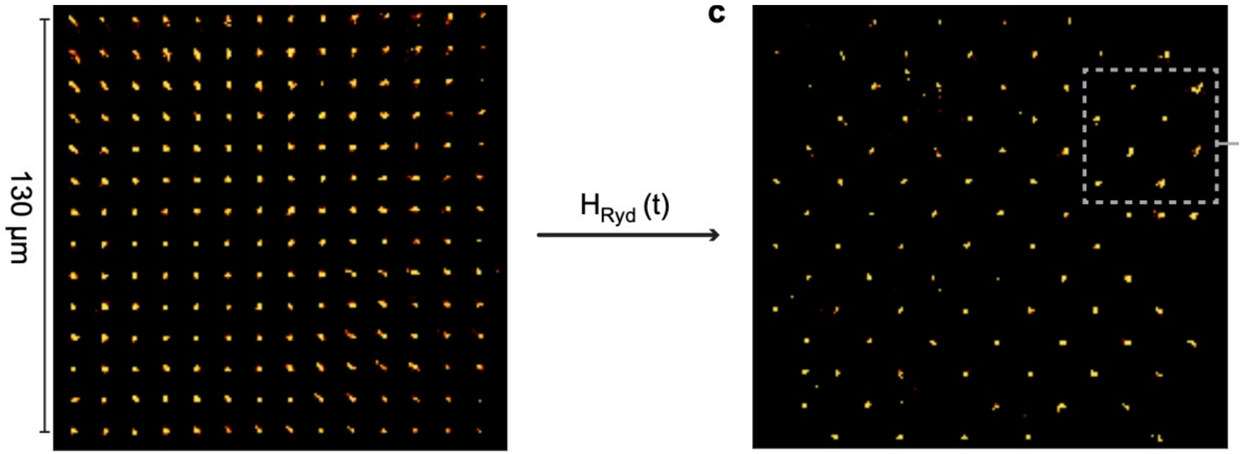
PASQAL

Kemal Bidzhiev & Anton Quelle

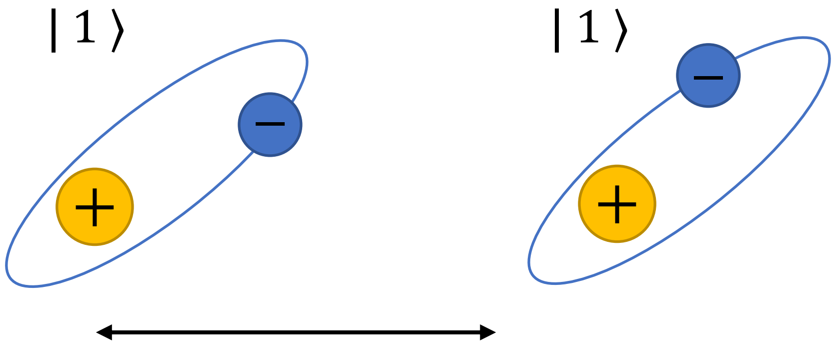
HPC and Quantum Computing CINECA 2022

Cloud on-demand quantum dynamics with Tensor Networks

Rydberg atoms

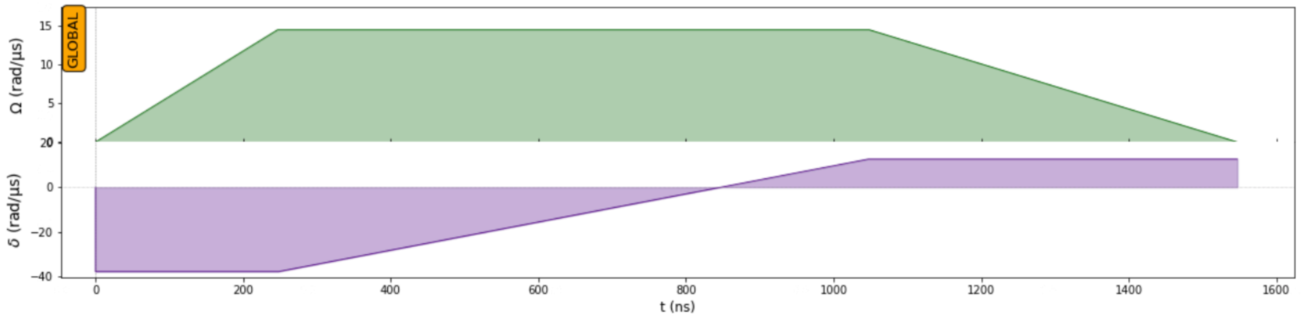


Programmable Rydberg atoms. *Nature* **595,** 233–238 (2021)



Dipole – dipole interaction

$$H_{Ryd}(t) = \sum_{i,j} \frac{C_6}{r_{ij}^6} n_i n_j + \frac{\Omega(t)}{2} \sigma_i^x - \delta(t) n_i$$



Cloud access for customers

Pulser : An open-source package for the design of pulse sequences in programmable neutral-atom arrays. *Quantum* **6**, 629 (2022)

Quantum complexity

$$|\psi\rangle_1 = \sum_i A_{i_1}|i_1\rangle = \alpha_0|0\rangle + \alpha_1|1\rangle =$$

$$\alpha_0 \begin{bmatrix} 1 \\ 0 \end{bmatrix} + \alpha_1 \begin{bmatrix} 0 \\ 1 \end{bmatrix} = \begin{bmatrix} \alpha_0 \\ \alpha_1 \end{bmatrix}$$

Qbits	Type	Parameters
1	A_i	2

Quantum complexity

$$|\psi\rangle_1 = \sum_i A_{i_1}|i_1\rangle = \alpha_0|0\rangle + \alpha_1|1\rangle$$

$$|\psi\rangle_2 = \sum_i A_{i_1 i_2}|i_1\, i_2\rangle = \alpha_{00}|00\rangle + \alpha_{01}|01\rangle + \alpha_{10}|10\rangle + \alpha_{11}|11\rangle$$

$$\alpha_{00}\begin{pmatrix}1\\0\\0\\0\end{pmatrix} + \alpha_{01}\begin{pmatrix}0\\1\\0\\0\end{pmatrix} + \alpha_{10}\begin{pmatrix}0\\0\\1\\0\end{pmatrix} + \alpha_{11}\begin{pmatrix}0\\0\\0\\1\end{pmatrix} = \begin{pmatrix}\alpha_{00}\\\alpha_{01}\\\alpha_{10}\\\alpha_{11}\end{pmatrix}$$

$$\begin{array}{c|cc} |i_2\rangle \uparrow & \alpha_{01} & \alpha_{11} \\ & \alpha_{00} & \alpha_{10} \\ \hline & \multicolumn{2}{c}{|i_1\rangle \rightarrow} \end{array} \qquad A_{i_1 i_2}$$

Qbits	Type	Parameters
1	A_i	2
2	A_{ij}	4

The same information can be represented in a vector or a matrix forms

Quantum complexity

$$|\psi\rangle_1 = \sum_i A_{i_1} |i_1\rangle = \alpha_0 |0\rangle + \alpha_1 |1\rangle$$

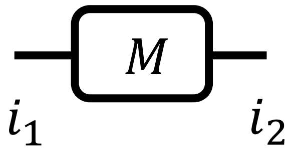
$$|\psi\rangle_2 = \sum_i A_{i_1 i_2} |i_1\, i_2\rangle = \alpha_{00} |00\rangle + \alpha_{01} |01\rangle + \alpha_{10} |10\rangle + \alpha_{11} |11\rangle$$

Qbits	Type	Parameters
1	A_i	2
2	A_{ij}	4

$A_{i_1 i_2}$

$$\begin{array}{c|cc} & & \\ |i_1\rangle \uparrow & \alpha_{01} & \alpha_{11} \\ & \alpha_{00} & \alpha_{10} \\ \hline & \multicolumn{2}{c}{|i_2\rangle \rightarrow} \end{array}$$

Matrix A



The same information can be represented in a vector or a matrix forms

PASQAL

Quantum complexity

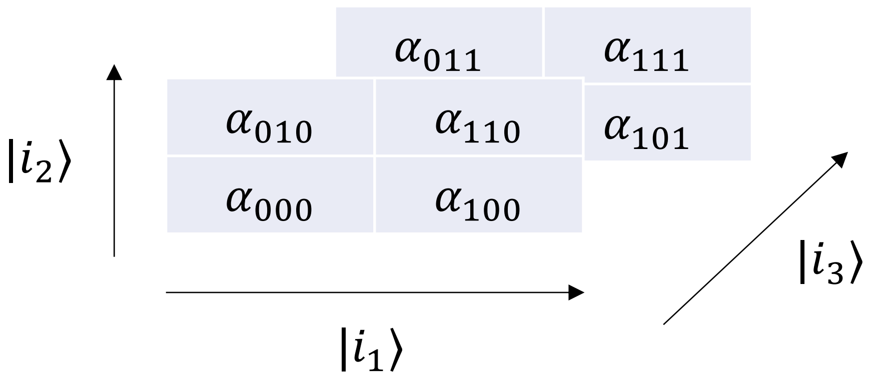
$$|\psi\rangle_1 = \sum_i A_{i_1}|i_1\rangle = \alpha_0|0\rangle + \alpha_1|1\rangle$$

$$|\psi\rangle_2 = \sum_i A_{i_1 i_2}|i_1\, i_2\rangle = \alpha_{00}|00\rangle + \alpha_{01}|01\rangle + \alpha_{10}|10\rangle + \alpha_{11}|11\rangle$$

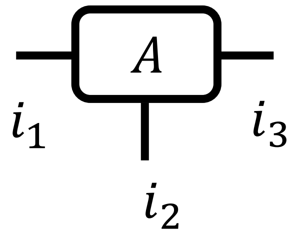
$$|\psi\rangle_N = \sum_i A_{i_1 \ldots i_N}|i_1 \ldots i_N\rangle$$

Qbits	Type	Parameters
1	A_i	2
2	A_{ij}	4
3	A_{ijk}	8
N	$A_{i_1 \ldots i_N}$	2^N

$A_{i_1 i_2 i_3}$



3d tensor



For 3 qubits the quantum state can be encoded as a 3d tensor $A_{i_1 i_2 i_3}$

Quantum complexity

$$|\psi\rangle_1 = \sum_i A_{i_1}|i_1\rangle = \alpha_0|0\rangle + \alpha_1|1\rangle$$

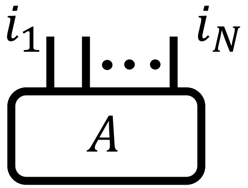
$$|\psi\rangle_2 = \sum_i A_{i_1 i_2}|i_1\, i_2\rangle = \alpha_{00}|00\rangle + \alpha_{01}|01\rangle + \alpha_{10}|10\rangle + \alpha_{11}|11\rangle$$

$$|\psi\rangle_N = \sum_i A_{i_1 \dots i_N}|i_1 \dots i_N\rangle$$

Qbits	Type	Parameters
1	A_i	2
2	A_{ij}	4
3	A_{ijk}	8
N	$A_{i_1 \dots i_N}$	2^N

$A_{i_1 \dots i_N}$ is a N dim array = ‘tensor’

Linear algebra starts to play an important role



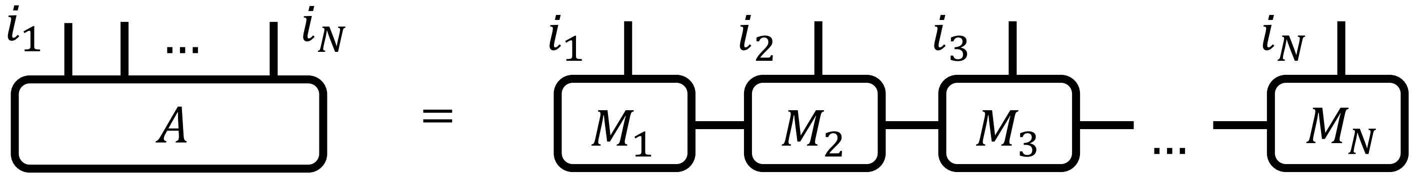
parameter for $N = 300$ is $\approx 10^{90}$
of particles in the observable universe *vigintillion* $\approx 10^{80}$

Array of coefficients for 40 qbits requires $\sim$1 Tb of memory.
How to store it in a computer ?

PASQAL

Tensor Networks

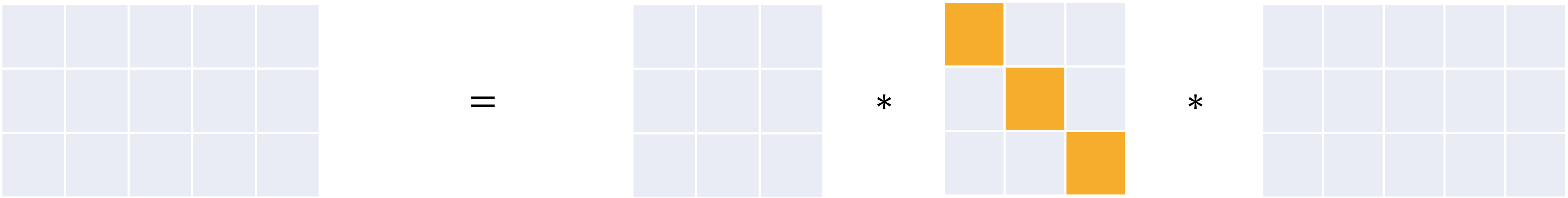
Matrix Product State/ Tensor Train $$|\psi\rangle_N = \sum_i A_{i_1 \dots i_N} |i_1 \dots i_N\rangle$$



We represent a high rank tensor as a product of low rank tensors.
One can ‘compress’ each M.

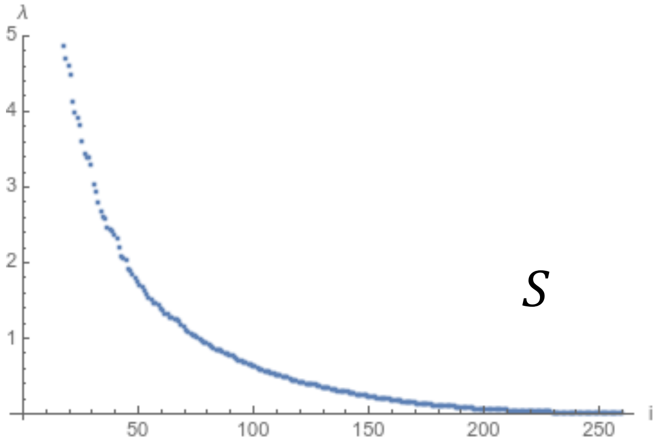
Tensor Networks: SVD compression

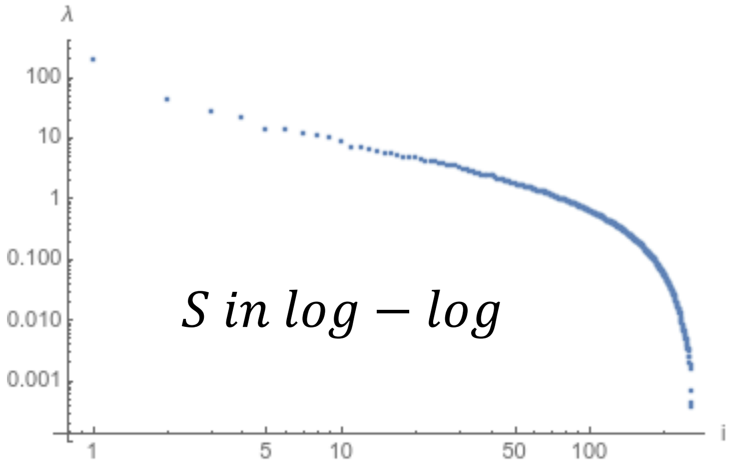
Multidimensional tensors can be compressed with using SVD – Singular Value Decomposition



$$M \quad = \quad U \quad * \quad S \quad * \quad V^{\dagger}$$

Pascal

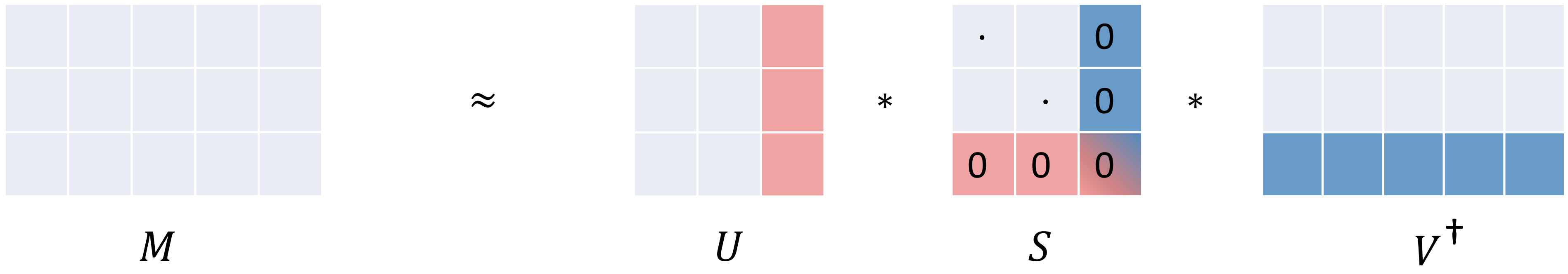


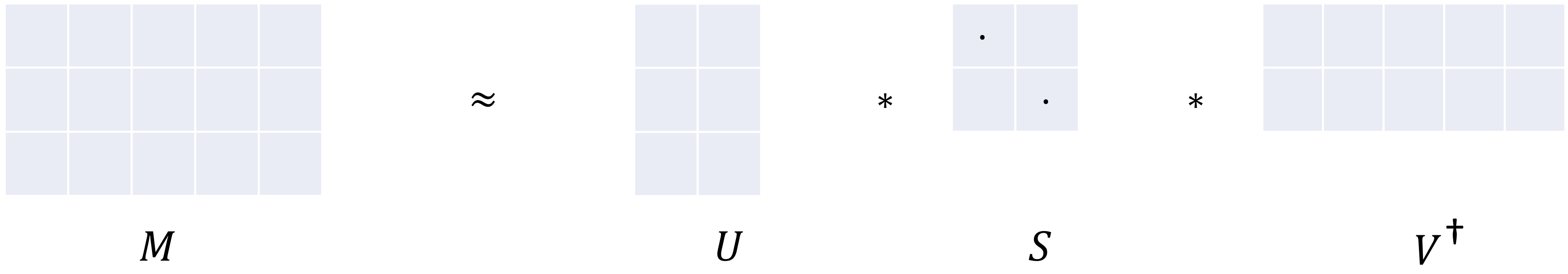


PASQAL

Tensor Networks: SVD compression

SVD – Singular Value Decomposition $M = U\,S\,V^\dagger$





Tensor Networks: SVD compression

Pascal – 95%

10 values

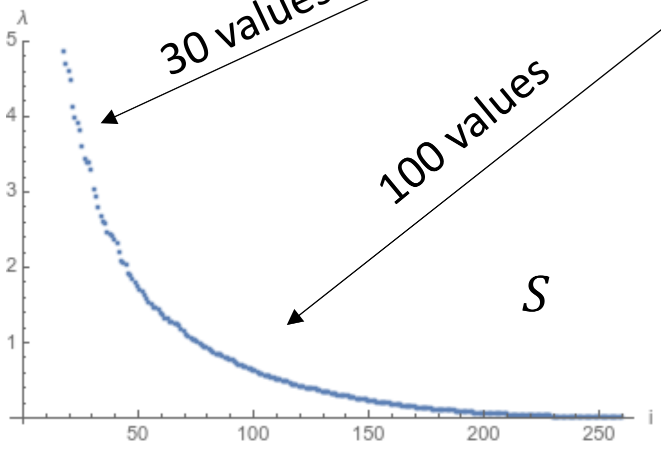
Pascal – 90%

Pascal – 65%

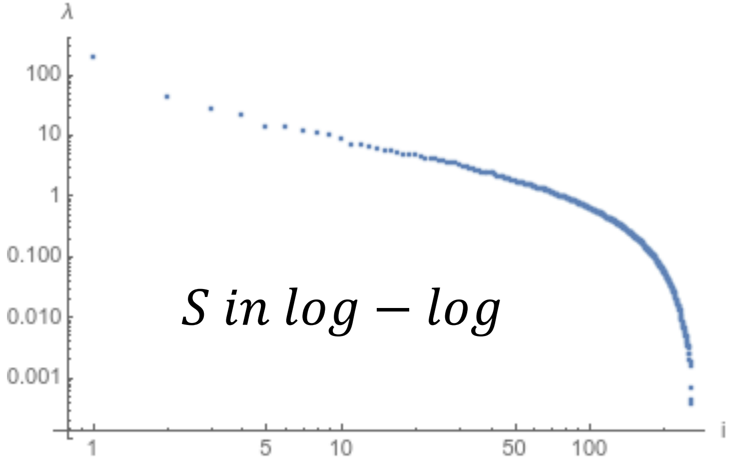
Pascal

30 values

100 values



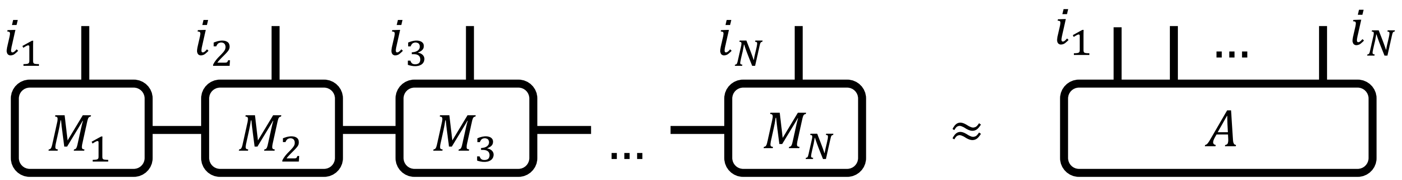
S



$S\ in\ log - log$

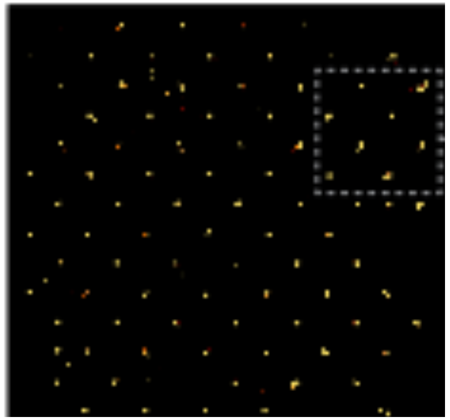
PASQAL

Tensor Networks benchmark

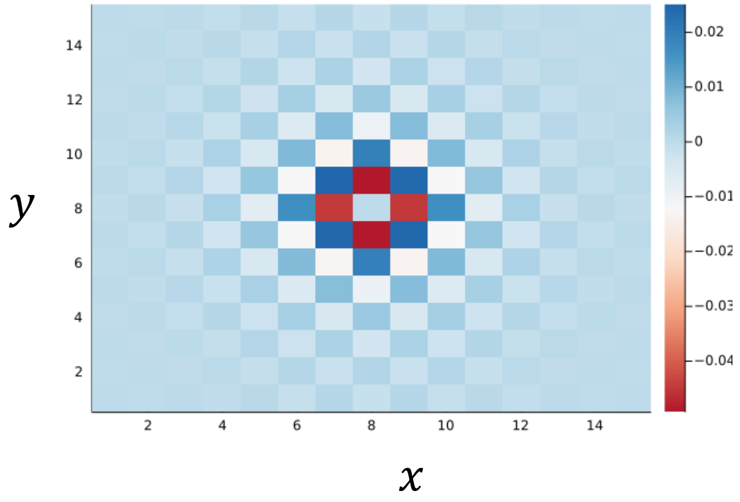


Matrix Product State(MPS)/ Tensor Train

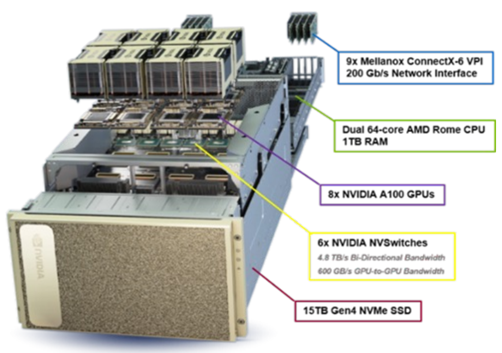
$$g(x, y) = \langle n_x n_y \rangle - \langle n_x \rangle \langle n_y \rangle$$



QPU



TensorNetwork simulation




PASQAL
FRESNEL
PASQAL