計算科学・量子計算における情報圧縮

Data Compression in Computational Science and Quantum Computing

2022.10.27

#4:特異値分解のテンソルへの拡張＋応用

Application of SVD and generalization to tensors

理学系研究科　大久保　毅

Graduate School of Science, **Tsuyoshi Okubo**

I put (URLs of) recordings of previous lectures on ITC-LMS.
You can also download lecture slide from ITC-LMS.

Today's topic

1. Computational science, quantum computing, and data compression
2. Review of linear algebra
3. Singular value decomposition
4. Application of SVD and generalization to tensors
5. Entanglement of information and matrix product states
6. Application of MPS to eigenvalue problems
7. Tensor network representation
8. Data compression in tensor network
9. Tensor network renormalization
10. Quantum mechanics and quantum computation
11. Simulation of quantum computers
12. Quantum-classical hybrid algorithms and tensor network
13. Quantum error correction and tensor network

Outline

- Low-rank approximation
  - Low-rank approximation by SVD (Cont.)
  - Generalization to tensors
    - Tensor network diagram
    - CP decomposition
    - Tucker decomposition
- Application of the low-rank approximations to images
- (Toward another “low-rank” approximation)

Low-rank approximation

Low-rank approximation

Low-rank approximation（低ランク近似）

Find an approximate matrix

$$A \simeq \tilde{A}$$

with lower rank:

$$\mathrm{rank}(\tilde{A}) > \mathrm{rank}(A)$$

Through the low-rank approximation,
we can reduce the amount of data.

An example of data compressions.

Low-rank approximation by SVD

Consider a matrix obtained by neglecting smaller singular values

$$A = \bar{U}\Sigma_{r\times r}\bar{V}^\dagger \quad \Rightarrow \quad \tilde{A} = \tilde{U}\Sigma_{k\times k}\tilde{V}^\dagger \qquad (k < r)$$

$$\Sigma_{r\times r} = \mathrm{diag}(\sigma_1, \sigma_2, \ldots, \sigma_r)$$
$$\bar{U} = (\vec{u}_1, \vec{u}_2, \ldots, \vec{u}_r)$$
$$\bar{V} = (\vec{v}_1, \vec{v}_2, \ldots, \vec{v}_r)$$

$$\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_r > 0$$

$$\mathrm{rank}(A) = r$$

$$\Sigma_{k\times k} = \mathrm{diag}(\sigma_1, \sigma_2, \ldots, \sigma_k)$$
$$\tilde{U} = (\vec{u}_1, \vec{u}_2, \ldots, \vec{u}_k)$$
$$\tilde{V} = (\vec{v}_1, \vec{v}_2, \ldots, \vec{v}_k)$$

Keep the largest k singular values
(and corresponding singular vectors).

$$\mathrm{rank}(\tilde{A}) = k < r$$

This approximation is one of the low rank approximation.

* For this approximation, we need $\mathrm{O}(MNk)$ calculations for SVD of a $M\times N$ matrix.

Norm of matrices $\|A\|$

There are two popular norms:

(1) **Frobenius norm** （フロベニウス　ノルム）

$$\|A\|_F = \sqrt{\sum_{i,j} |A_{ij}|^2} = \sqrt{\mathrm{Tr}(A^\dagger A)}$$

*Trace（対角和）

$$\mathrm{Tr}(X) = \sum_i X_{ii}$$

(2) **Operator norm**（作用素ノルム）

$$\|A\|_O = \inf\{c \geq 0; \|A\vec{x}\| \leq c\|\vec{x}\|\}$$
$$= \sigma_1(A)$$

*inf =infimum（下限）

*We define the norm for a vector as

$$\|\vec{x}\| = \sqrt{\sum_i |x_i|^2}$$

By using these norms, we define the distance between matrices:

$$\|A - \tilde{A}\|$$

Accuracy of low rank approximation by SVD

(A part of) Eckart-Young-Mirsky Theorem

C. Eckart, and G. Young, Psychometrika **1**, 211 (1936).
L. Mirsky, Q. J. Math. **11**, 50 (1960).

For $A : M \times N$

$$\min\{\|A - B\|_F : \mathrm{rank}(B) = k\} = \sqrt{\sum_{i=k+1}^{\min(N,M)} \sigma_i^2}$$

$$\min\{\|A - B\|_O : \mathrm{rank}(B) = k\} = \sigma_{k+1}$$

Because the k-rank approximation by SVD gives

$$\|A - \tilde{A}\|_F = \sqrt{\sum_{i=k+1}^{\min(N,M)} \sigma_i^2}, \qquad \|A - \tilde{A}\|_O = \sigma_{k+1}$$

it is an "optimal" approximation with rank k.

Short proof of the theorem: Frobenius norm (optional)

*This proof is based on
"システム制御のための数学（１）"太田快人 著

From the inequality of singular values for matrix addition (property 5),

for j=1,... ,$\min(M,N)$ $\quad (\mathrm{rank}(B) = k)$

$$\sigma_{j+k}(A) = \sigma_{j+k}((A-B)+B) \leq \sigma_j(A-B)$$

Property 5

By taking a square and summing up them

$$\sum_{i=k+1}^{\min(M,N)} \sigma_i^2(A) \leq \sum_{j=1}^{\min(M,N)} \sigma_j^2(A-B) = \|A-B\|_F^2$$

*Note $\sigma_j(A) = 0 \quad (j > \mathrm{rank}(A))$

Short proof of the theorem: operator norm (optional)

*This proof is based on
"システム制御のための数学（１）"太田快人 著

From the min-max theorem of singular values (property 4),

$$(\mathrm{rank}(B) = k)$$

$$\sigma_{k+1}(A) \leq \max_{\vec{x}\in\ker(B),\|\vec{x}\|=1} \|A\vec{x}\| = \max_{\vec{x}\in\ker(B),\|\vec{x}\|=1} \|(A-B)\vec{x}\|$$

Property4 with
$\boldsymbol{S} = \mathrm{img}(B^\dagger)$
$\boldsymbol{S}^\perp = \ker(B)$

$B\vec{x} = 0 \quad (\vec{x} \in \ker(B))$

$$\leq \max_{\|\vec{x}\|=1} \|(A-B)\vec{x}\| = \|A-B\|_O$$

Expand the vector space

Definition of the operator norm

Relation to **principal component analysis** （主成分分析）

Data set $\{X_{ij}\}$: X: $N \times M$ matrix

i = index for data, j = data type (coordinates, momentum, ...)

* Suppose the mean of data is 0: $\sum_i X_{ij} = 0$

Covariance matrix（共分散行列）: $C = X^T X$

Principal component analysis (PCA):

Data compression through the spectrum decomposition of C.

$C = V \Lambda V^T$ Λ: diagonal matrix, $\Lambda_{ii} = \lambda_i \geq 0$

$V = (\vec{v}_1, \vec{v}_2, \cdots, \vec{v}_N)$

$\vec{v}_i$ corresponding to large λ_i contains important information.

By construction, λ and V are related to SVD of X!

$X = U \Sigma V^T$ $, \sigma_i = \sqrt{\lambda_i}$

PCA can be regarded as the low-rank approximation of X.

Low-rank approximation: generalization to tensors

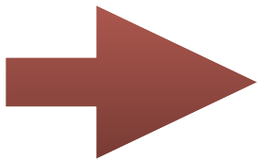
Scalar, Vector, Matrix, Tensor,...

Scalar: c Number

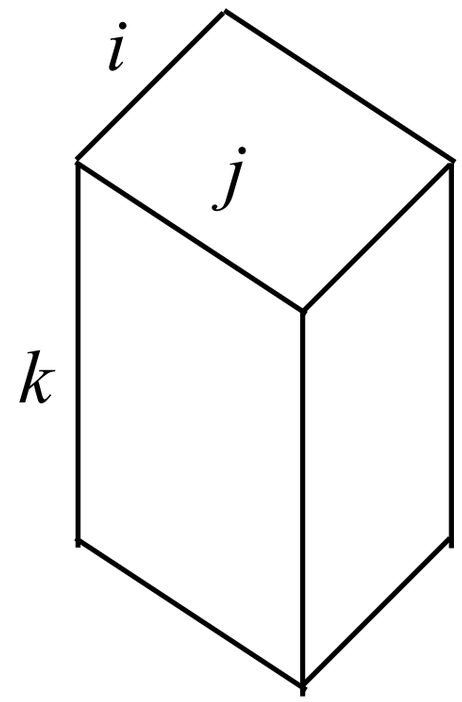
Vector: v_i One dimensional array of numbers

Matrix: M_{ij} Two dimensional array of numbers

Tensor: $T_{ijk\ldots}$ Higher dimensional array of numbers



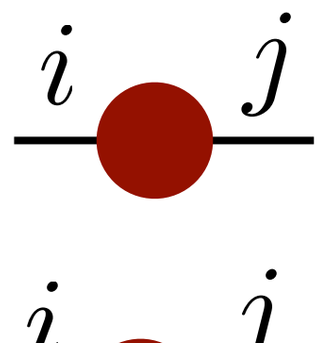
Scalar: 0-dim. tensor
Vector: 1-dim. tensor
Matrix: 2-dim. tensor



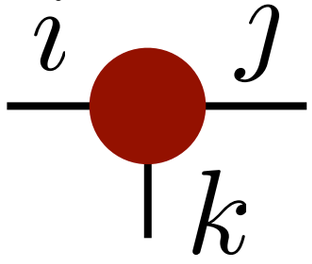
Graphical representations for tensor network

- Vector $\vec{v} : v_i$

- Matrix $M : M_{i,j}$

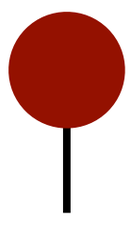


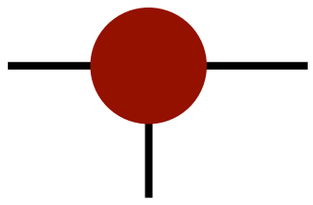
- Tensor $T : T_{i,j,k}$



*** n-rank tensor = n-leg object**

When indices are not presented in a graph, it represent a tensor itself.

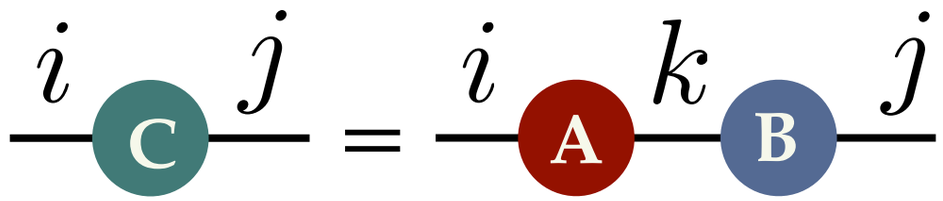
$\vec{v} =$ 

$T =$ 

Graphical representations for tensor network

Matrix product

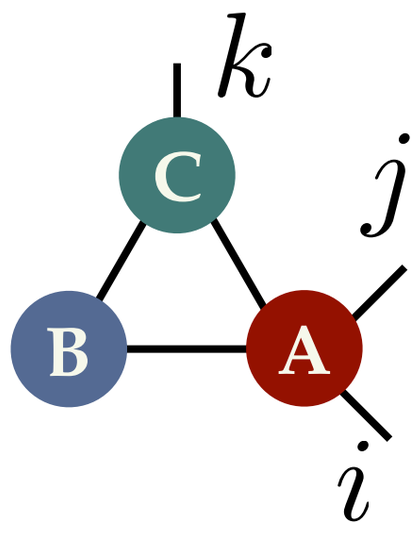
$$C_{i,j} = (AB)_{i,j} = \sum_k A_{i,k} B_{k,j}$$



$$C = AB$$

Generalization to tensors

$$\sum_{\alpha,\beta,\gamma} A_{i,j,\alpha,\beta} B_{\beta,\gamma} C_{\gamma,k,\alpha}$$



Contraction of a network = Calculation of a lot of multiplications

Low-rank approximation: generalization to tensor

Tensor: $T_{ijk\ldots}$

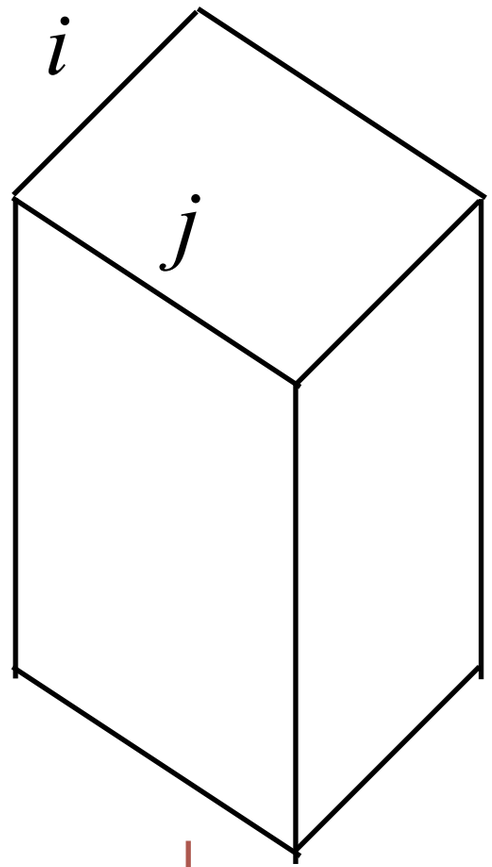
Naive application of SVD:

Make a matrix by dividing indices into two parts.

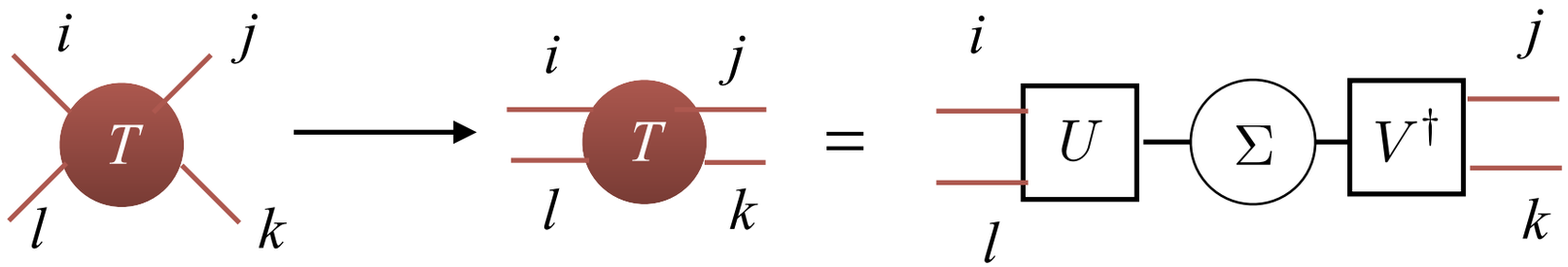
$$T_{ijkl} \to T_{(il),(jk)} = T_{IJ}$$

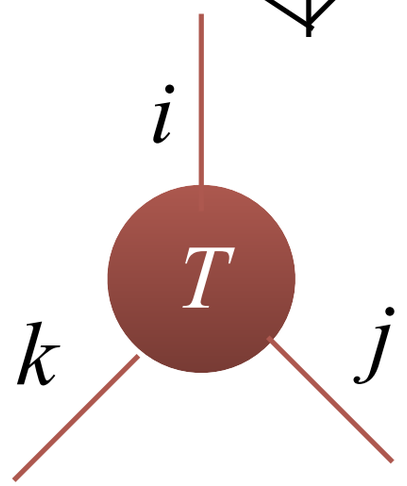
*Example

$$i,j,k,l = 0,1 \;\to\; \begin{array}{cc} (i,l) & I \\ (0,0) & \to 0 \\ (0,1) & \to 1 \\ (1,0) & \to 2 \\ (1,1) & \to 3 \end{array}$$



Then apply SVD (and low rank approximation).



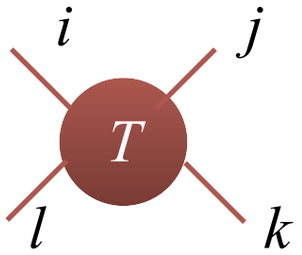


Note: The result depends on the initial mapping to a matrix.

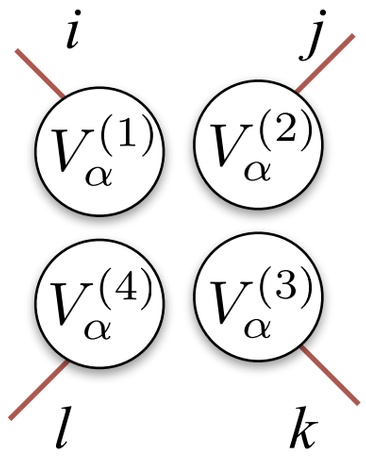
CP decomposition

Review: T. G. Kolda et al, SIAM Review **51**, 455 (2009)

CP (Canonical Polyadic) decomposition: Hitchcock (1927)



$$= \sum_{\alpha=1}^{r}$$



$$T_{ijkl} = \sum_{\alpha=1}^{r} (V_\alpha^{(1)})_i (V_\alpha^{(2)})_j (V_\alpha^{(3)})_k (V_\alpha^{(4)})_l$$

Simplified notation: $(T = \sum_{\alpha=1}^{r} \boldsymbol{V}_\alpha^{(1)} \circ \boldsymbol{V}_\alpha^{(2)} \circ \boldsymbol{V}_\alpha^{(3)} \circ \boldsymbol{V}_\alpha^{(4)})$

$\boldsymbol{V}_\alpha^{(1)}$: Vector of $(V_\alpha^{(1)})_i$

min (r) = tensor rank

*Determining tensor rank is NP-hard problem.

Low-rank approximation

$$T_{ijkl} \approx \sum_{\alpha=1}^{r'} (\tilde{V}_\alpha^{(1)})_i (\tilde{V}_\alpha^{(2)})_j (\tilde{V}_\alpha^{(3)})_k (\tilde{V}_\alpha^{(4)})_l$$

$r' < r$

rank-r' approximation

Difficulty in low-rank CP approximation

T. G. Kolda et al, SIAM Review **51**, 455 (2009)

$$T_{ijkl} = \sum_{\alpha=1}^{r} (V_\alpha^{(1)})_i (V_\alpha^{(2)})_j (V_\alpha^{(3)})_k (V_\alpha^{(4)})_l \quad \Rightarrow \quad T_{ijkl} \approx \sum_{\alpha=1}^{r'} (\tilde{V}_\alpha^{(1)})_i (\tilde{V}_\alpha^{(2)})_j (\tilde{V}_\alpha^{(3)})_k (\tilde{V}_\alpha^{(4)})_l$$

1. The best r'-rank approximation is not necessarily given by neglecting several terms in the original CP decomposition.

   $\left\{\tilde{V}_\alpha^n\right\}$ can be different from $\{V_\alpha^n\}$.

Cf. T. G. Kolda, SIAM J. Matrix Anal. Appl., **23**, 243 (2001).

2. The best r'-rank approximation problems become ill-posed in some cases.

rank 3 tensor: $T = \boldsymbol{a}_1 \circ \boldsymbol{b}_1 \circ \boldsymbol{c}_2 + \boldsymbol{a}_1 \circ \boldsymbol{b}_2 \circ \boldsymbol{c}_1 + \boldsymbol{a}_2 \circ \boldsymbol{b}_1 \circ \boldsymbol{c}_1$ $\quad (\boldsymbol{a}_1, \boldsymbol{a}_2) = 0$, $(\boldsymbol{b}_1, \boldsymbol{b}_2) = 0$, $(\boldsymbol{c}_1, \boldsymbol{c}_2) = 0$

rank 2 tensor: $\tilde{T} = n\left(\boldsymbol{a}_1 + \frac{1}{n}\boldsymbol{a}_2\right) \circ \left(\boldsymbol{b}_1 + \frac{1}{n}\boldsymbol{b}_2\right) \circ \left(\boldsymbol{c}_1 + \frac{1}{n}\boldsymbol{c}_2\right) - n\boldsymbol{a}_1 \circ \boldsymbol{b}_1 \circ \boldsymbol{c}_1$

$$\Rightarrow \quad \|T - \tilde{T}\| \propto \frac{1}{n}$$

Error in rank-2 approximation can be arbitrarily small.
However, each term diverges as we decrease the error.

Cf. V. de Silva et al. SIAM J. Matrix Anal. Appl., **30**, 1084 (2008).

Tucker decomposition: generalization of SVD

Review: T. G. Kolda et al, SIAM Review **51**, 455 (2009)

Tucker decomposition:

(Tucker (1963))

$U^{(i)}$: Factor matrix (usually unitary)

S :Core tensor

$$T_{ijkl} = \sum_{i'=1}^{I} \sum_{j'=1}^{J} \sum_{k'=1}^{K} \sum_{l'=1}^{L} S_{i'j'k'l'} U^{(1)}_{ii'} U^{(2)}_{jj'} U^{(3)}_{kk'} U^{(4)}_{ll'}$$

*If S is "diagonal", Tucker decomposition becomes CP decomposition.

Low "rank" approximation

$$T_{ijkl} = \sum_{i'=1}^{I'} \sum_{j'=1}^{J'} \sum_{k'=1}^{K'} \sum_{l'=1}^{L'} \tilde{S}_{i'j'k'l'} \tilde{U}^{(1)}_{ii'} \tilde{U}^{(2)}_{jj'} \tilde{U}^{(3)}_{kk'} \tilde{U}^{(4)}_{ll'}$$

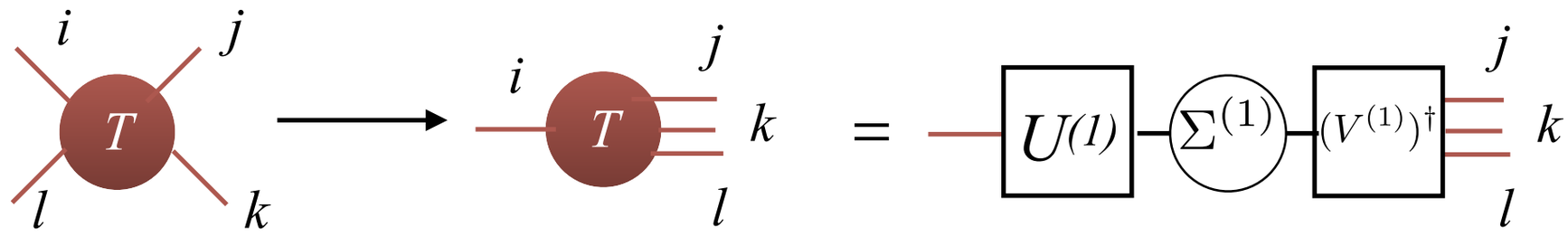
$I' < I, \quad J' < J, \quad K' < K, \quad L' < L$

rank-(I',J', K', L') approximation

Higher order SVD (HOSVD)

L. De Lathauwer et al, SIAM J. Matrix Anal. & Appl., **21**, 1253 (2000)

Define a factor matrix from matrix SVD:



Core tensor is calculated as

$$S_{i'j'k'l'} \equiv \sum_{ijkl} T_{ijkl} (U^{(1)})^{\dagger}_{i'i} (U^{(2)})^{\dagger}_{j'j} (U^{(3)})^{\dagger}_{k'k} (U^{(4)})^{\dagger}_{l'l}$$

Properties of the core tensor

$$S^{*}_{:,i_n=\alpha,:,:} \cdot S_{:,i_n=\beta,:,:} = \begin{cases} 0 & (\alpha \neq \beta) \\ (\sigma^{(n)}_{\alpha})^2 & (\alpha = \beta) \end{cases}$$

Dot product

$$A \cdot B \equiv \sum_{i,j,k,l} A_{ijkl} B_{ijkl}$$

Generalization of the diagonal matrix Σ in matrix SVD.

* Low-rank approximation based on HOSVD is not optimal.

Application of low rank approximation

Sample codes are uploaded on ITC-LMS.

Image_SVD.zip

(python and jupyter notebook codes)

Image compression: grayscale image

Image: 1024 × 768 pixels

768 x1024 matrix A

$$\mathrm{rank}(A) = 768$$

Amount of data=786,432

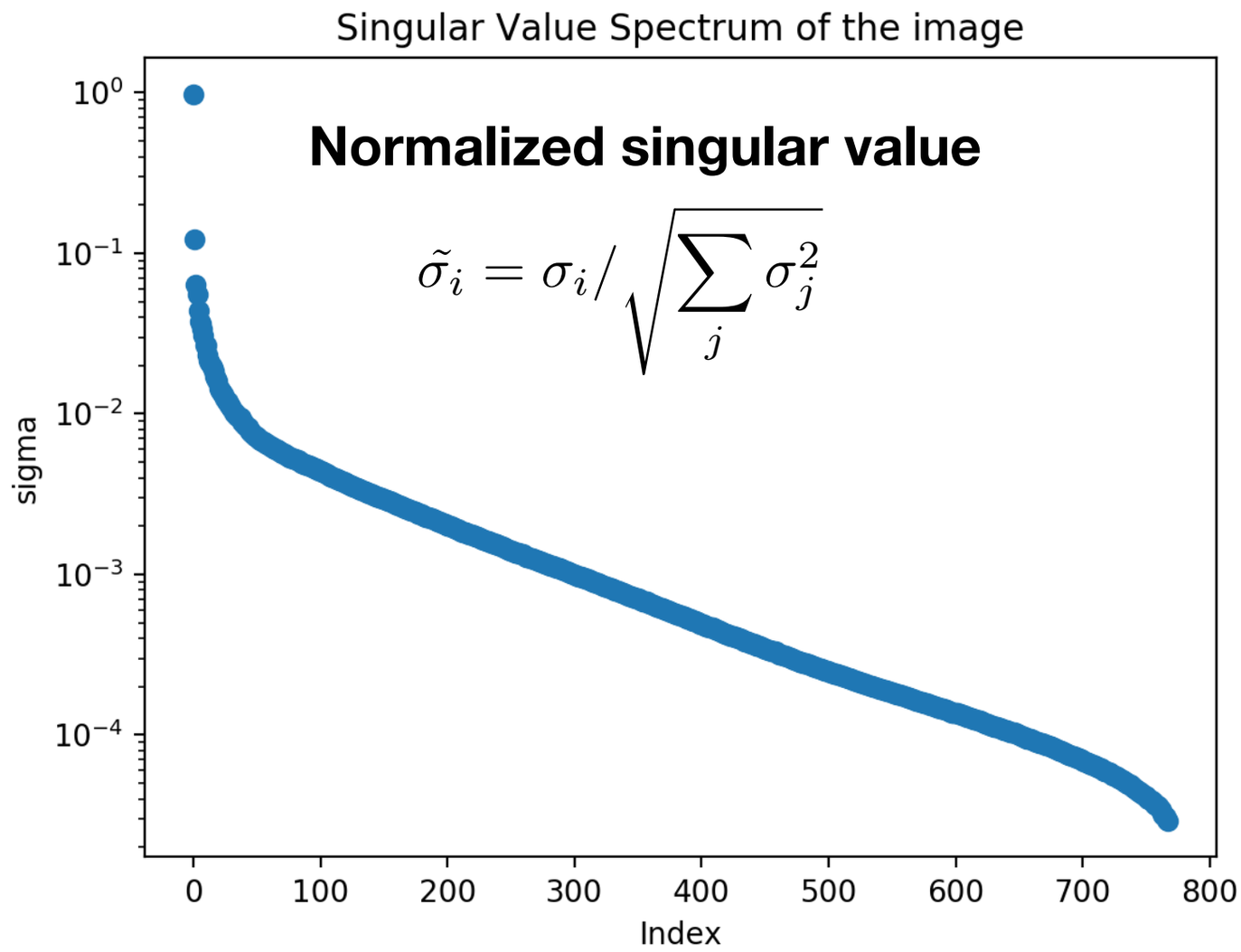
Perform SVD of A: $A = U\Sigma V^{\dagger}$

$\mathrm{rank}(\chi)$ approximation

Amount of data=(768 +1024 + 1)×χ

Image compression: grayscale image

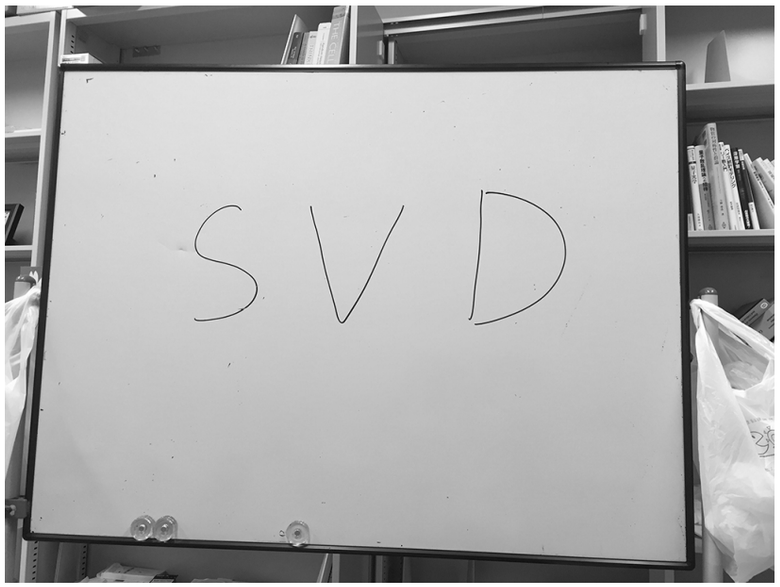
Image: 1024 × 768 pixels

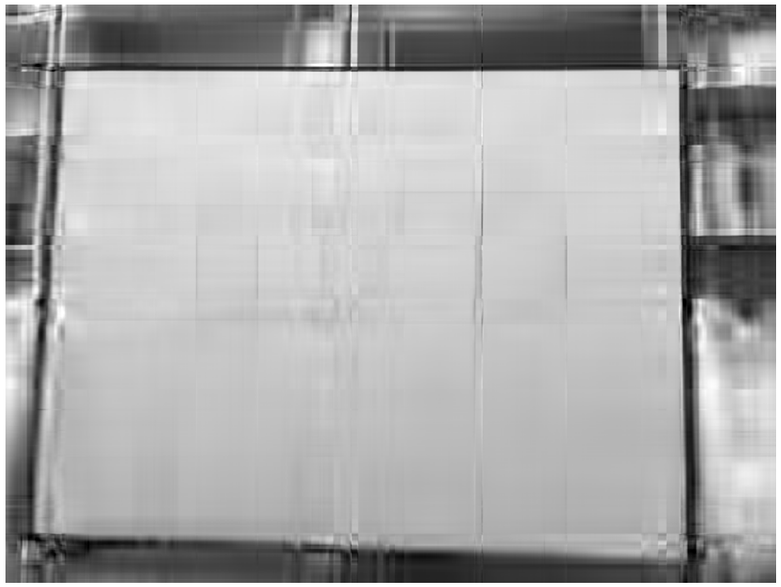


For k-rank approximation, the relative error is calculated as

$$\frac{\|A - \tilde{A}\|_F}{\|A\|_F} = \sqrt{\sum_{i=k+1} \tilde{\sigma}_i^2}$$

Image compression: grayscale image





Rank: $\chi = 768$ $\chi = 100$ $\chi = 10$

Data: **786,432** **179,300** **179,30**

(Original)

Image compression: color image

Image: 1024 × 768 pixels

768 x1024x3 tensor T

Amount of data=2,359,296

* Sub matrices for RGB colors

$$R_{ij} = T_{ij1}, \quad G_{ij} = T_{ij2}, \quad B_{ij} = T_{ij3}$$

Two image compressions:

Perform SVD for R, G, B → $\mathrm{rank}(\chi)$ approximation for RGB matrices

Amount of data=3×(768 +1024 + 1)×χ

Perform HOSVD for T → $\mathrm{rank}-(\chi', \chi', 3)$ approximation

Amount of data= (768 +1024 + 3χ)×χ

Image compression: color image1

Original

Data: **2,359,296**

~10%
Compression

SVD

$\chi = 50$

268,950

HOSVD

$\chi' = 100$

209,200

Image compression: color image2

Original

Data: **2,359,296**

~10%
Compression

SVD

$\chi = 50$

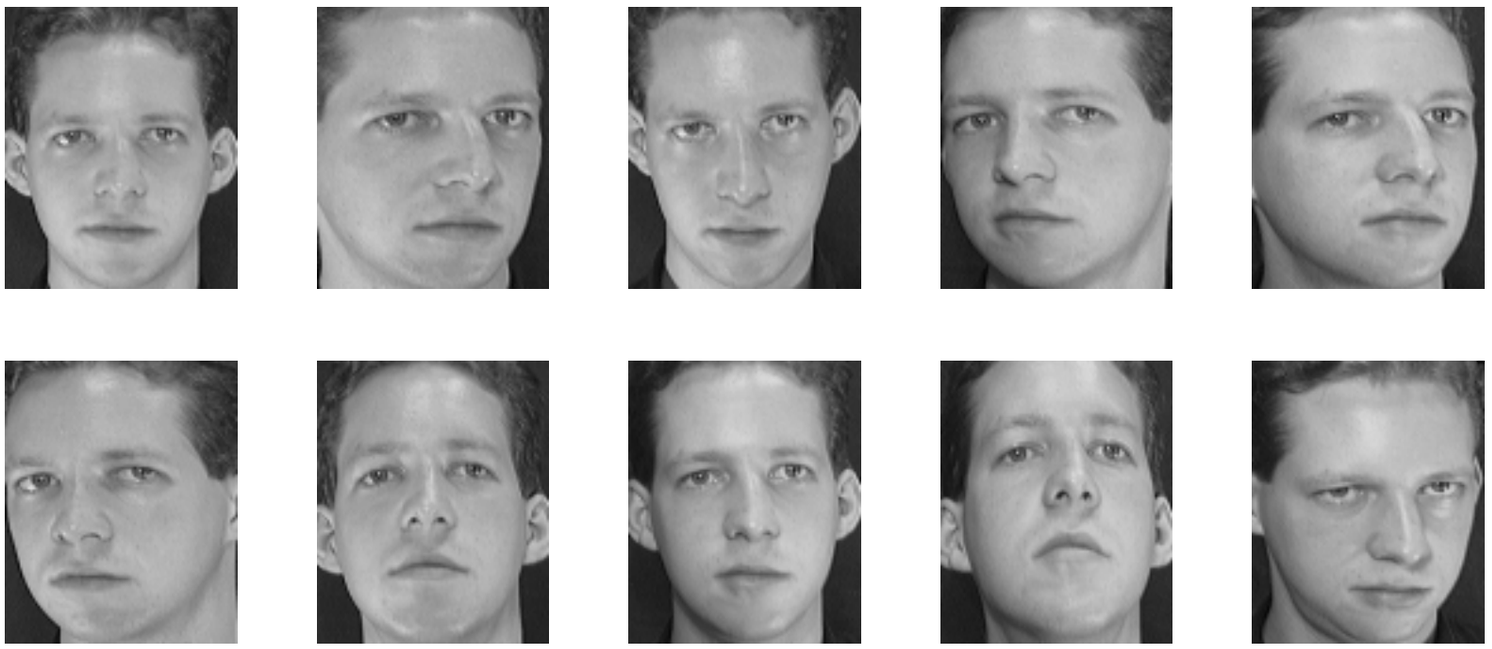
268,950

HOSVD

$\chi' = 100$

209,200

Image compression: multi images



92×122 pixels 10 images

Images were taken from ORL Database of Faces, AT&T Laboratories Cambridge

92 x122 x 10 tensor T

Amount of data=112,240

by HOSVD

$\mathrm{rank}-(\chi, \chi, \chi')$ approximation

Amount of data=
$(92 + 122) \times \chi + 10 \times \chi' + \chi^2 \chi'$

Image compression: multi images

		Data
Original		112,240
$\chi = 30$ $\chi' = 10$		15,520
$\chi = 30$ $\chi' = 9$		14,610
$\chi = 30$ $\chi' = 5$		10,970

Image compression: multi images (color)

From COIL-100 dataset = 128 x 128 x 3 x 20 x 72 tensor

Pixel color object direction

Objects:

Several directions:

Amount of data=70,778,880

by HOSVD

$$\mathrm{rank}\text{–}(\chi_p, \chi_p, \chi_c, \chi_o, \chi_r)$$

approximation

Image compression: multi images (color)

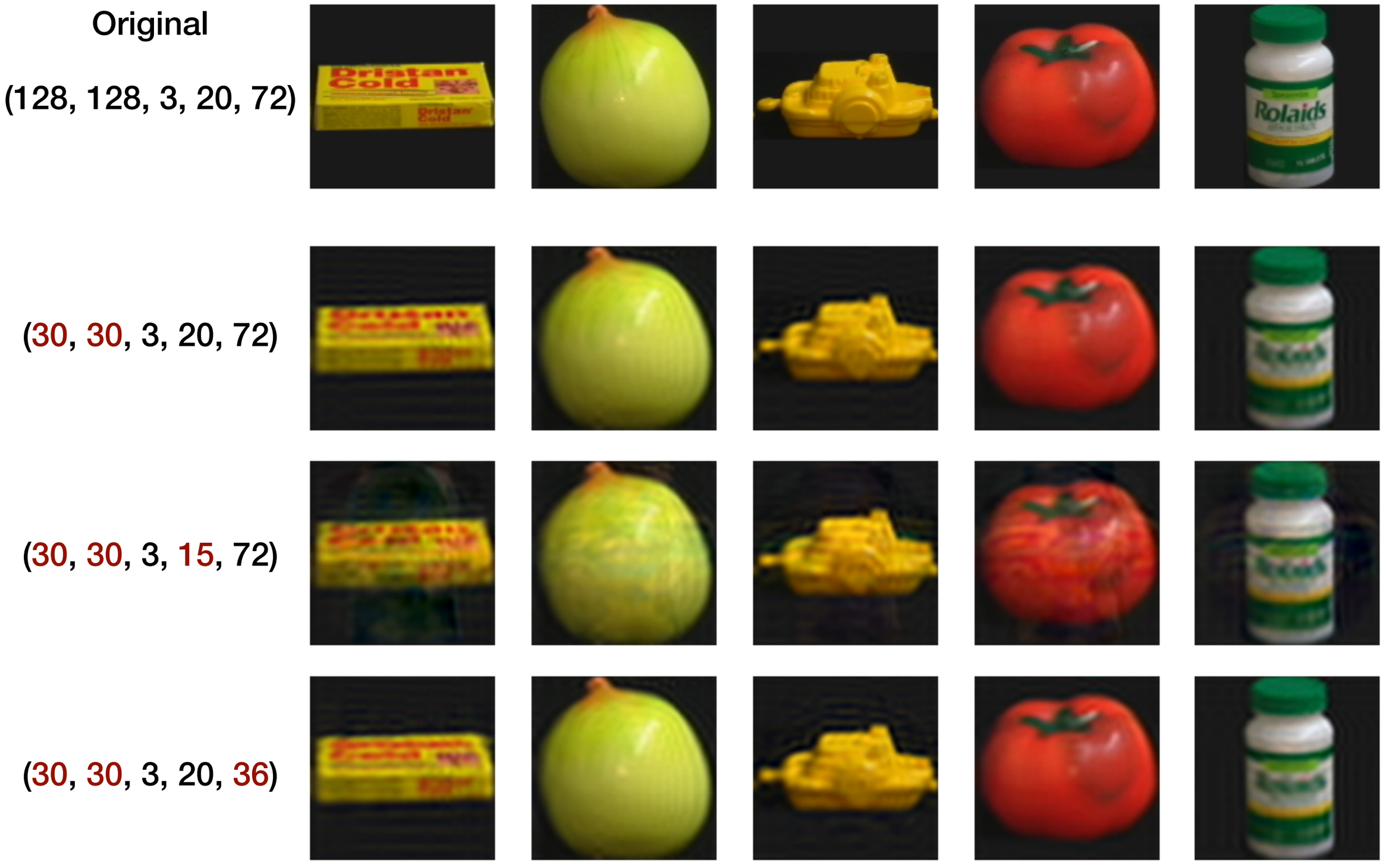
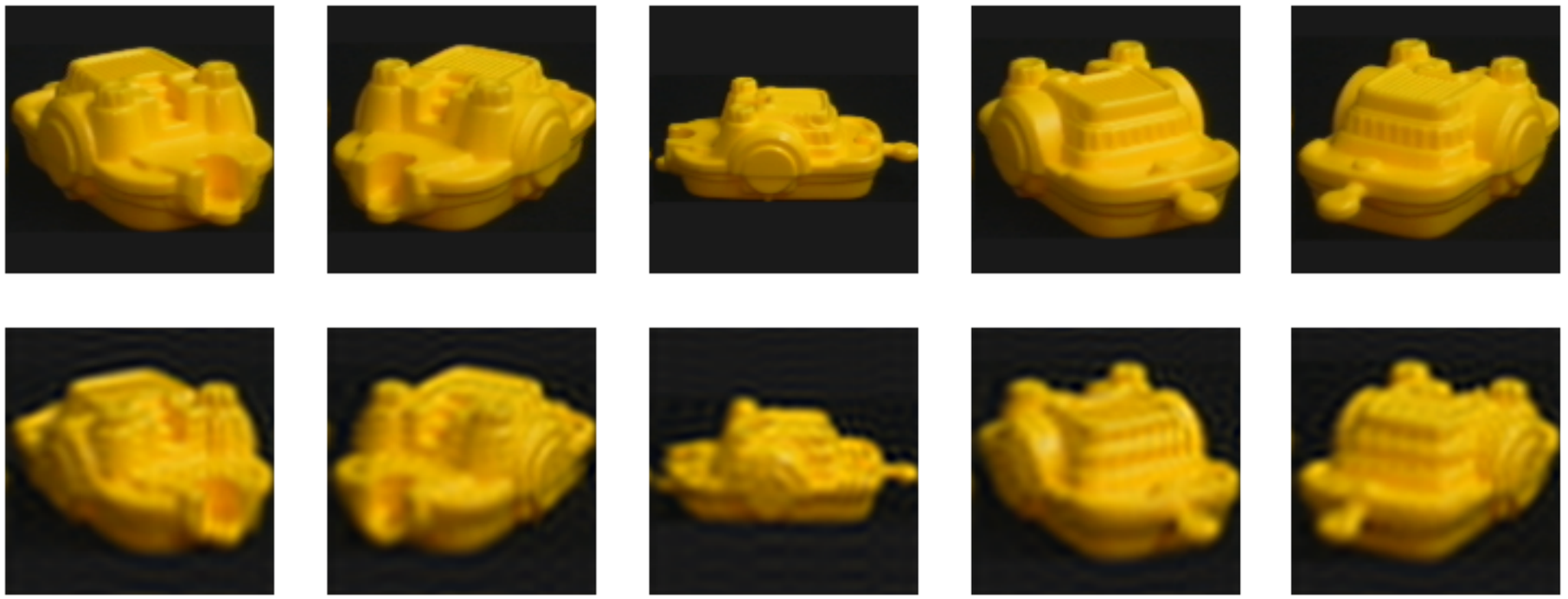


Image compression: multi images (color)

Original

(128, 128, 3, 20, 72)



(30, 30, 3, 20, 72)

(30, 30, 3, 15, 72)

(30, 30, 3, 20, 36)

Sample code: Low-rank approximation for pictures

You can try low rank approximations of images through sample codes in **"image_svd.zip".**

In the zip file you find four directories:

- image_gray
- image_color
- multi_image
- coil100_image

They correspond to the examples I showed in this lecture.

Each directory contains jupyter notebook (.ipynb) and python (.py) files.
(In addition, there are sample images.)

You can run them with **PIL, numpy, matplotlib**, modules.

I recommend you to use google colaboratory,
https://colab.research.google.com
where you can run .ipynb from your web browser.

(When you use google colab, don't forget to upload image files.)

References:

- 齋藤正彦、「線形代数入門」東京大学出版会
- 太田快人、「システム制御のための数学（１）ー線形代数編ー」、コロナ社
- T. G. Kolda et al, “Tensor Decompositions and Applications,” SIAM Review **51**, 455 (2006).

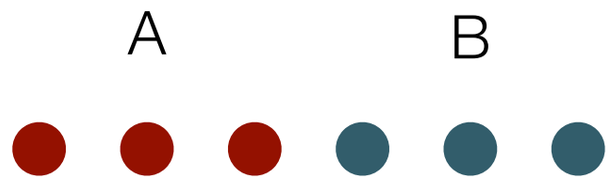
You can find references for other tensor decompositions applications in various fields!

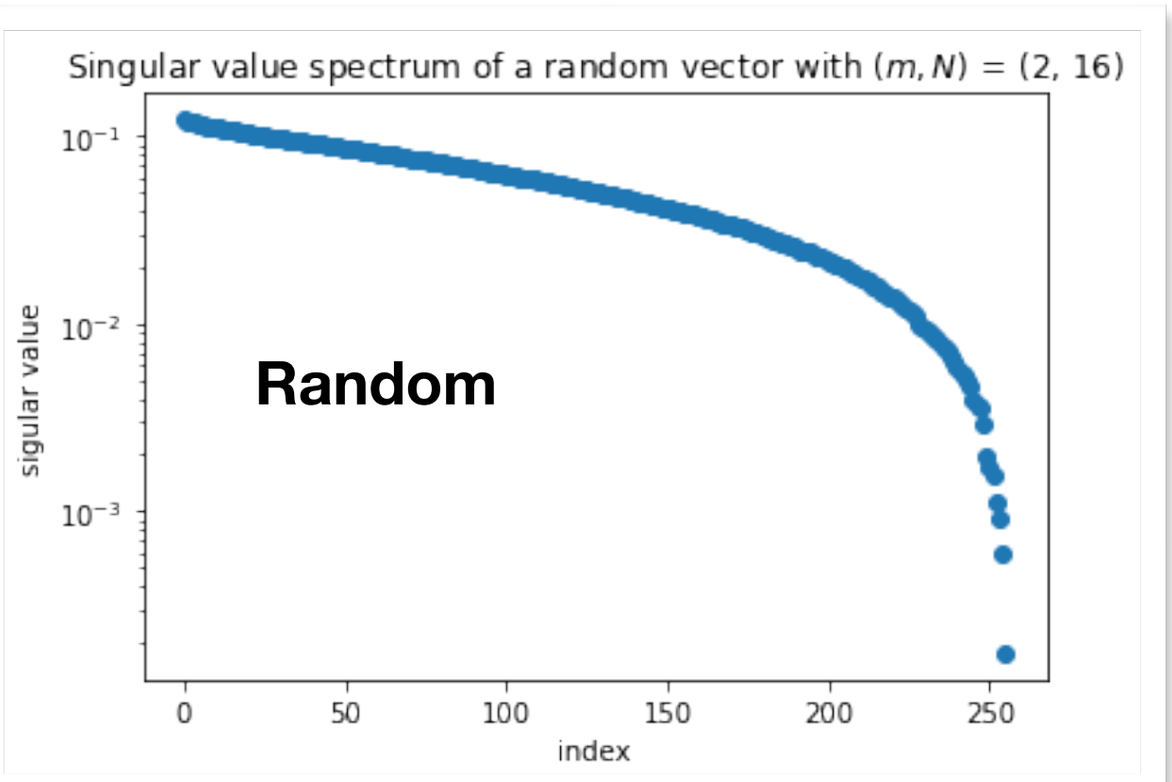
Toward another “low-rank” approximation

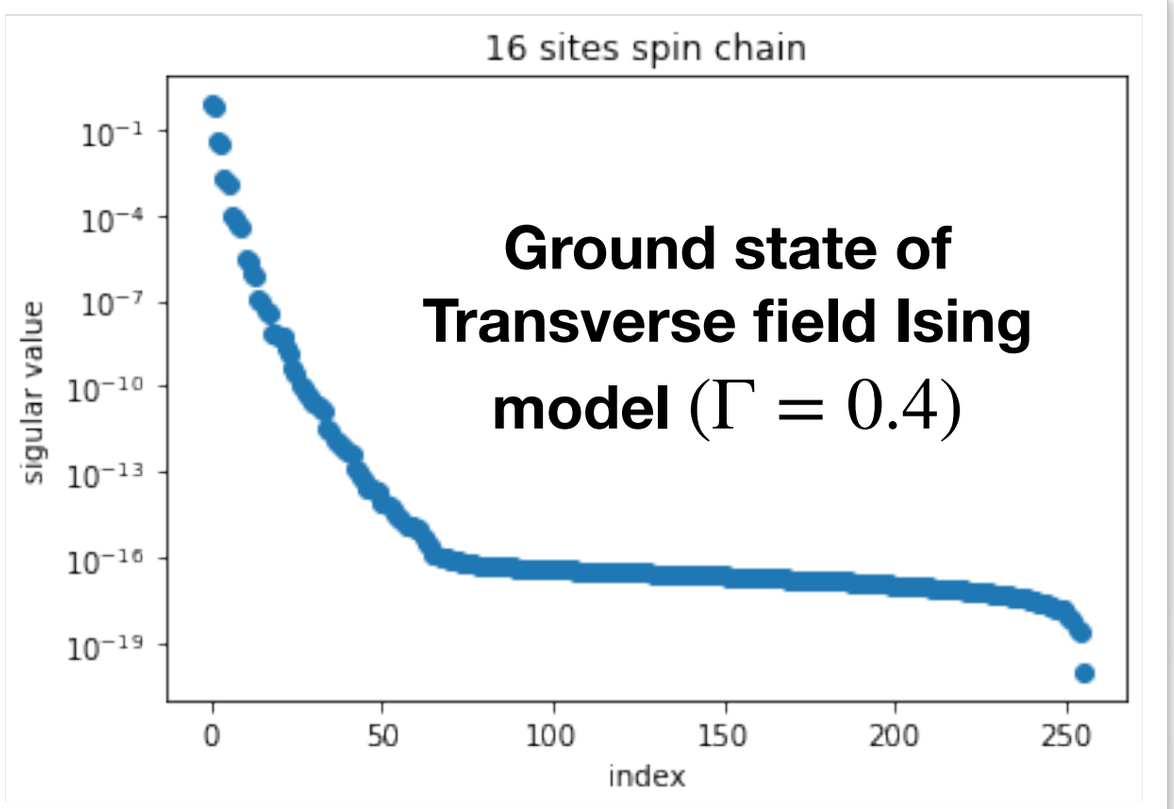
Singular values for N=16 qubits

$$\vec{v} \in \mathbb{C}^{2^{16}}$$

$$|\Psi\rangle = \sum_{i,j} M_{i,j} |A_i\rangle \otimes |B_j\rangle = \sum_i \lambda_i |\alpha_i\rangle \otimes |\beta_i\rangle$$





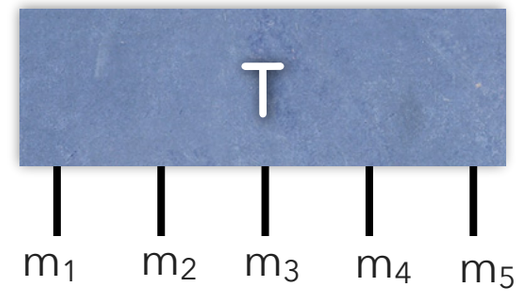


The ground state shows rapid decay of singular values.

➡ Can we perform data-compression for quantum state?

Another "generalization" of SVD to tensors.

$T_{m_1,m_2,\cdots,m_N}$:N-leg tensor (or Vector)

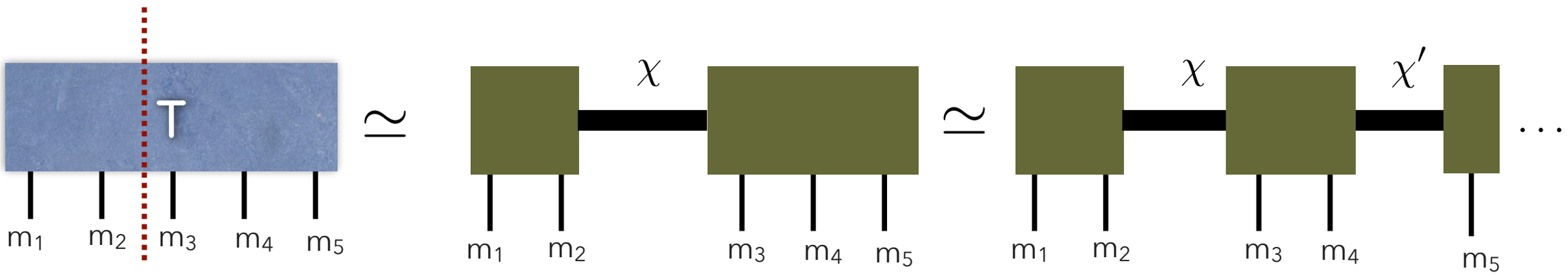


Cf. wave function: $|\Psi\rangle = \sum_{\{m_i=0,1\}} T_{m_1,m_2,\cdots,m_N} |m_1, m_2, \cdots, m_N\rangle$

We can consider it as a matrix by making two groups:

$T_{\{m_1,m_2,\cdots,m_M\},\{m_{M+1},\cdots,m_N\}}$

➡ We can perform the low rank approximation of T.



*obtained two objects are again tensors.

What does it mean? ➡ Tensor network decomposition!

Next week (Nov. 10)

Notice

- **No classes on Nov. 3,** Nov. 17, and Nov. 22
- Classes will be also held on Jan. 5 and Jan. 19

---

1. **Computational science, quantum computing, and data compression**
2. **Review of linear algebra**
3. **Singular value decomposition**
4. **Application of SVD and generalization to tensors**
5. **Entanglement of information and matrix product states**
6. **Application of MPS to eigenvalue problems**
7. **Tensor network representation**
8. **Data compression in tensor network**
9. **Tensor network renormalization**
10. **Quantum mechanics and quantum computation**
11. **Simulation of quantum computers**
12. **Quantum-classical hybrid algorithms and tensor network**
13. **Quantum error correction and tensor network**