Image as MPS

For the group meeting of 24/10

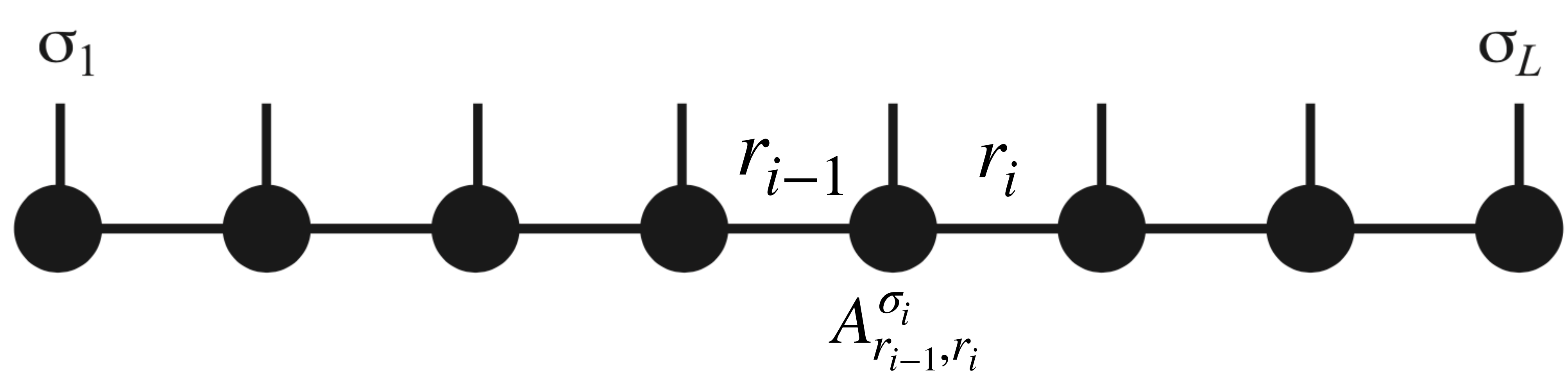
MPS : reminders

Theorem : Any rank L tensor ψ with leg dimensions d can be written exactly as a product of matrices.

$$\psi_{\sigma_1,\ldots,\sigma_L} = A^{\sigma_1}\cdots A^{\sigma_L}$$

- $\sigma_i \in \{1,\ldots,d\}$ are called physical indices
- $A^{\sigma_i} \in \mathbb{R}^{r_{i-1},r_i}$

Diagrammatically



The algorithm : from data to MPS

Consists essentially in doing SVD, reshaping, slicing. Here are the steps.

- $\psi_{\sigma_1,\ldots,\sigma_L} \to \psi_{\sigma_1,(\sigma_2\cdots\sigma_L)}$

$d^L \times 1$ $\quad$ $d \times d^{L-1}$

- $\psi = USV^\dagger$

$d \times r_1$ $\quad$ $r_1 \times r_1$ $\quad$ $r_1 \times d^{L-1}$

$r_i d \times r_{i+1}$ $\quad$ $r_{i+1} \times r_{i+1}$ $\quad$ $r_{i+1} \times d^{L-1-i}$

$r_1 = \min(d, d^{L-1})$

$r_{i+1} = \min(r_i d, d^{L-1-i})$

- Slice U into d matrices A^{σ_1} of size $1 \times r_1$

Slice U into d matrices $A^{\sigma_{i+1}}$ of size $r_i \times r_{i+1}$

- Repeat for site $i+1$

- Reshape $SV^\dagger \to \psi$ into $r_1 d \times d^{L-2}$

Reshape $SV^\dagger \to \psi$ into $r_{i+1} d \times d^{L-1-i-1}$

- At site $L-1 : \psi = USV^\dagger$

$r_{L-2} d \times r_{L-1}$ $\quad$ $r_{L-1} \times r_{L-1}$ $\quad$ $r_{L-1} \times d$

$SV^\dagger \to d$ matrices $\mathbb{R}^{r_{L-1},1}$

◆ Done by MPS$(d, L, \psi, k, \epsilon)$

Truncation : definitions

Let $\psi \in \mathbb{R}^{n\times m}$, with SVD decomposition : $\psi = USV^\dagger$

- $S = \begin{bmatrix} s_1 & & \\ & \ddots & \\ & & s_r \end{bmatrix}$,
- $r = \min(n, m)$
- $s_1 \geq s_2 \geq s_3 \geq \cdots \geq s_r$

Let $\epsilon > 0$ and $a_\epsilon = \#\{s_i \mid s_i > \epsilon\}$. Let $k \in \{2,3,...\}$. Then :

If $k \leq a_\epsilon$	If $k > a_\epsilon$
$U \to U_k \in \mathbb{R}^{n,k}$	$U \to U_\epsilon \in \mathbb{R}^{n,a_\epsilon}$
$S \to S_k \in \mathbb{R}^{k,k}$	$S \to S_\epsilon \in \mathbb{R}^{a_\epsilon,a_\epsilon}$
$V^\dagger \to V_k^\dagger \in \mathbb{R}^{k,m}$	$V^\dagger \to V_\epsilon^\dagger \in \mathbb{R}^{a_\epsilon,m}$

$$r \to \min(a_\epsilon, k) =: r(\epsilon, k)$$

$$[r_1, r_2, \ldots, r_{L-1}]$$

$$\Big\downarrow \text{ Truncation } k, \epsilon$$

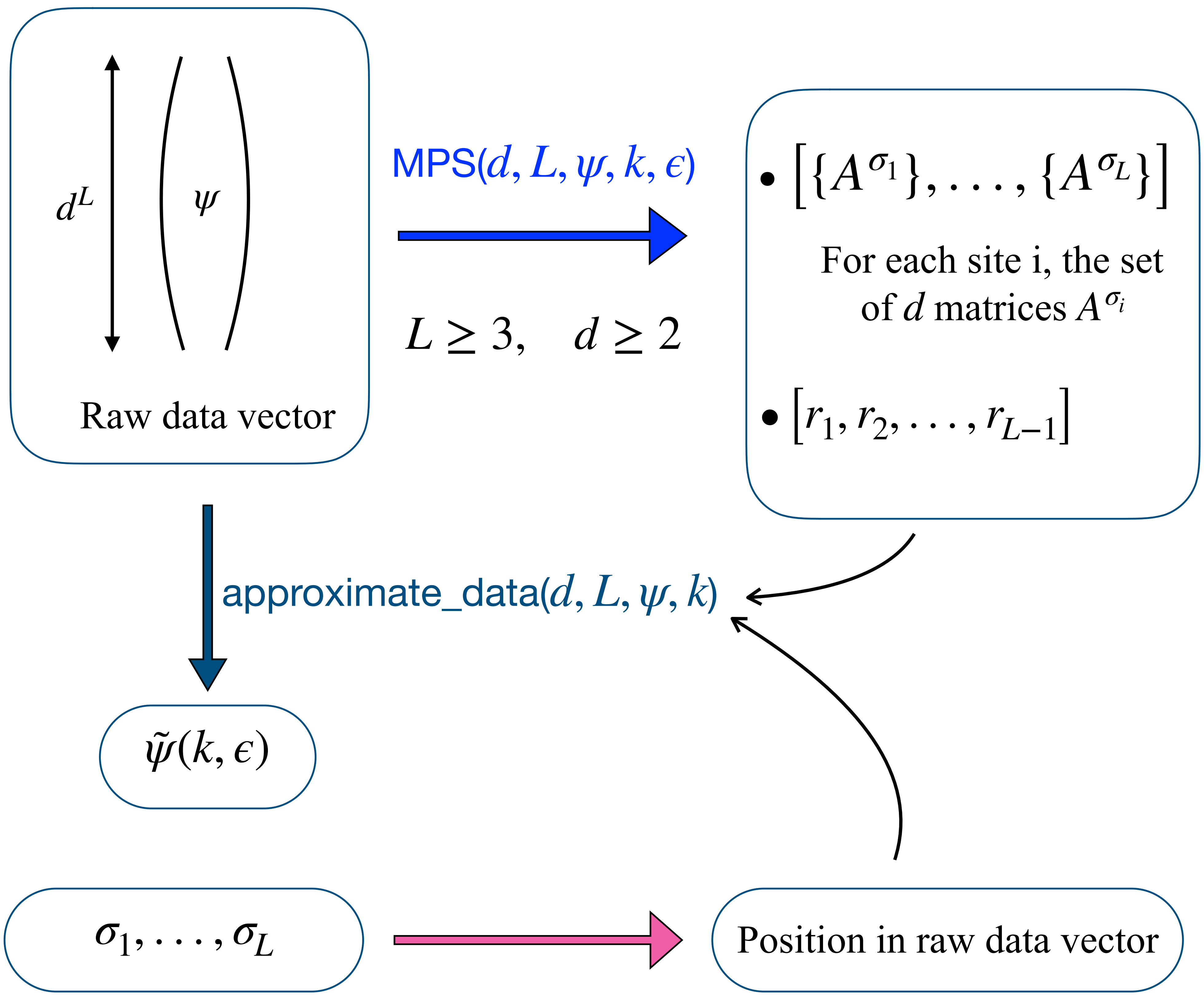
$$[r_1(\epsilon,k), r_2(\epsilon,k), \ldots, r_{L-1}(\epsilon,k)]$$

$k = 1$ not allowed since scalar cannot be reshaped

k is an « additional » truncation, same at each step

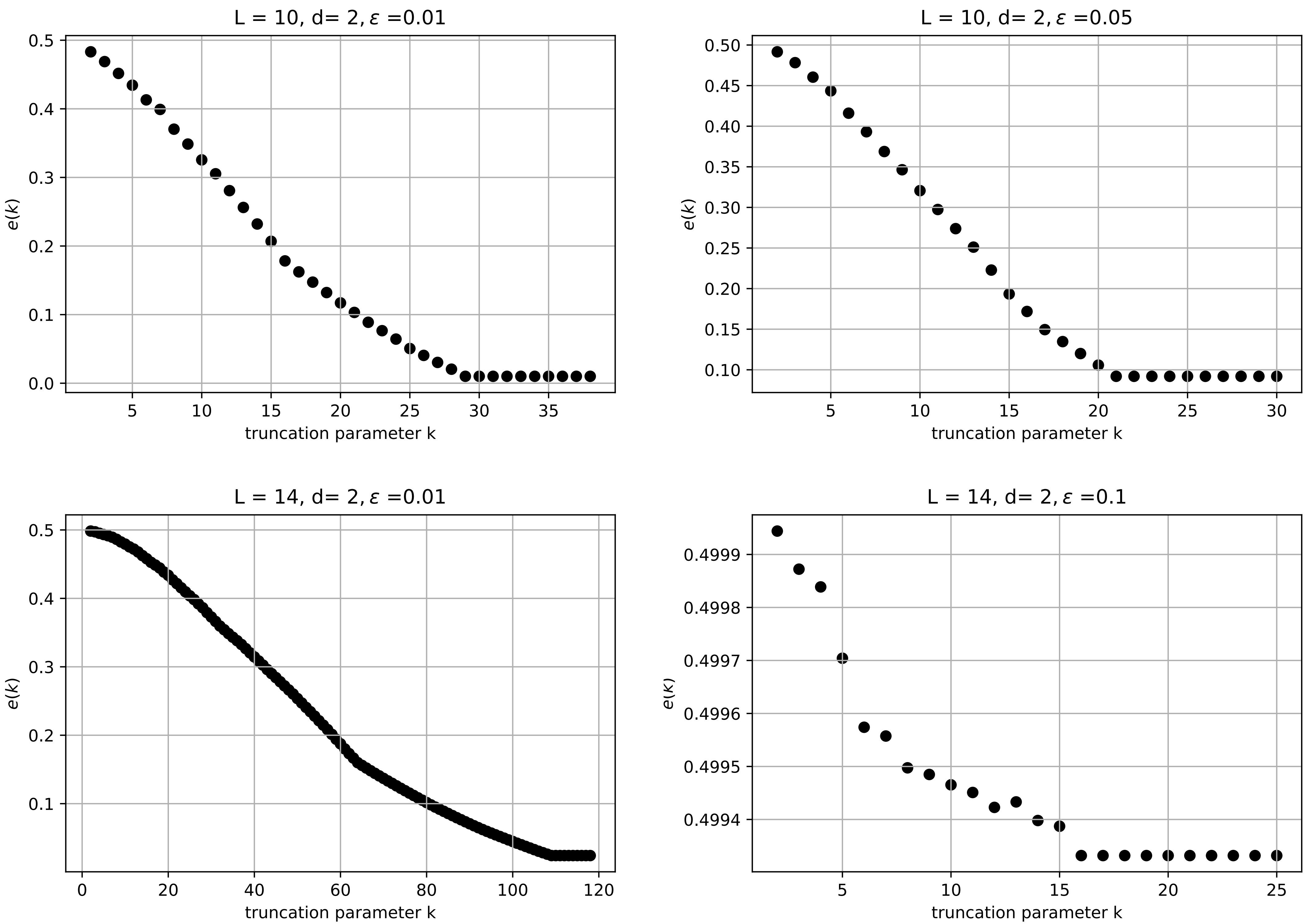
Error induced by truncation :

$$e_\psi(k,\epsilon) := \frac{|\psi - \tilde{\psi}(k,\epsilon)|}{|\psi|}$$


d^L
ψ
Raw data vector
MPS(d, L, ψ, k, ϵ)
$L \geq 3, \quad d \geq 2$
• $\left[\{A^{\sigma_1}\}, \ldots, \{A^{\sigma_L}\}\right]$
For each site i, the set of d matrices A^{σ_i}
• $\left[r_1, r_2, \ldots, r_{L-1}\right]$
approximate_data(d, L, ψ, k)
$\tilde{\psi}(k, \epsilon)$
$\sigma_1, \ldots, \sigma_L$
Position in raw data vector

Plots of the error : fixed ϵ

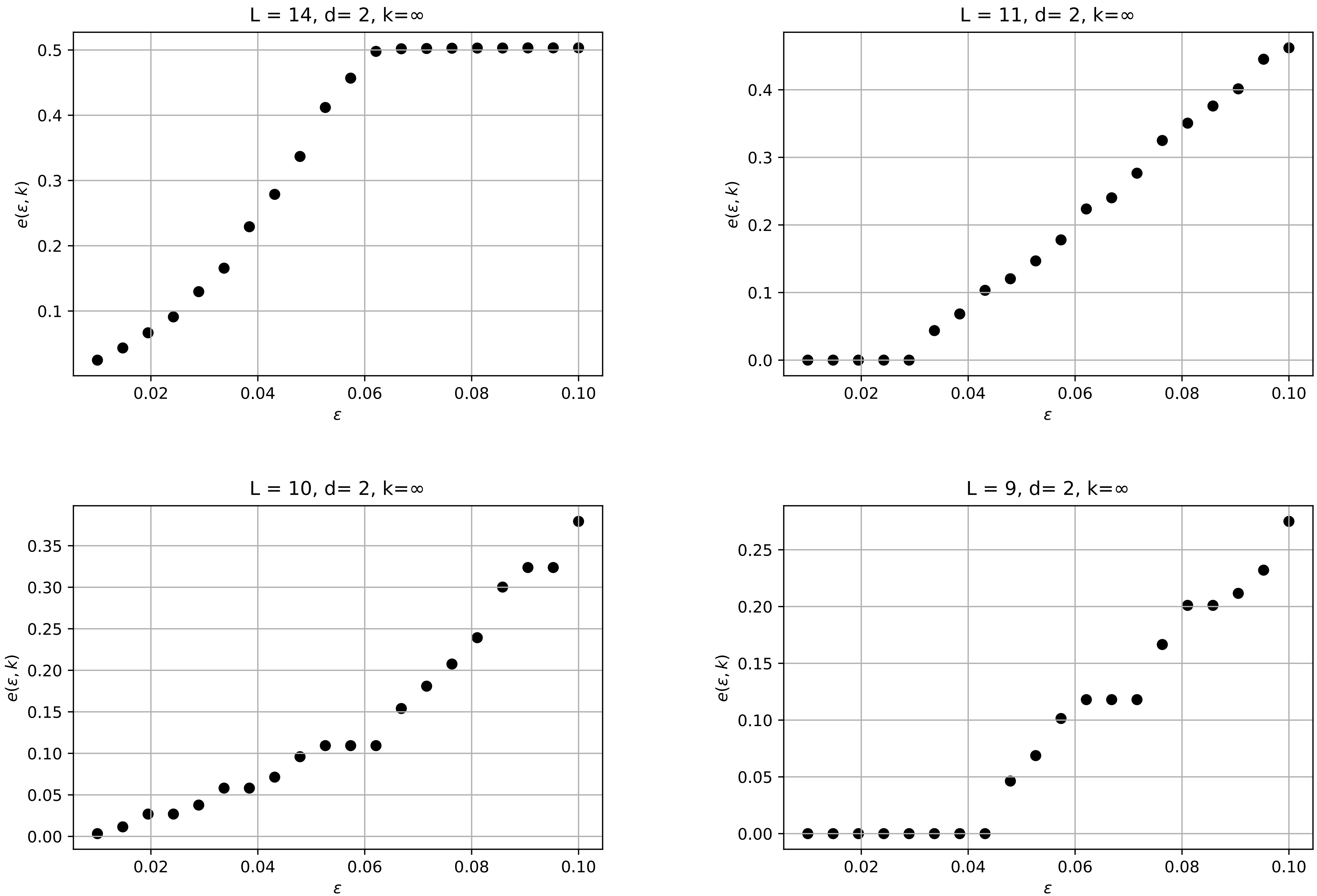
Input : normalized $\psi \in \mathbb{R}^{d^L \times 1}$ with entries $\sim \mathcal{U}(0,1)$

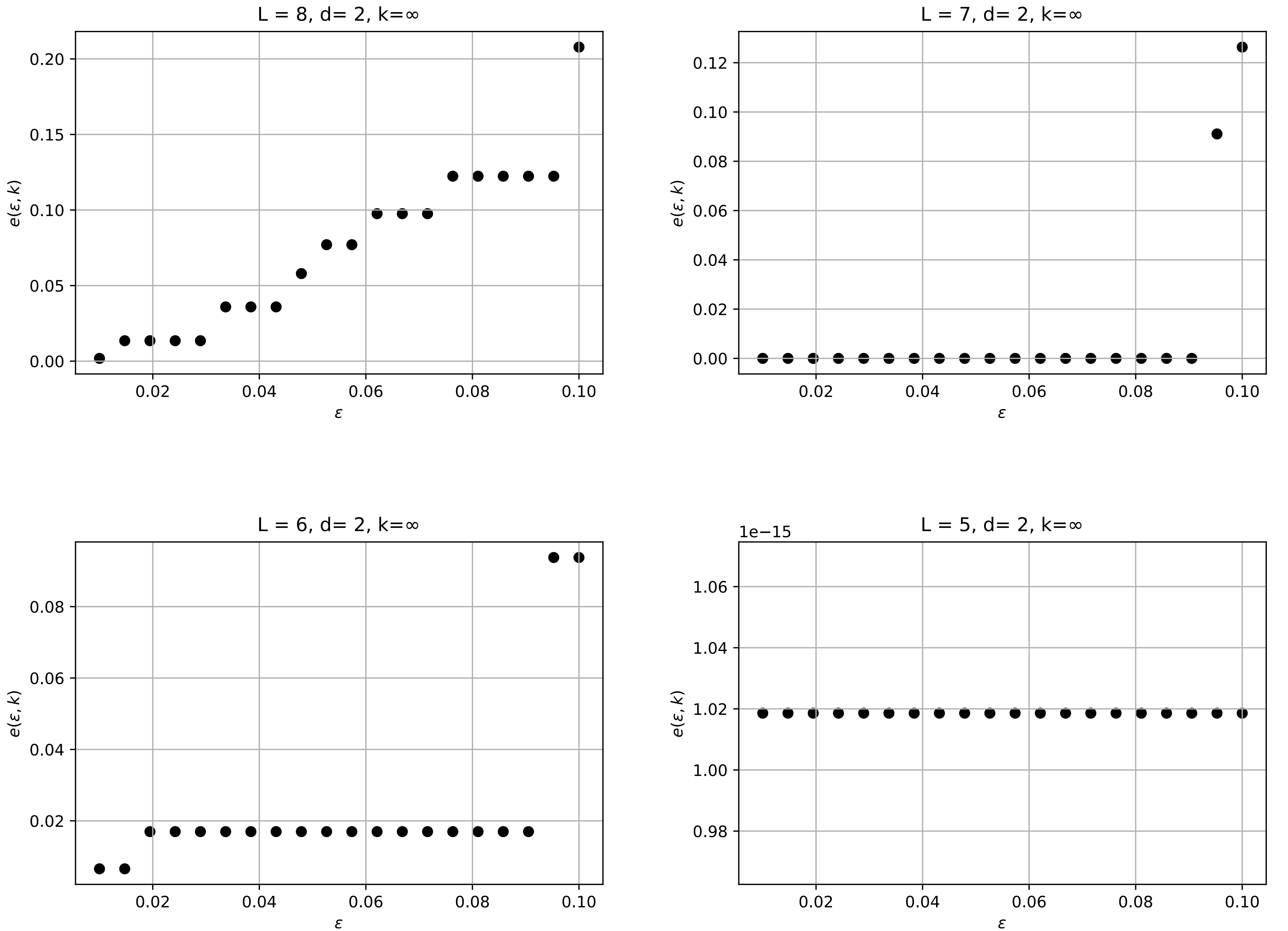


- Goes to zero
- When increase ϵ, jumps

Plots of the error : fixed k

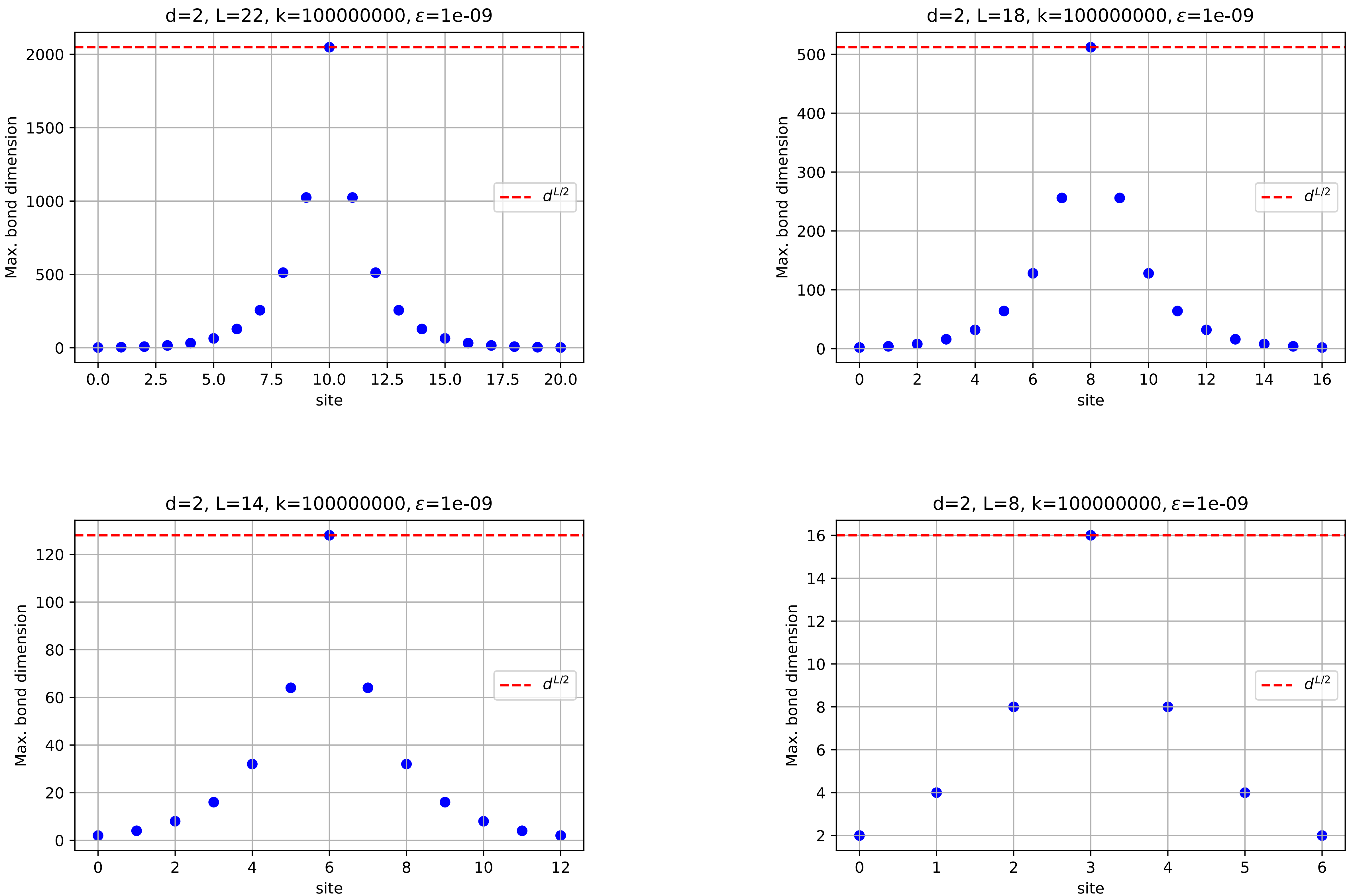
Input : normalized $\psi \in \mathbb{R}^{d^L \times 1}$ with entries $\sim \mathscr{U}(0,1)$



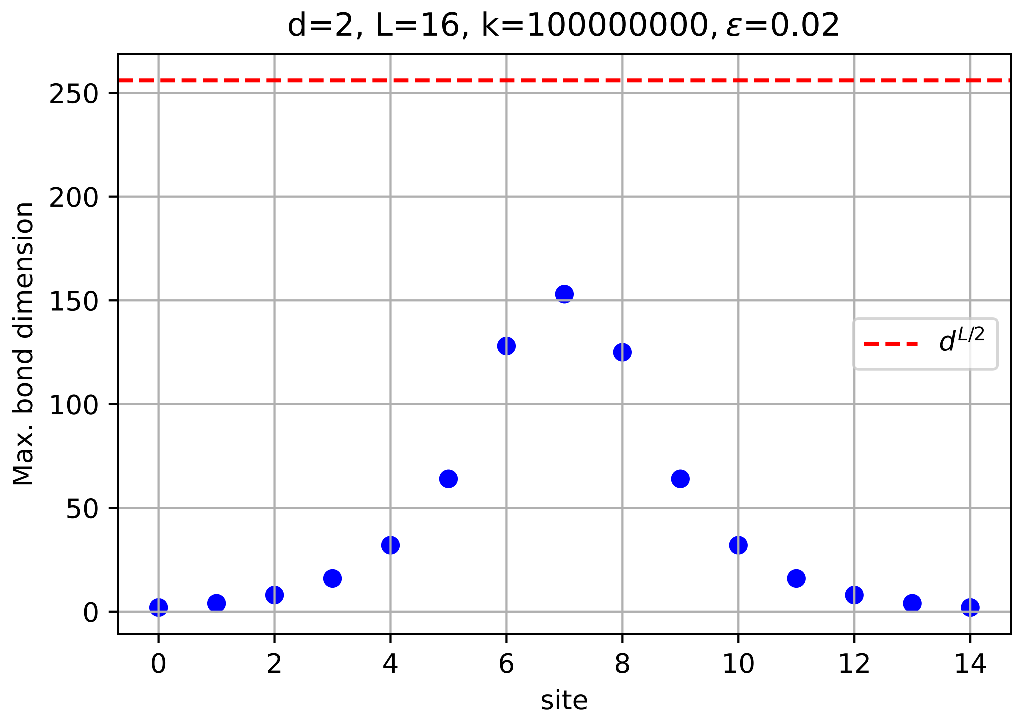

L = 8, d= 2, k=∞
L = 7, d= 2, k=∞
L = 6, d= 2, k=∞
L = 5, d= 2, k=∞
1e−15
e(ε, k)
ε

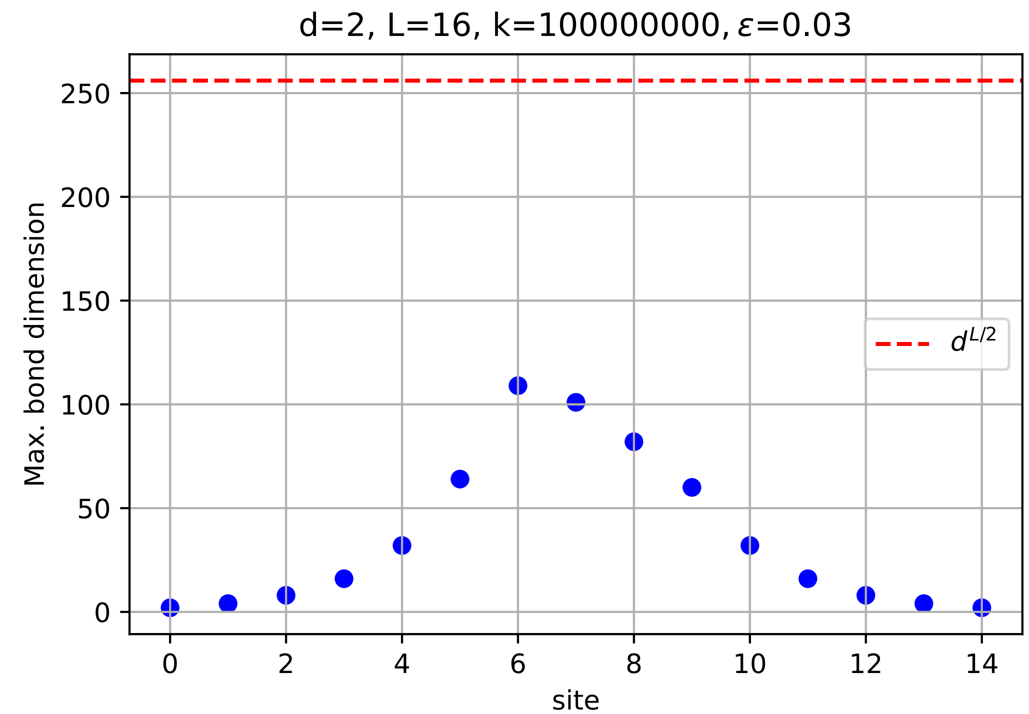
Bond dimensions

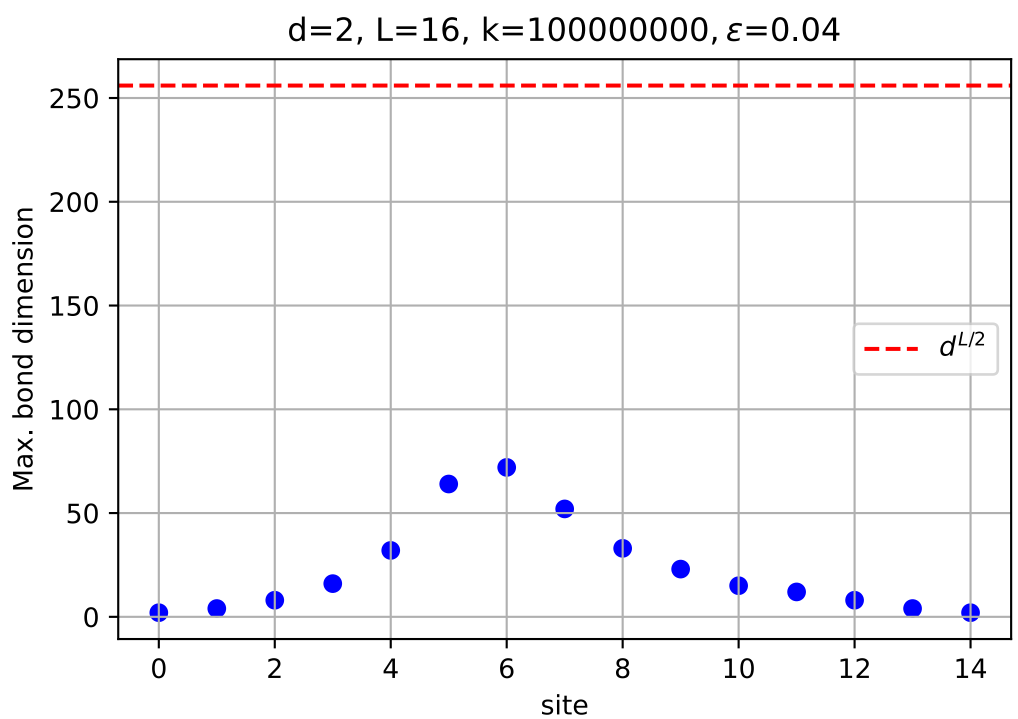
- Input : norm. $\psi \in \mathbb{R}^{d^L \times 1}$ with entries $\sim \mathcal{U}(0,1)$
- $k = \infty, \epsilon \to 0$: no truncation, max. bond dim.

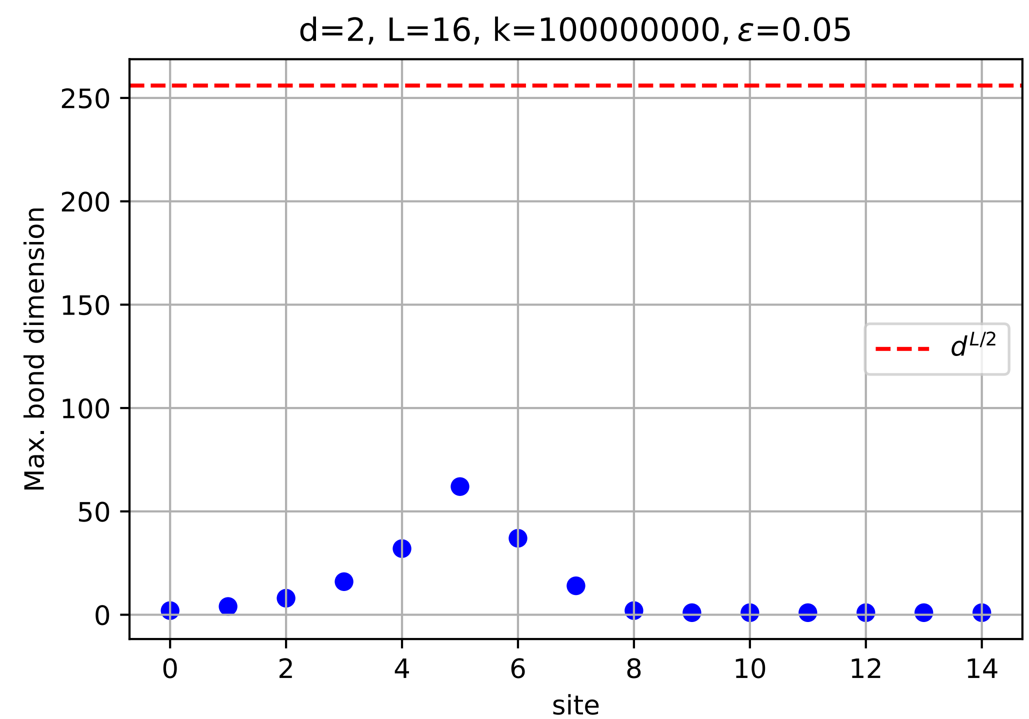


$k = \infty$ fixed





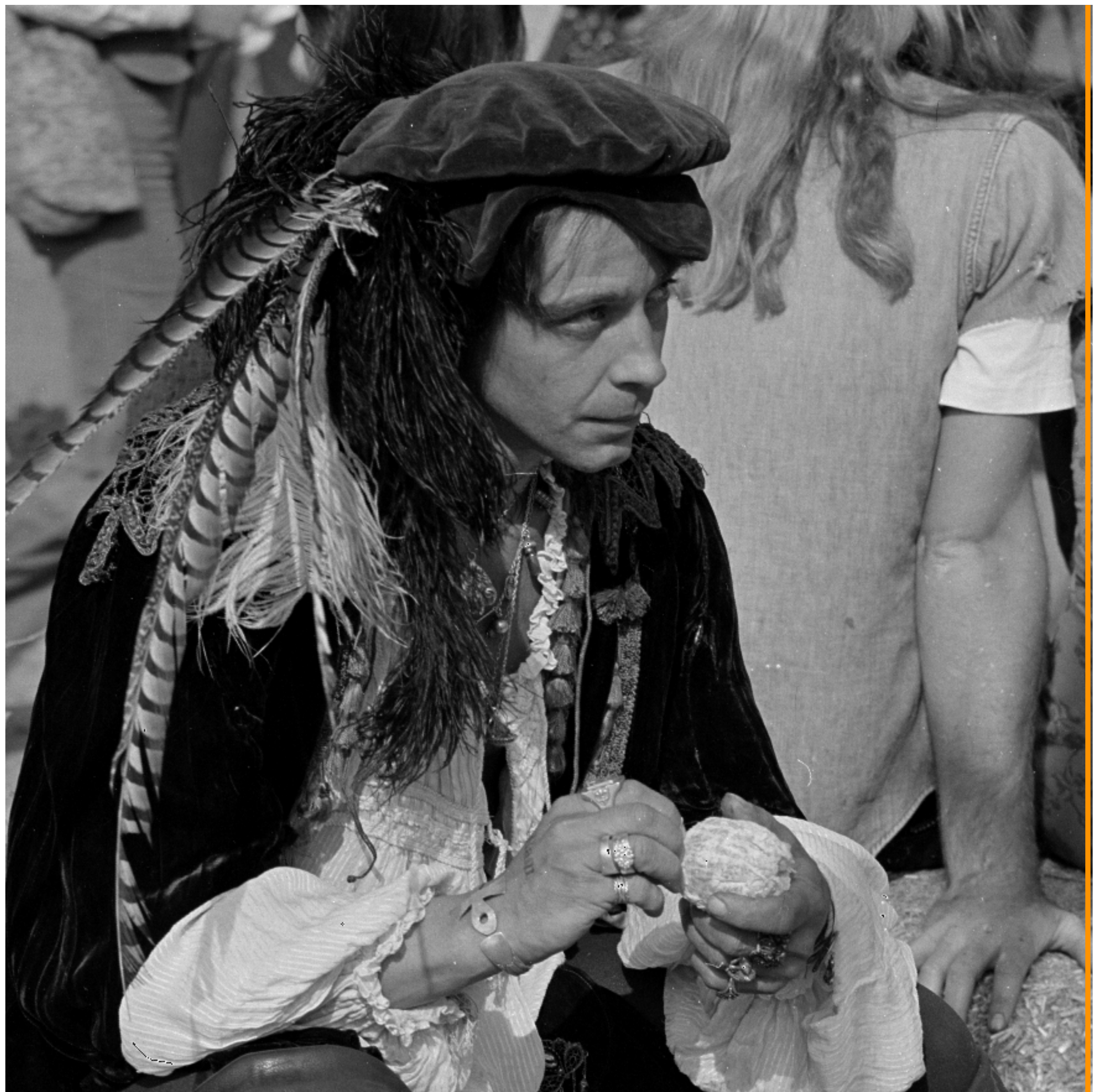




What about images ?

$d^R < n$

d^R



◆image_to_vector(image_name,d, R) does that

Python allows to convert tiff images to np.array

$= \quad M \in \mathbb{R}^{n,n}$

n is the number of pixels

Reshaping,
Row-major ordering
Normalization

$$d^{2R} \quad \left(\psi \right)$$

No loss of information when Reshaping.

Raw data vector

Compressing the image

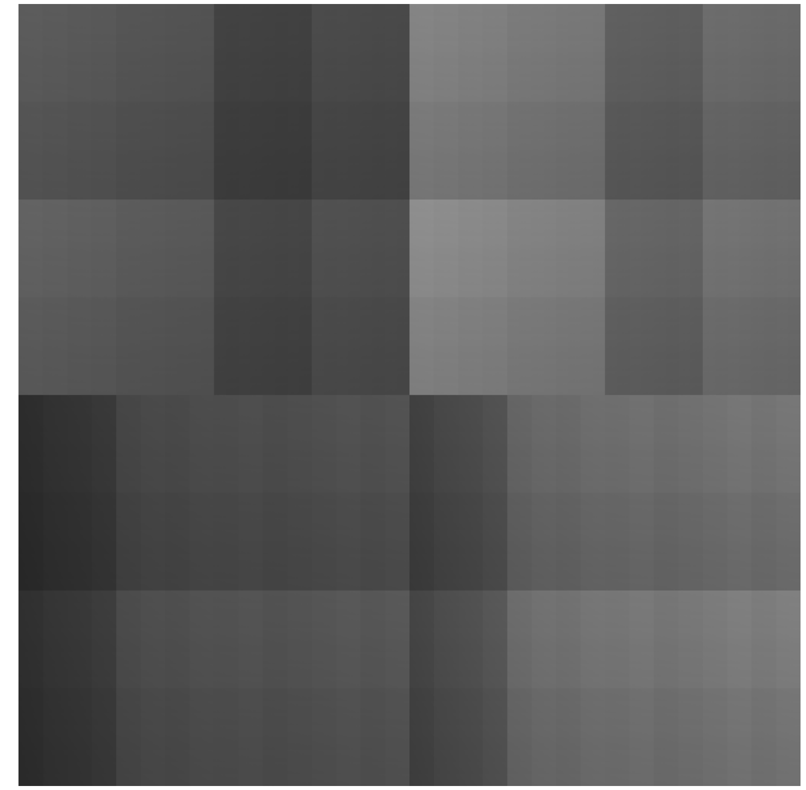
Now that we have the raw data vector from the image, we can use our MPS() function.

Original

$k = 10^6, \epsilon = 0.01$

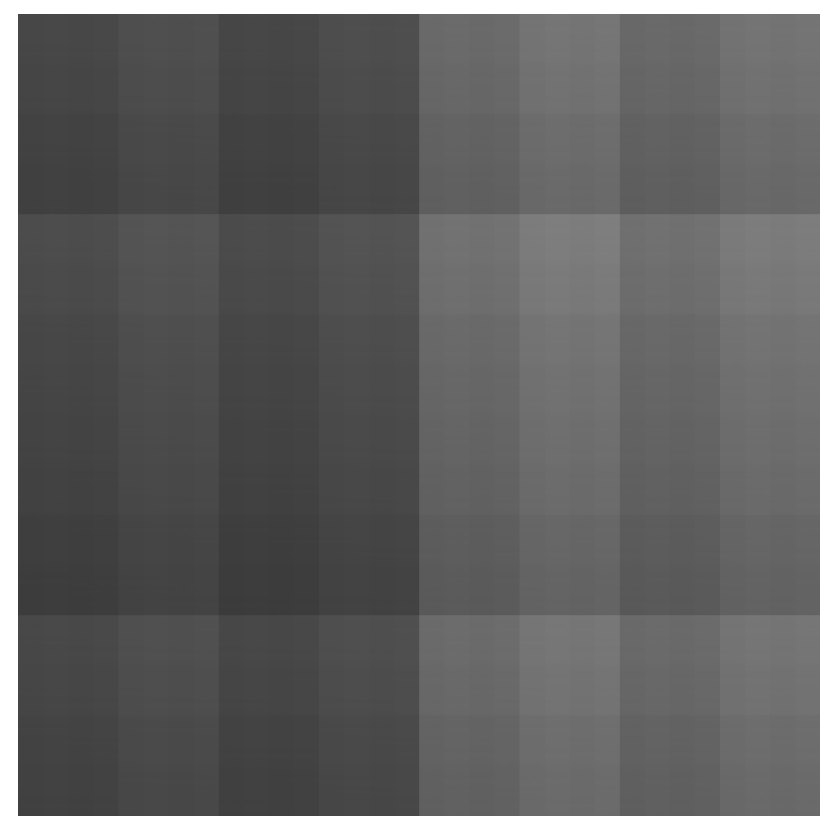
Original

$k = 10^6, \epsilon = 0.2$



$k = 100, \epsilon = 0.001$

$k = 10^6, \epsilon = 0.5$



- We have seen a first way of writing an image as MPS.

- There is **another way**, that consists in setting the physical indices to be the value of the bit representation of the coordinates of each point in the image

- **Advantage** : two consecutive indices are highly entangled, allowing more compression with no loss of information [1]

[1] Shinaoka, H., Wallerberger, M., Murakami, Y., Nogaki, K., Sakurai, R., Werner, P., & Kauch, A. (2022). Multi-scale space-time ansatz for correlation functions of quantum systems. *arXiv preprint arXiv:2210.12984*.

Consider a $n \times n$ image, $n := 2^R$

$\forall (i,j) \in \{0,\ldots,n-1\}^2$, then $\dfrac{i}{n}, \dfrac{j}{n} \in [0,1[$ so have bit representation !

$$\frac{i}{N} = \frac{x_1}{2} + \frac{x_2}{2^2} + \frac{x_3}{2^3} + \cdots + \frac{x_R}{2^R}$$

$$x_i, y_i \in \{0,1\}$$

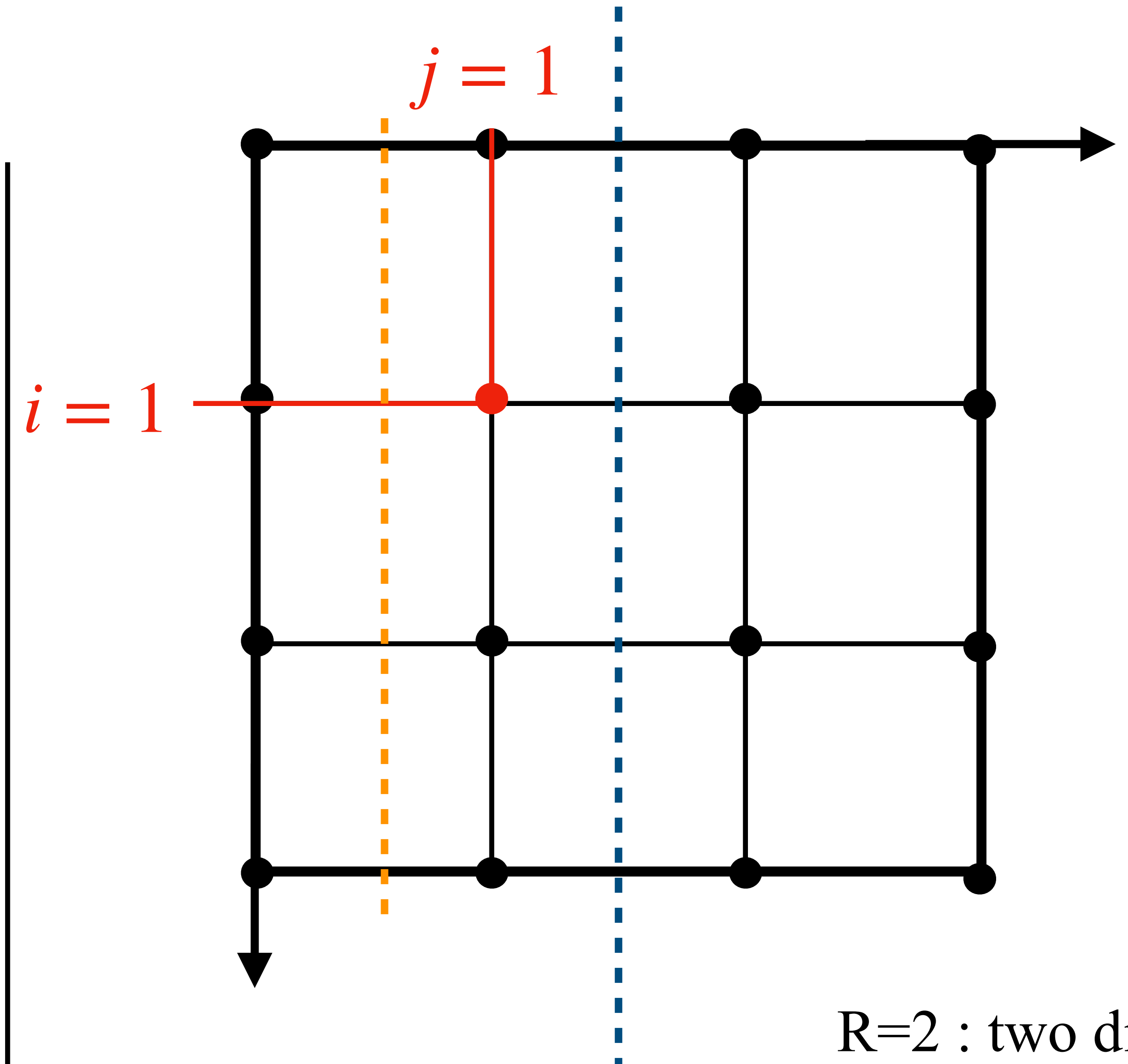
$$\frac{j}{N} = \frac{y_1}{2} + \frac{y_2}{2^2} + \frac{y_3}{2^3} + \cdots + \frac{y_R}{2^R}$$

$$i \leftrightarrow N(x_1, \ldots, x_R), \quad j \leftrightarrow N(y_1, \ldots, y_R)$$

$$\sigma_1, \sigma_2 = x_1, y_1, \cdots, \quad \sigma_{2R-1}, \sigma_{2R} = x_R, y_R$$

Let f be the color of pixel

$$f(x_1, y_1, x_2, y_2, \ldots, x_R, y_R) = \psi_{\sigma_1, \sigma_2, \ldots, \sigma_{2R-1}, \sigma_{2R}}$$



R=2 : two division

$i \backslash j$	0	1	2	3
0	0000	0001	0100	0101
1	0010	**0011**	0110	0111
2	1000	1001	1100	1101
3	1010	1011	1110	1111

$$\frac{0}{2^1} + \frac{1}{2^2}$$

$y_1 = 0, y_2 = 1$

$x_1 = 0, x_2 = 1$

Comparing two ways of encoding image : bond dimensions