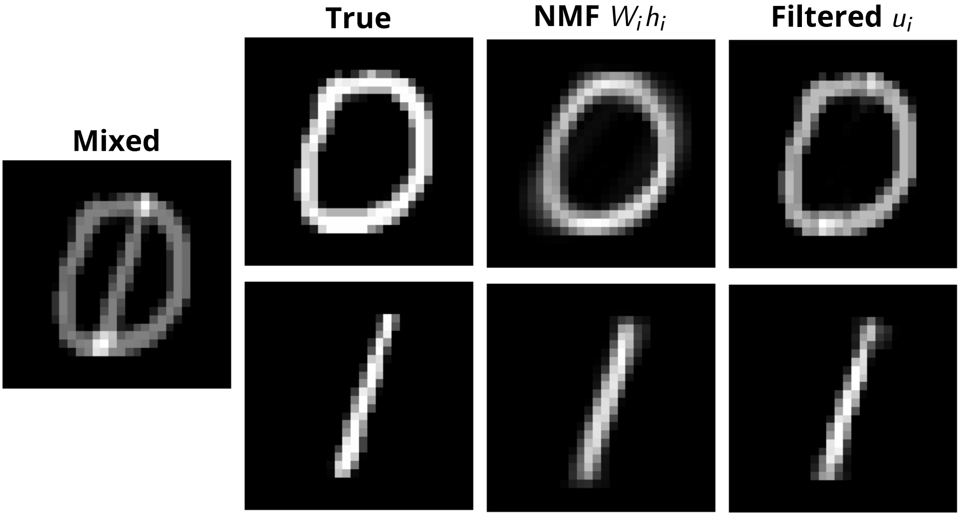


Figure: NMF projections (columns of $W_0 H_0$ and $W_1 H_1$) using $N = 2500$ and $d = 16$ for zero and one digits onto their respective bases.

NMF separation



NMF interpreted as projection onto convex cone

Define the convex cone $C(W) = \{Wh : W \in \mathbb{R}_+^{M\times d}, h \in \mathbb{R}_+^d\}$.

NMF taining can be formulated as the bi-level problem

$$\min_{W\geq 0} \|U - WH(U)\|_F,$$
$$\text{where } H(U) = \arg\min_{\hat{H}\geq 0} \|U - W\hat{H}\|_F.$$

where the upper part consists of fitting the cone $C(W)$.

Lower level training/test problem is projecting u onto the cone $C(W)$

$$\min_{h\geq 0} \|u - Wh\| = \min_{v\in C(W)} \|u - v\| = \|u - P_{C(W)}(u)\| = d_{C(W)}(u)$$

where $P_{C(W)}(u) = Wh(u)$ is the projection onto $C(W)$, and $d_C(u)$ is the distance to $C(W)$.

NTNU

Can we do better than fitting the convex cones individually? Can we use more of the data available?

Data setting

- Distribution of individual sources, $u_i \sim \mathbb{P}_{U_i}$.
- Distribution of measured mixed signals $v \sim \mathbb{P}_V$.
- Joint distribution $(v, u_1, ..., u_S) \sim \mathbb{P}_{V \times U_1 \times ... \times U_S} = \mathbb{P}_{V \times U}$.

Data setting

- Distribution of individual sources, $u_i \sim \mathbb{P}_{U_i}$.
- Distribution of measured mixed signals $v \sim \mathbb{P}_V$.
- Joint distribution $(v, u_1, ..., u_S) \sim \mathbb{P}_{V \times U_1 \times ... \times U_S} = \mathbb{P}_{V \times U}$.
- **Supervised**: Have access to $\mathbb{P}_U, \mathbb{P}_V$ and the joint $\mathbb{P}_{V \times U}$.
- **Unsupervised**: Have access to individual sources $\mathbb{P}_{U_i}$, mixed signals $\mathbb{P}_V$, but not joint $\mathbb{P}_{V \times U}$.
- **Synthetic supervised**: Unsupervised, but we use the forward model to create supervised data.
- We are interested in the unsupervised case where creating synthetic supervised data is infeasible.

Supervised data setting

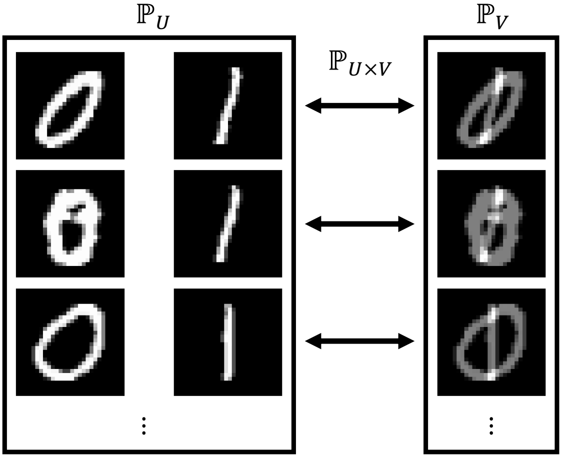


Figure: Supervised: Have access to all labeled data and joint between them.

Unsupervised data setting

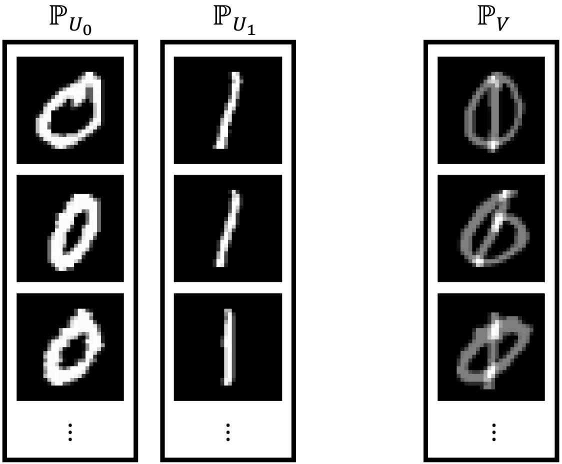


Figure: Unsupervised: Only have access to individual labeled data, but no "links" between them. We also do not have $\mathbb{P}_U = \mathbb{P}_{U_0 \times U_1}$.

Source separation with regularization functions

Regularization approach to source separation with pre-trained regularization functions R_i:

$$\min_{u_i, \forall i} \frac{1}{2} \| \sum_{i=1}^{S} u_i - v \|^2 + \sum_{i=1}^{S} \lambda_i R_i(u_i).$$

where $\lambda_i > 0$ are regularization parameters.

Goal: Learn suitable R_i $\rightarrow$ learn sets C_i that approximate the data and select $R_i = d_{C_i}$.

Adverserial regularization functions for source separation

Based on work by S. Lunz, O. Öktem and C. Schönlieb (*Adversarial Regularizers in Inverse Problems*, 2018)

- Define true data for source i, $u_i \sim \mathbb{P}_{U_i}$.
- Define the adverserial data for source i, $\mathbb{P}_{Z_i}$, which we choose as data from other sources and mixed data.
- R_i should be small for data belonging to $\mathbb{P}_{U_i}$.
- R_i should be large for data belonging to $\mathbb{P}_{Z_i}$.
- R_i should be sufficiently regular.

Fit regularization function by solving

$$\min_{R_i \in \Theta : \|R\|_L \leq 1} \mathbb{E}_{u \sim \mathbb{P}_{U_i}} [R_i(u)] - \mathbb{E}_{u \sim \mathbb{P}_{Z_i}} [R_i(u)]$$

where Θ is a parameterized space of functions, like neural networks, and $\|.\|_L$ denotes the Lipschitz constant.

Wasserstein Distance

$$\mathbb{W}(\mathbb{P}_Z, \mathbb{P}_U) = \min_{R:\|R\|_L \leq 1} \mathbb{E}_{u\sim\mathbb{P}_U}[R(u)] - \mathbb{E}_{u\sim\mathbb{P}_Z}[R(u)]$$

is the (dual form of the) Wasserstein distance between the probability distributions $\mathbb{P}_U$ and $\mathbb{P}_Z$.

Using the Wasserstein distance for learning structures of distributions in machine learning was popularized in work by M. Arjovsky et. al (*Wasserstein GAN* 2017).

Adverserial regularization as distance to convex set

Idea: Parameterize regularization function $R(u) = \|u - P_C(u)\| = d_C(u)$, where C is a convex set, specifically a convex cone.

- R is convex.
- R is Lipschitz continuous with constant $L = 1$.
- R has an easily computable proximal operator (soft-thresholding).
- If C is a convex cone, R is positive 1-homogenous: $R(cu) = cR(u)$ for $c > 0$.
- Naturally links to existing methods like NMF/SVD, autoencoders, GANs...

Adverserial NMF (ANMF)

Minimize Wasserstein distance where R is constrained to the distance to convex cones:

$$\min_{W \geq 0} \mathbb{E}_{u \sim \mathbb{P}_U}[d_{C(W)}(u)] - \mathbb{E}_{u \sim \mathbb{P}_Z}[d_{C(W)}(u)]$$

$$\approx \min_{W \geq 0} \frac{1}{N_r} \sum_{i=1}^{N_r} \|u_r^{(i)} - Wh(u_r^{(i)})\|_2 - \frac{1}{N_z} \sum_{i=1}^{N_z} \|u_z^{(i)} - Wh(u_z^{(i)})\|_2$$

where $h(u) = \arg\min_{\hat{h} \geq 0} \|u - W\hat{h}\|$.

- If we ignore the second term, this is just standard NMF.
- Can alternatively use Frobenius (squared) norm instead of 2-norm.
- With ANMF we want to both fit the true data $\mathbb{P}_U$ well and the adverserial data $\mathbb{P}_Z$ poorly.

NTNU

Weighted ANMF

Problem: NMF is already low complexity $\rightarrow$ NMF is bad at reconstructing data that is not in $\mathbb{P}_U$.

Weighted ANMF

Problem: NMF is already low complexity $\rightarrow$ NMF is bad at reconstructing data that is not in $\mathbb{P}_U$.

Solution: Fit a mix between NMF and ANMF

$$\min_{W\geq 0}(1-\tau)\underbrace{\mathbb{E}_{u\sim\mathbb{P}_U}[d_{C(W)}(u)]}_{\text{NMF}}+\tau\underbrace{(\mathbb{E}_{u\sim\mathbb{P}_U}[d_{C(W)}(u)]-\mathbb{E}_{u\sim\mathbb{P}_Z}[d_{C(W)}](u))}_{\text{ANMF}}$$
$$=\min_{W\geq 0}\mathbb{E}_{u\sim\mathbb{P}_U}[d_{C(W)}(u)]-\tau\mathbb{E}_{u\sim\mathbb{P}_Z}[d_{C(W)}(u)],$$

where $0\leq\tau\leq 1$ is a tuning parameter chosen a priori.

Low τ values $\rightarrow$ fit real data well.

High τ values $\rightarrow$ fit adverserial data poorly.

Numerical algorithm for NMF

Multiplicative algorithm popularized by D. Lee and H. Seung (*Algorithms for Non-Negative Matrix Factorization* 2001).

Multiplicative updates for fitting NMF to the matrix U with (quared) Frobenius norm:

$$W \leftarrow W \odot \frac{UH^T}{WHH^T + \mu_W}$$
$$H \leftarrow H \odot \frac{W^T U}{W^T WH + \mu_H}$$

All updates are non-negative, can prove that this converges to local minimizer. Initialize with non-negative random matrices.

For large datasets, divide U and H into batches and apply updates for each batch, Stochastic Multiplicative Update.

Numerical algorithm for ANMF

Similarily, we obtain multiplicative updates for fitting ANMF to $U_r \approx WH_r$ and adverserially to $U_z \neq WH_z$ with $N_r = N_z$ in squared Frobenius norm:

$$W \leftarrow W \odot \frac{U_r H_r^T + W H_z H_z^T}{U_z H_z^T + W H_r H_r^T + \mu_W}$$

$$H_r \leftarrow H_r \odot \frac{W^T U_r}{W^T W H_r + \mu_H}$$

$$H_z \leftarrow H_z \odot \frac{W^T U_z}{W^T W H_z + \mu_H}$$

Complexity scales with the total amount of data $\rightarrow$ use standard NMF as warm start initial conditions.

Convergence of NMF and ANMF algorithms

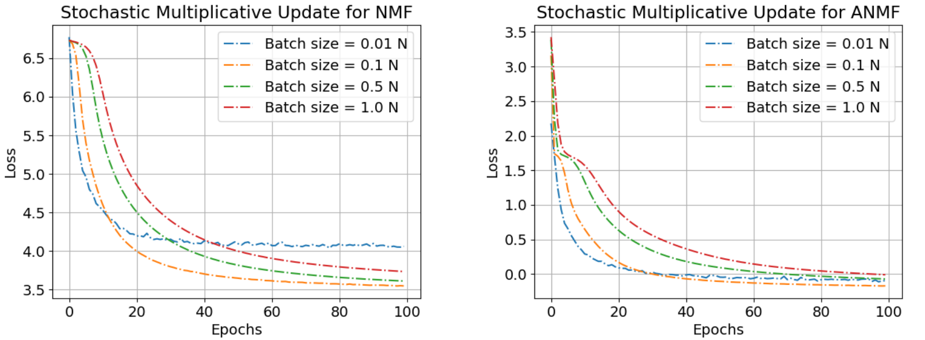


Figure: Convergence of numerical algorithms for NMF (left) and ANMF (right) with different batch sizes. $N = 2500$, $d = 32$, $\tau = 1.0$. For both experiments choosing the batch size to be $0.1N$ yields the best results as well as shortest computation time.

SCSS with NMF and ANM

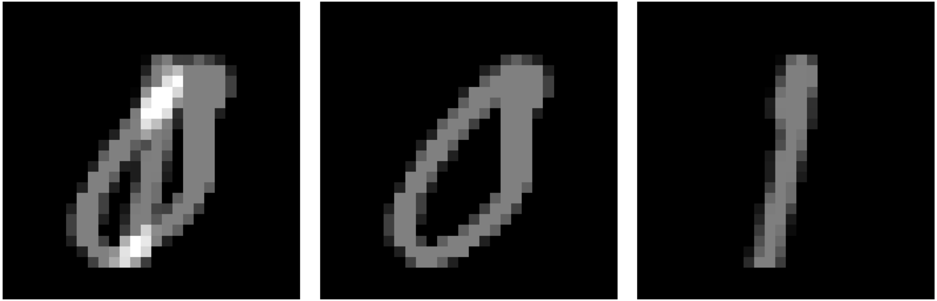


Figure: Real mixed and unmixed images test data.

SCSS with NMF and ANMF

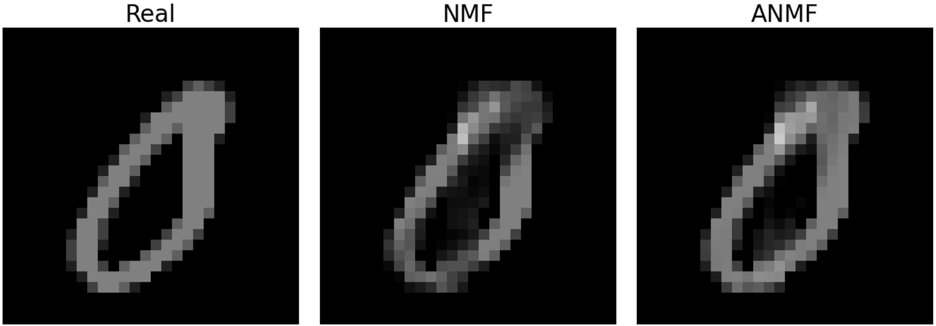


Figure: True data, NMF recovered and ANMF recovered solutions with $d = 32$, $\tau = 0.5$, $N = 2500$.

SCSS with NMF and ANMF

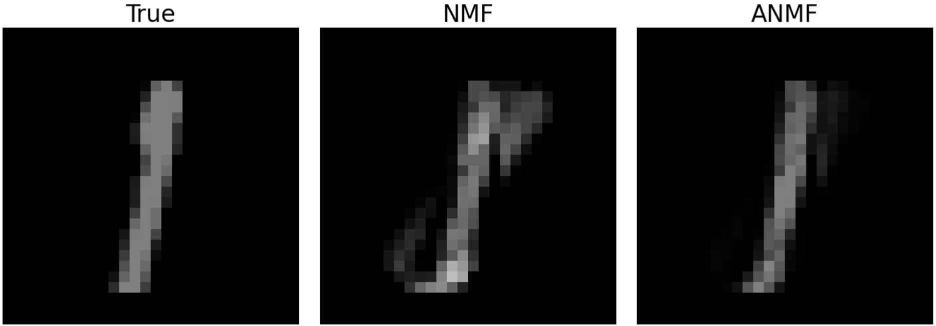


Figure: True data, NMF recovered and ANMF recovered solutions with $d = 32$, $\tau = 0.5$, $N = 2500$.

SCSS with ANMF for different d

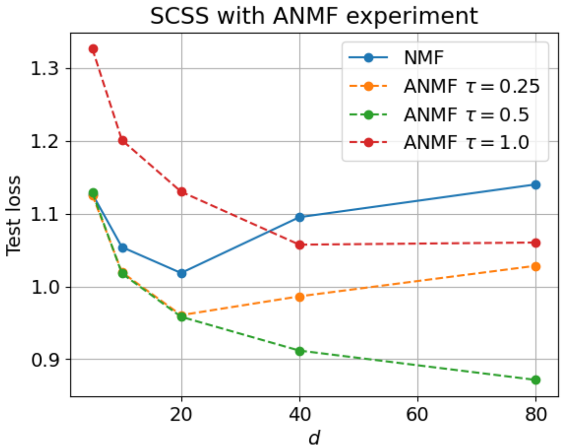


Figure: Comparison of separatation accuracy (distance between recovered and real data) for NMF and ANMF with different τ using 800 test data. All experiments are done using 100 epochs of training.

Generalization to any generative method

Generator	$g_C : H \to C$	$g_C(h) = Wh$
Projection	$P_C : X \to C$	$P_C(v) = g(\arg\min_{h\in H} \lVert v - g(h)\rVert)$ $= \arg\min_{u\in C} \lVert v - u\rVert$
Distance	$d_C : X \to \mathbb{R}_+$	$d_C(v) = \min_{u\in C} \lVert v - u\rVert$ $= \lVert v - P_C(v)\rVert$

Different testing problems:

$$\min_{u_i} \lVert \sum_i u_i - v\rVert^2 + \sum_i \lambda_i d_{C_i}(u_i)$$

$$\lambda_i \to \infty \implies \min_{h_i\in H} \lVert \sum_i g_i(h_i) - v\rVert = \min_{u_i\in C_i} \lVert \sum_i u_i - v\rVert$$

$$\lambda_i \to 0 \implies \min_{u_i, \sum_i u_i = v} \sum_i \mu_i d_{C_i}(u_i)$$