# Investigating the feature extraction capabilities of nonnegative matrix factorisation algorithms for black-and-white images

Liew How Hui, Ng Wei Shean, Chen Huey Voon

May 2024

## 1 Introduction

Objectives:

- Understand the mathematical formulation of nonnegative matrix factorisation (NMF);
- investigate the software implementations of various NMF algorithms;
- investigate the image features that can be extracted from NMF algorithms empirically on some black-and-white images.

Definition: A **nonnegative matrix** is a matrix where all the elements are nonnegative:

$$X = [x_{ij}]_{i=1,..,n}^{j=1,\dots,p}, \quad x_{ij} \geq 0.$$

It is usually denoted as $X \geq 0$.

Why are we interested in NMF?

Reason 1: It is related to our research domain — Linear Algebra.

Reason 2: It is more meaningful compared to canonical matrix factorisations for $X \geq 0$:

- LU (lower–upper decomposition): $X = PLU$. $P$ is a "binary" permutation matrix. There may be negative numbers in lower matrix $L$ and upper matrix $U$.
- QR (orthogonal-upper decomposition): $X = QR$. There may be negative numbers in $Q$ and $R$.
- SVD (singular value decomposition): $X = U\Sigma V^T$. There may be negative numbers in $U$ and $V$.

For text mining and image factorisations, the negative numbers usually do not provide useful information.

In contrast, NMF, as a class of matrix factorisations:

$$X \approx WH, \quad W \in M_{n,k}(\mathbb{R}^{\geq 0}), \quad H \in M_{k,p}(\mathbb{R}^{\geq 0})$$

with the integer $k$ chosen such that $k(n+p) < np$, can be interpreted as follows:

- If each row of the matrix $X$ corresponds to a data (e.g. word counts for text data or whiteness of gray image data), then $\boxed{W \text{ is the weight matrix}}$; $\boxed{H \text{ is the feature matrix (each row of } H \text{ is a feature data)}}$;
- If each column of the matrix $X$ corresponds to a data, then $W$ is the feature matrix (each column of $W$ is a feature data); $H$ is the weight matrix.

We are using the former interpretation in our research.

Unlike the canonical matrix factorisations, the **NMF is not unique**.

## 2 NMF Formulation

The **mathematical formulation** of NMF is developed by experts such as Cichocki, et al. into an optimisation problem:

$$\min_{W,H} D(X||WH). \tag{1}$$

Most NMF algorithms are based on a subclass of the Bregman divergence called the $\beta$-**divergence** (Cichocki, Amari 2010):

$$D_\beta(X||Y) = \frac{1}{\beta(\beta-1)} \sum_{i,j} (X_{ij}^\beta - \beta X_{ij} Y_{ij}^{\beta-1} + (\beta-1) Y_{ij}^\beta) \tag{2}$$

where $Y = WH$, $\beta \in \mathbb{R} \setminus \{0,1\}$.

When $\beta = 2$, the loss function (2) becomes the **squared Frobenius norm**:

$$D_2(X||Y) = \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{p}(x_{ij} - y_{ij})^2 =: \frac{1}{2}\|X - Y\|_F^2. \qquad (3)$$

Taking limit $\beta \to 1$, the **Kullback-Leibler (KL) divergence** (also known as the I-divergence) is obtained:

$$D_1(X||Y) = \sum_{i,j}(X_{ij}\ln\frac{X_{ij}}{Y_{ij}} - X_{ij} + Y_{ij}). \qquad (4)$$

Taking limit $\beta \to 0$, the Itakura-Saito (IS) divergence is obtained:

$$D_0(X||Y) = \sum_{i,j}(\frac{X_{ij}}{Y_{ij}} - \ln\frac{X_{ij}}{Y_{ij}} - 1). \qquad (5)$$

Most algorithms only implement the squared Frobenius norm (3) and the Kullback-Leibler divergence (4).

## 3 Classification of NMF Algorithms

Lee, Seung (1999) popularised the research of NMF algorithms.

Since then, we have many NMF publications and three most important references provide the classification of NMF algorithms:

- Yu-Xiong Wang, Yu-Jin Zhang (2013): Nonnegative Marix Factorization: A Comprehensive Review, IEEE Transactions on Knowledge and Data Engineering 25(6), pp. 1336–1353
- Andrzej Cichocki, Rafal Zdunek, Anh Huy Phan, Shun-Ichi Amari, Nonnegative Matrix and Tensor Factorizations: Applications to Exploratory Multi-Way Data Analysis and Blind Source Separation, John Wiley & Sons, Ltd, 2009
- Nicolas Gillis, Nonnegative Matrix Factorization, SIAM, 2020

NMF algorithms are classified based on various variations of NMFs:

(i) **Basic NMF (BNMF)**: A factorisation based on (1).

(ii) **Constraint NMF (CNMF)**: An NMF with general regularisation constraints $J_1(\cdot)$ and $J_2(\cdot)$:

$$D(X||WH) + \alpha_1 J_1(W) + \alpha_2 J_2(H). \tag{6}$$

The constraints can be further classified into sparsity constraints, orthogonality constraints, discriminant constraints and manifold (usually graph-regularised) constraints.

(iii) **Structured NMF (SNMF)**: The NMF with extra matrix structures such as extra weights, convolutive forms, multi-layer factorisation, etc.

(iv) **Generalised NMF (GNMF)**: An extension of NMF to partial negativity (e.g. Semi-BNF), tensors (e.g. NTF), kernels, etc.

BNMF and CNMF Algorithms work by finding the minimum to the cost function (6) for the data $X$.

In particular, the **multiplicative Update (MU)** and variants are the most popular NMF algorithms in dealing with $\beta$-divergence $D_\beta(X||WH)$ (which are implemented by Python, Julia and MATLAB):

$$H \leftarrow H \odot \frac{W^T[(WH)^{\beta-2} \odot X]}{W^T(WH)^{\beta-1}}, \quad W \leftarrow W \odot \frac{[(WH)^{\beta-2} \odot X]H^T}{(WH)^{\beta-1}H^T}.$$

**Coordinate Descent (CD)** is implemented in Python and Julia only for NMF with Frobenius norm (3) but it is not implemented in MATLAB. Instead, MATLAB implements the **Alternative Least Square (ALS)** and variants: For $t = 0, 1, \ldots$

1. Fix $W$, update $H^{(t+1)} \leftarrow \text{argmin}_H D_\beta(W^{(t)}, H^{(t)})$;
2. Fix $H$, update $W^{(t+1)} \leftarrow \text{argmin}_W D_\beta(W^{(t)}, H^{(t+1)})$.

Let $E = X - W^{(0)}H^{(0)}$, $t = 0, 1, 2, \ldots$. Let $W^{(0)}$ and $H^{(0)}$ be the initial guesses. The CD algorithm (`solver='cd'` in sklearn) for the Frobenius norm cost function is:

1. Fix $H$, update $W$: $$\begin{cases} s \leftarrow \max\left\{-[W^{(t)}]_{ir},\ \dfrac{[E(H^{(t)})^T]_{ir}}{[H^{(t)}(H^{(t)})^T]_{rr}}\right\}, \\ [W^{(t+1)}]_{ir} \leftarrow [W^{(t)}]_{ir} + s, \\ E_{i,:} \leftarrow E_{i,:} - s[H^{(t)}]_{r,:} \end{cases}$$

2. Fix $W$, update $H$: $$\begin{cases} s \leftarrow \max\left\{-[H^{(t)}]_{rj},\ \dfrac{[(W^{(t+1)})^T E]_{rj}}{[(W^{(t+1)})^T W^{(t+1)}]_{rr}}\right\}, \\ [H^{(t+1)}]_{rj} \leftarrow [H^{(t)}]_{rj} + s, \\ E_{:,j} \leftarrow E_{:,j} - s[W^{(t+1)}]_{:,r}. \end{cases}$$

The basic MU algorithm (`solver='mu'`, `beta_loss='frobenius'` in sklearn) for Frobenius norm is:

$$\begin{aligned}[H^{(t+1)}]_{rj} &\leftarrow [H^{(t)}]_{rj} \frac{[(W^{(t)})^T X)]_{rj}}{[(W^{(t)})^T W^{(t)} H^{(t)}]_{rj} + \epsilon} \\ [W^{(t+1)}]_{ir} &\leftarrow [W^{(t)}]_{ir} \frac{[X(H^{(t+1)})^T]_{ir}}{[W^{(t)} H^{(t+1)} (H^{(t+1)})^T]_{ir} + \epsilon}.\end{aligned} \tag{7}$$

The basic MU algorithm (`solver='mu'`, `beta_loss='kullback-leibler'` in sklearn) for KL divergence is:

$$\begin{aligned}[H^{(t+1)}]_{rj} &\leftarrow [H^{(t)}]_{rj} \frac{[(W^{(t)})^T ([X]_{rj}/[W^{(t)} H^{(t)}]_{rj})]_{rj}}{[(W^{(t)})^T]_{rj} + \epsilon} \\ [W^{(t+1)}]_{ir} &\leftarrow [W^{(t)}]_{ir} \frac{[([X]_{ir}/[W^{(t)} H^{(t)}]_{ir})(H^{(t+1)})^T]_{ir}}{[(H^{(t+1)})^T]_{ir} + \epsilon}.\end{aligned} \tag{8}$$

Here, $i = 1, ..., n$, $r = 1, ..., k$, $j = 1, ..., p$.

The binary matrix factorisation (BMF), relevant to the black-and-white images (represented by Boolean matrices) that we are considering in this research, is a CNMF (6) with the loss function with "binary" constraints (Zhang et al, 2007):

$$L(W,H) := D_2(W,H) + \alpha_W \|W_{ij}^2 - W_{ij}\|_F^2 + \alpha_H \|H_{ij}^2 - H_{ij}\|_F^2.$$

The algorithm is a variation of the MU algorithm:

$$[H^{(t+1)}]_{rj} \leftarrow [H^{(t)}]_{rj} \frac{[(W^{(t)})^T X)]_{rj} + 3\alpha_H [H^{(t)}]_{rj}^2}{[(W^{(t)})^T W^{(t)} H^{(t)}]_{rj} + 2\alpha_H [H^{(t)}]_{rj}^3 + \alpha_H [H^{(t)}]_{rj} + \epsilon}$$

$$[W^{(t+1)}]_{ir} \leftarrow [W^{(t)}]_{ir} \frac{[X(H^{(t+1)})^T]_{ir} + 3\alpha_W [W^{(t)}]_{ir}^2}{[W^{(t)} H^{(t+1)} (H^{(t+1)})^T]_{ir} + 2\alpha_W [W^{(t)}]_{ir}^3 + \alpha_W [W^{(t)}]_{ir} + \epsilon}.$$

## 4 Software Implementations

Abbreviations: MU-Fro="MU for Frobenius cost function (3)", MU-KL="MU for KL divergence (4)", ADMM="Alternating Direction Method of Multipliers", ALS="Alternating Least Square", ANLS="Alternating Nonnegative Least Squares", CD="Coordinate Descent", PGD="Projected Gradient Descent".

Popular Numerical Analysis Software:

- MATLAB: NMFLibrary (supports MU-Fro, MU-KL, ANLS, PGD, ADMM, ALS, etc.), NMFtoolbox, etc.
- Python: Scikit-learn (supports MU-Fro, MU-KL, CD), libNMF, Nimfa, etc.
- Julia: NMF.jl (supports MU-Fro, MU-KL, CD, ANLS, ALS, etc.)

Python is picked in this research for being the most popular numerical analysis software in artificial intelligence (AI) with three famous machine learning libraries — Scikit-learn, TensorFlow and Torch.

The initialisation strategies for the software implementation of NMF algorithms:

- Random initialisation of the matrices $W$ and $H$ with $Normal(0, \sqrt{\overline{X}/k})$, $k$ is the rank (`init=random'`). We usually fix a seed to make the computation reproducible.
- Nonnegative Double Singular Value Decomposition (NNDSVD) of the matrices $W$ and $H$ (`init='nnsvd'`).
- User defined initial matrices (`init='custom'`).

```
seed=2024
from sklearn.decomposition import NMF
# NMF defaults to Frobenius norm with CD solver
nmf = NMF(n_components=rank, random_state=seed,
  init='random', tol=1e-10)
nmf.fit(X)
W = nmf.fit_transform(A)
H = nmf.components_
```

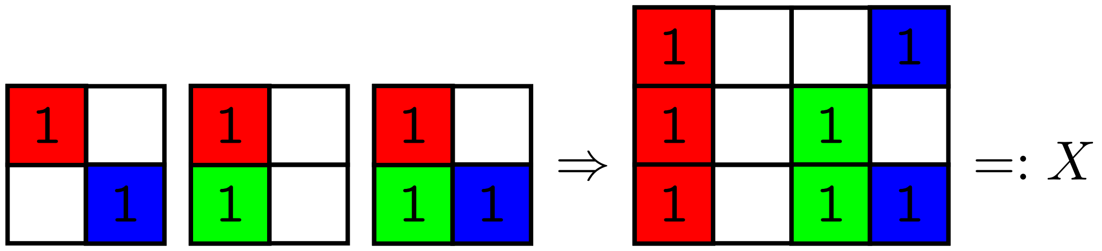
```
import nimfa
alphaW = 0.5
alphaH = 1.5
bmf = nimfa.Bmf(X, seed="nndsvd", rank=2, max_iter=1200,
  lambda_w=alphaW, lambda_h=alphaH)
bmf_fit = bmf()
W = bmf_fit.basis()
H = bmf_fit.coef()
```

## 5 Empirical Analysis of Boolean Matrices

Our framework: Study the feature extraction abilities of MU-Fro, MU-KL, BMF for three case studies of black-and-white images:

1. A set of three $2 \times 2$ images with rank 2.
2. MNIST images converted to black-and-white images (rank unknown but there are 10 digits. So rank is presumed to be 10)
3. Black-and-white emojis (rank is unknown).

**Case Study 1**: Boolean matrix from three 2x2 black-and-white images.



The Boolean matrix $X$ is rank 2 and has a Boolean matrix decomposition given below:

$$X = \underbrace{\begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 1 & 1 \end{bmatrix}}_{W_X} \underbrace{\begin{bmatrix} 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 \end{bmatrix}}_{H_X}.$$

The Boolean matrix $H$ is feature matrix while the Boolean matrix $W$ is the weight matrix.

The MU-Fro with a seeded random initialisation:

$$X \approx \underbrace{\begin{bmatrix} 1.1046 & 0.0084 \\ 0.1037 & 0.6814 \\ 0.8544 & 0.4878 \end{bmatrix}}_{W_1} \underbrace{\begin{bmatrix} 0.7641 & 0. & 0.0815 & 0.999 \\ 1.1364 & 0. & 1.6071 & 0. \end{bmatrix}}_{H_1} = \begin{bmatrix} 0.8536 & 0. & 0.1036 & 1.1036 \\ 0.8536 & 0. & 1.1036 & 0.1036 \\ 1.2071 & 0. & 0.8536 & 0.8536 \end{bmatrix}.$$

The MU-KL a seeded random initialisation:

$$X \approx \underbrace{\begin{bmatrix} 1.4354 & 0. \\ 0. & 0.5811 \\ 1.0766 & 0.4358 \end{bmatrix}}_{W_2} \underbrace{\begin{bmatrix} 0.5971 & 0. & 0. & 0.7962 \\ 1.475 & 0. & 1.9667 & 0. \end{bmatrix}}_{H_2} = \begin{bmatrix} 0.8571 & 0. & 0. & 1.1429 \\ 0.8571 & 0. & 1.1429 & 0. \\ 1.2857 & 0. & 0.8571 & 0.8571 \end{bmatrix}$$

The BMF ($\alpha_W = 0.5$, $\alpha_H = 1.5$) with an NNDSVD initialisation gives

$$X \approx \underbrace{\begin{bmatrix} 0. & 1.1432 \\ 1.1769 & 0.6332 \\ 1.1732 & 1.1431 \end{bmatrix}}_{W_3} \underbrace{\begin{bmatrix} 0. & 0. & 0.8463 & 0. \\ 0.9579 & 0. & 0.0064 & 0.7715 \end{bmatrix}}_{H_3} = \begin{bmatrix} 1.095 & 0. & 0.0073 & 0.8819 \\ 0.6065 & 0. & 1. & 0.4885 \\ 1.095 & 0. & 1.0001 & 0.8819 \end{bmatrix}$$

In contrast, consider a rank-2 nonnegative matrix and the usual matrix multiplication of $W_X$ and $H_X$:

$$X_2 = \begin{bmatrix} 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 \\ 2 & 0 & 1 & 1 \end{bmatrix}$$

The MU-Fro with a seeded random initialisation gives the **equality**:

$$X_2 = \underbrace{\begin{bmatrix} 1.2226 & 0. \\ 0. & 0.5759 \\ 1.2226 & 0.5759 \end{bmatrix}}_{W_1} \underbrace{\begin{bmatrix} 0.8179 & 0. & 0. & 0.8179 \\ 1.7365 & 0. & 1.7365 & 0. \end{bmatrix}}_{H_1}$$

The MU-KL a seeded random initialisation gives the **equality**:

$$X_2 = \underbrace{\begin{bmatrix} 1.3488 & 0. \\ 0. & 0.5326 \\ 1.3488 & 0.5326 \end{bmatrix}}_{W_2} \underbrace{\begin{bmatrix} 0.7414 & 0. & 0. & 0.7414 \\ 1.8776 & 0. & 1.8776 & 0. \end{bmatrix}}_{H_2}$$

The BMF ($\alpha_W = 0.5$, $\alpha_H = 1.5$) with an NNDSVD initialisation performs badly by only providing an approximation:

$$X_2 \approx \underbrace{\begin{bmatrix} 4.9797 & 0.8749 \\ 0. & 1.232 \\ 4.4807 & 2.113 \end{bmatrix}}_{W_3} \underbrace{\begin{bmatrix} 0. & 0. & 0. & 0.1898 \\ 0.9777 & 0. & 0.4182 & 0.0476 \end{bmatrix}}_{H_3} = \begin{bmatrix} 0.8553 & 0. & 0.3659 & 0.9867 \\ 1.2045 & 0. & 0.5153 & 0.0587 \\ 2.0657 & 0. & 0.8837 & 0.9509 \end{bmatrix}$$

The features $H_1$, $H_2$ and $H_3$ obtained from Frobenius-norm based NMF, KL divergence based NMF and BMF lose information due to the difference between

$$\textbf{Boolean addition: } 1+1=1$$

and

$$\textbf{real number addition: } 1+1=2.$$

It is interesting to observe that the feature $H_1$ from Frobenius-norm and the feature $H_2$ from KL divergence based NMF are closer to the actual feature $H$ compare to the feature $H_3$ from BMF despite the fact that BMF were developed to deal with binary matrix factorisation.

**Case Study 2**: 42000-row MNIST data (from `https://github.com/sbussmann/kaggle-mnist`) with $28 \times 28$-columns of squared 8-bit grey images (grey colours ranging from 0 to 255).

The MNIST data is converted to a Boolean matrix using 100 as threshold:

$$X_B = [[\text{MINST}]_{ij} > 100].$$

Figure 1 shows that after applying the threshold of 100, the Boolean matrices in the second row are obtained and they are reasonable approximation to the original grey images.

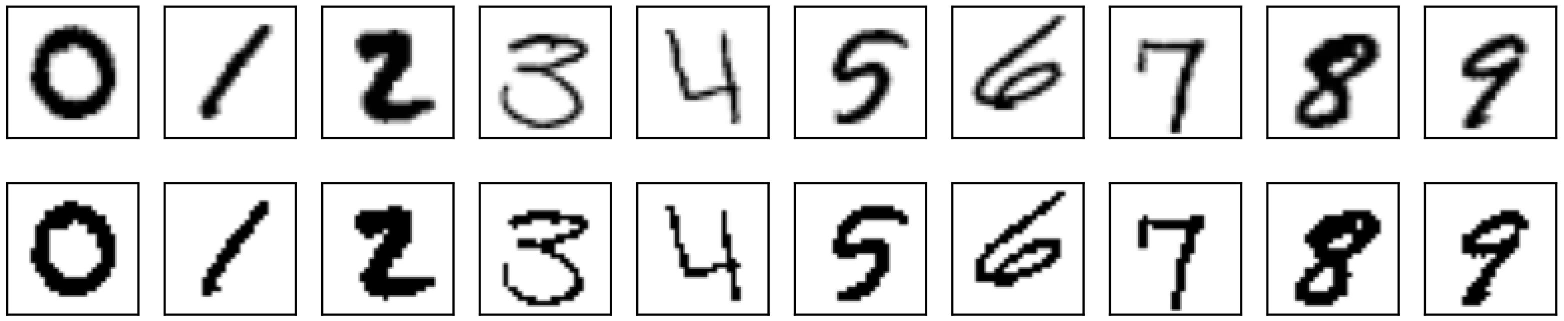


**Figure 1.**

If we choose a larger threshold, the digit image will be thinner while a smaller threshold will lead to a thicker digit image.

Since

1. the Frobenius-norm NMF is related to kernel K-mean clustering (Ding, He, Simon, 2005); and
2. we expect the features of the MNIST data to be the 10 digits (corresponding to 10 clusters of digits 0 to 9),

the rank $k$ is chosen to be 10.

Custom initialisation:

1. $W_0 = 10 \times 10$ identity matrix;
2. $H_0 =$ the rows of Figure 1.

Applying the three NMF algorithms with custom initialisation, the features obtained are shown in three rows in Figure 2.

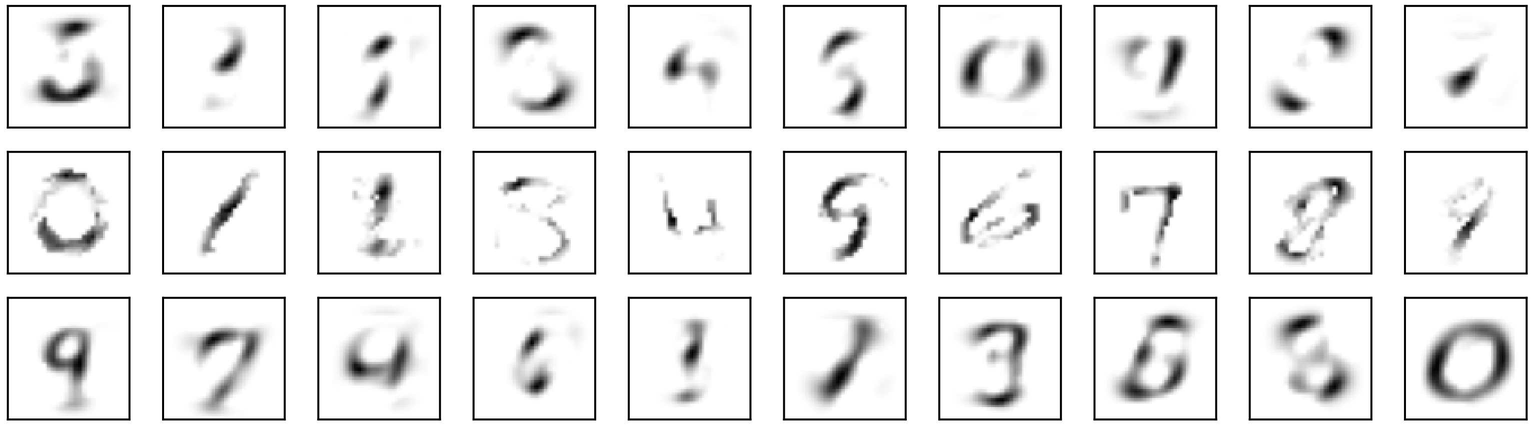


**Figure 2.**

Row 1 (MU-Fro): features are strokes of the digits

Row 2 (MU-KL): 0 to 9 digits seems to be preserved.

Row 3 (BMF): 9, 7, 1, 3, 6, 8 and 0 seems recognisable.

Applying the three NMF algorithms with seeded random initialisation, the features obtained are shown in three rows in Figure 3.

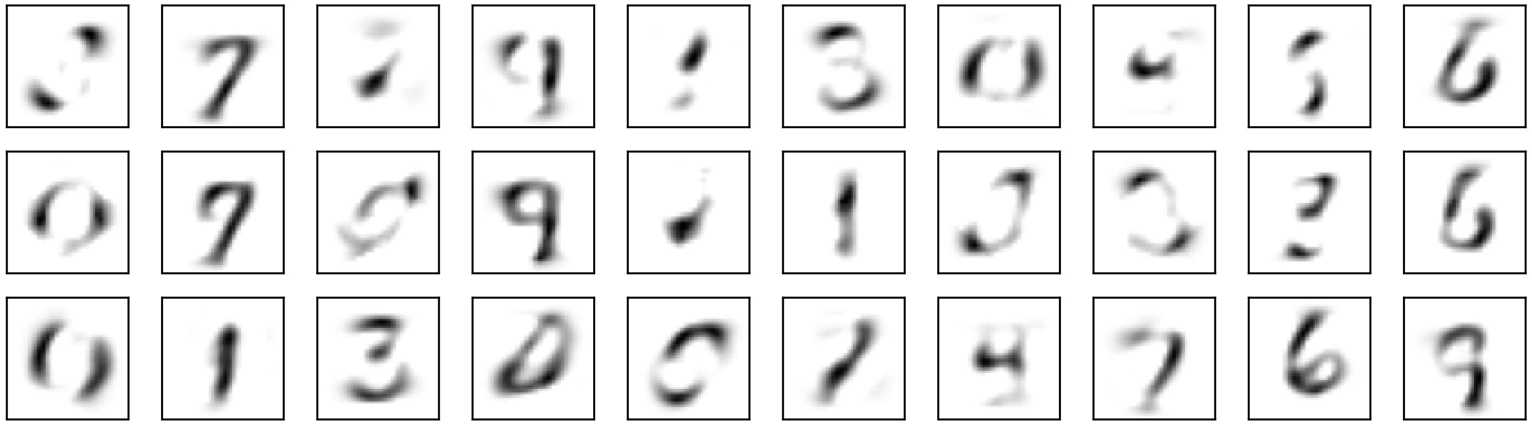


**Figure 3**.

Row 1 (MU-Fro): 7, 9, 3, 0 and 6 seem recognisable, other are strokes

Row 2 (MU-KL): 0, 7, 9, 1 and 6 seem recognisable, other are strokes

Row 3 (BMF): 0, 1, 3, 7, 6 and 9 seem recognisable, other are strokes

**Case Study 3**: A 175 non-facial black-and-white emojis of the size $50 \times 50$. There is no prior information on the rank.

To determine the rank, we analyse the total within sum of squared errors of the k-means clustering of the data and obtain the Figure 4.

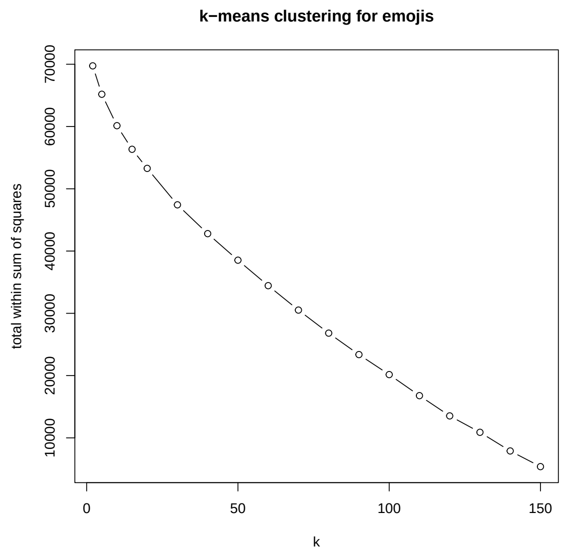
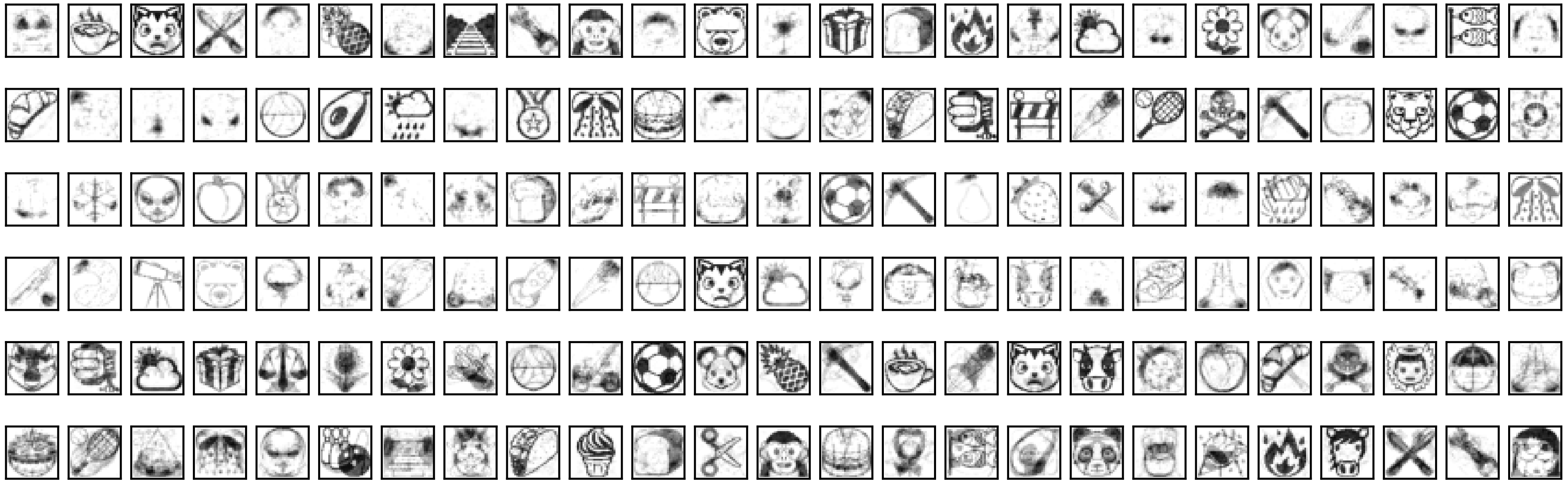


**Figure 4**.

There is no elbow in the curve indicating no clear clusters. We pick the rank of 50 which is close to 50% reduction in the error in Figure 4.

Applying the three NMF algorithms with seeded random initialisation, we obtain the features $H$ in **Figure 5** below.



Rows 1–2 (MU-Fro): Quite many original emojis with a few "parts"

Rows 3–4 (MU-KL): More "parts" and original emojis are blurred.

Rows 5–6 (BMF): Quite many original emojis with a few "parts"

Common recognisable features: cats, balls

## 6 Conclusion

We have achieved our objectives

1. The mathematical formulation of NMF is an optimisation problem based on divergence measure.
2. The NMF algorithms are all various kind of alternating optimisation techniques.
3. Not all "fast NMF algorithms" are implemented due to more complex formulas. Only MU is the most widely implemented.
4. NMF is still useful for black-and-white images despite the weights $W$ and features $H$ are no longer Boolean/binary. It can be used to identify "recognisable" features: e.g. strokes of drawings, special drawings (e.g. the cat, the ball in the emojis). Advertisement companies can use NMF to select "attracting" pictures.

Lessons learned:

- NMF is unsuitable to be included in undergraduate algebra since all NMF algorithms are based on divergence rather than algebra.
- NMF is unsuitable for dimensional reduction due to the non-uniqueness (compare to PCA).

Things to explore:

- Improvements of BMF for black-and-white images since BMF's constraint is not satisfactory with our case study 1.
- NMF in clustering, recommender system;
- Generative models for black-and-white images, etc.

# Q&A

## VISIT US HERE

IPSR Institute of Postgraduate Studies and Research
*Intellectual Pursuits Through Sustainable Research*

utar.edu.my

ipsr.utar.edu.my

research.utar.edu.my