Quantized Tensor Train

Entanglement analysis of encoding functions in quantum states

UC Berkeley / Lawrence Berkeley Laboratory Applied Mathematics Seminar

Oct 11 2023

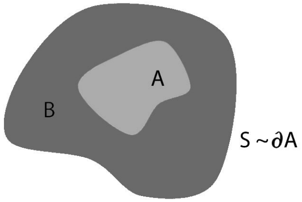
Chris (Jielun) Chen

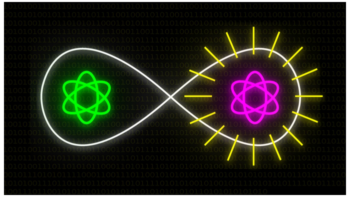
Caltech

Entanglement is one of the most important feature in quantum physics. (2022 Nobel Prize in Physics)

Often studied in the context of:

- Many-body physics
- Quantum cryptography
- Information theory
- …





These topics often consider physical (non-)locality:

$$|...010...\rangle \quad |...111...\rangle$$

However, in quantum algorithms, physical qubits store information "digitally":

$$\sum f(x)|x\rangle$$

No notion of physical locality!

$$|x\rangle = |000...00\rangle, ... \, |011...11\rangle, |100...00\rangle, ...$$

What happens to the entanglement there?

i.e. what properties of *f* affect the state's entanglement?

$\sum f(x)|x\rangle$ What properties of *f* affect the state's entanglement?

Answering this question will help:

1. Understand when might quantum algorithms beat classical (tensor network) algorithms.

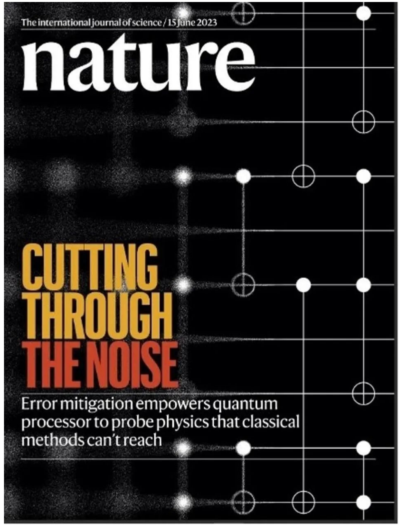
2. Design quantum-inspired classical algorithms.

Good classical algorithm by just "simulation"

Classically doable

Quantum Advantage?

Entanglement



Quantized Tensor Train (QTT)

Quantized Tensor Train

Outline

1. Notations

2. Introduction to QTT

3. Examples of efficient QTT

4. Applications of QTT

5. Summary & Discussion

Focus on QTT representations of functions & operators, rather than QTT algorithms

Quantized Tensor Train

Outline

1. Notations
2. Introduction to QTT
3. Examples of efficient QTT
4. Applications of QTT
5. Summary & Discussion

Notations: bra-ket & entanglement

$$\vec{v} = \begin{pmatrix} v_0 \\ v_1 \\ \vdots \\ v_{N-1} \end{pmatrix} = |v\rangle \qquad \langle v| = (|v\rangle)^\dagger = (v_0^* \ v_1^* \ \dots \ v_{N-1}^*)$$

$$|ab\rangle = |a\rangle|b\rangle = |a\rangle \otimes |b\rangle$$

What I meant by "entanglement":

Schmidt rank ≈ entanglement

Schmidt decomposition

= SVD

$$|\psi\rangle = \sum_{i=1}^{\boxed{\chi}} \sigma_i |L_i\rangle|R_i\rangle$$

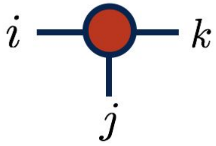
or some measure on how fast singular values decay

Notations: tensor network

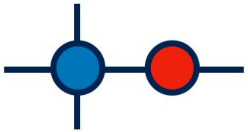
vector v_j

matrix M_{ij}

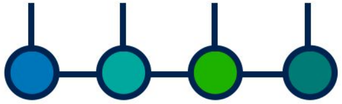
3-index tensor T_{ijk}



Tensor Contraction



$$= \sum_k T_{ijkl} V_{km}$$



$$= \sum_{\alpha_1,\alpha_2,\alpha_3} A^{s_1}_{\alpha_1} B^{s_2}_{\alpha_1\alpha_2} C^{s_3}_{\alpha_2\alpha_3} D^{s_4}_{\alpha_3}$$

tensornetwork.org

Quantized Tensor Train

Outline

1. Notations
2. Introduction to QTT
3. Examples of efficient QTT
4. Applications of QTT
5. Summary & Discussion

Quantized Tenor Train: (very brief) history

Condensed matter physics

1990s: Density Matrix Renormalization Group, Matrix Product State

2000s: Tensor Networks

Independently developed from two communities.

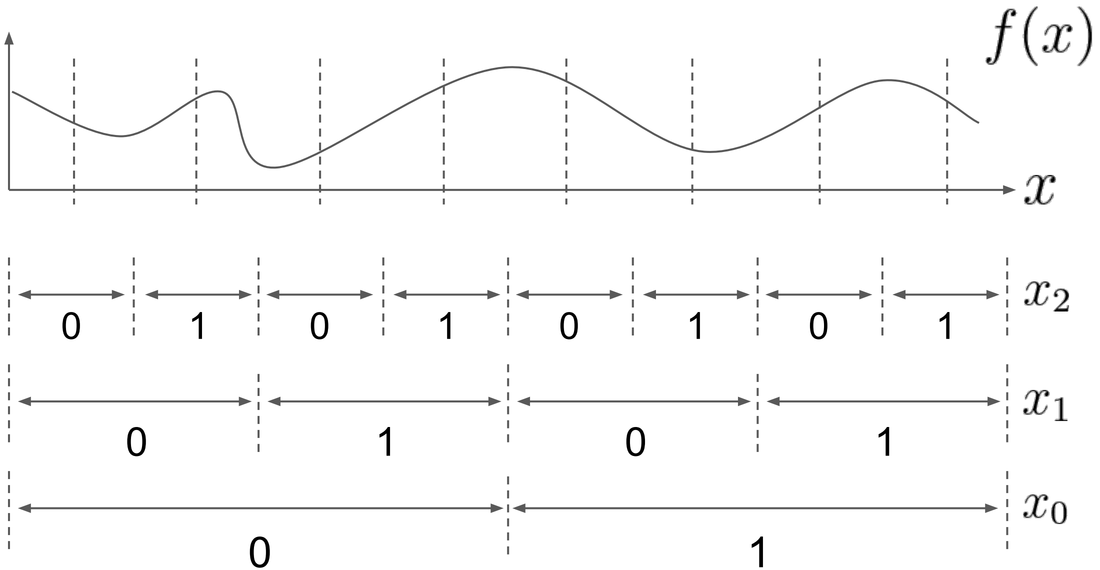
Many things about TT/QTT already known by physicists, but also many new things from math perspective!

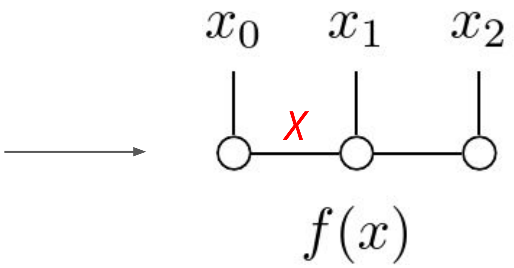
Applied Math

2009: Tensor Train

2010s: Quantized Tensor Train

Quantized Tenor Train for functions





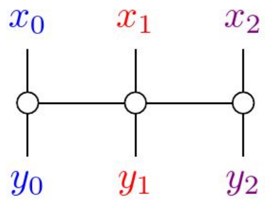
Efficient criteria: bond-dimension χ = poly(n) or even χ = O(1)

Space savings: $2^n \to O(n\chi^2)$

Quantized Tenor Train for operators & 2D functions

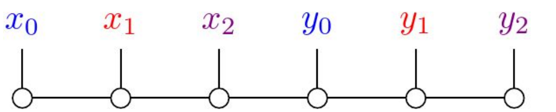
$$
\begin{array}{c|c}
 & \quad y_2 \quad y_1 \qquad\qquad y_0 \\
\hline
\begin{array}{c} x_2 \\ x_1 \\ \\ x_0 \end{array} &
\begin{bmatrix}
a_{11} & a_{12} & a_{13} & a_{14} & a_{15} & a_{16} & a_{17} & a_{18} \\
a_{21} & a_{22} & a_{23} & a_{24} & a_{25} & a_{26} & a_{27} & a_{28} \\
a_{31} & a_{32} & a_{33} & a_{34} & a_{35} & a_{36} & a_{37} & a_{38} \\
a_{41} & a_{42} & a_{43} & a_{44} & a_{45} & a_{46} & a_{47} & a_{48} \\
a_{51} & a_{52} & a_{53} & a_{54} & a_{55} & a_{56} & a_{57} & a_{58} \\
a_{61} & a_{62} & a_{63} & a_{64} & a_{65} & a_{66} & a_{67} & a_{68} \\
a_{71} & a_{72} & a_{73} & a_{74} & a_{75} & a_{76} & a_{77} & a_{78} \\
a_{81} & a_{82} & a_{83} & a_{84} & a_{85} & a_{86} & a_{87} & a_{88}
\end{bmatrix}
\end{array}
$$

operator



subtlety with ordering…

2D function



Space savings: $2^{2n} \to O(n\chi^2)$

Similar scheme for multilinear map & high-dimensional function

Quantized Tensor Train

Outline

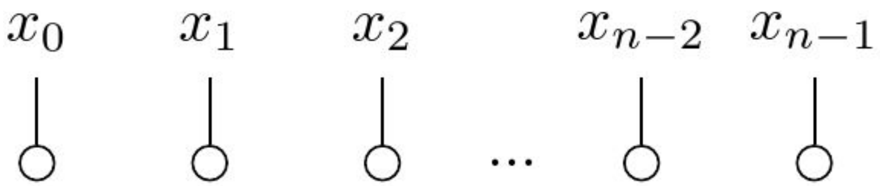
1. Notations
2. Introduction to QTT
3. Examples of efficient QTT (functions)
4. Applications of QTT
5. Summary & Discussion

Efficient QTT functions: exp(x)

exp(x) is a product of exponentials of individual bits:

$$\begin{aligned} e^x &= e^{x_0 + 2x_1 + 2^2 x_2 + \ldots} \\ &= \prod e^{2^i x_i} \end{aligned} \longrightarrow \begin{pmatrix} 1 \\ 1 \end{pmatrix} \otimes \begin{pmatrix} 1 \\ e^2 \end{pmatrix} \otimes \begin{pmatrix} 1 \\ e^{2^2} \end{pmatrix} \otimes \cdots$$

which corresponds to a $\chi = 1$ QTT:

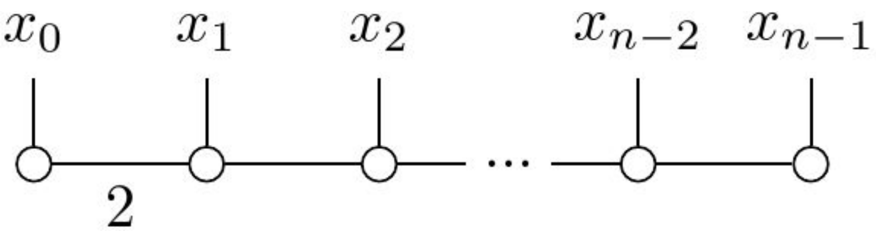


Efficient QTT functions: cos(x) & sin(x)

cos(x) & sin(x) = a sum of two exponentials:

$$\cos(x) = \frac{e^{ix} + e^{-ix}}{2} \qquad \sin(x) = \frac{e^{ix} - e^{-ix}}{2i}$$

which corresponds to a χ = 2 QTT: (canonical polyadic decomposition)



Generally:
$\chi \leq$ #Fourier coefficients

Efficient QTT functions: polynomials (1st order)

First-order polynomial: $x = x_0 + 2x_1 + 2^2 x_2 \ldots$

Want to construct QTT state $|x^1\rangle$ s.t. $\langle x_0 x_1 x_2 \ldots | x^1\rangle = x_0 + 2x_1 + 2^2 x_2 \ldots$

Solution:

$$|x^1\rangle = \sum_{j=0}^{n-1} |+\rangle^{\otimes j} |v_j^1\rangle |+\rangle^{\otimes n-j-1}$$

$$|v_j^k\rangle = \begin{pmatrix} 0 \\ 2^{jk} \end{pmatrix} \qquad |+\rangle = \begin{pmatrix} 1 \\ 1 \end{pmatrix}$$

$$\langle x_j | v_j^1 \rangle = 2^j x_j$$
$$\langle x_j | + \rangle = 1$$

χ = 2 QTT format:

$$\begin{pmatrix} |+\rangle & |v_0^1\rangle \end{pmatrix} \;—\; \begin{pmatrix} |+\rangle & |v_1^1\rangle \\ 0 & |+\rangle \end{pmatrix} \;—\; \begin{pmatrix} |+\rangle & |v_2^1\rangle \\ 0 & |+\rangle \end{pmatrix} \;—\; \cdots \;—\; \begin{pmatrix} |v_{n-1}^1\rangle \\ |+\rangle \end{pmatrix}$$

Efficient QTT functions: polynomials (2nd order)

Second-order polynomial: $x^2 = (x_0 + 2x_1 + 2^2 x_2 \ldots)^2 = \sum x_j x_k 2^{j+k}$

In the first-order example we defined $|v_j^1\rangle$ s.t. $\langle x_j | v_j^1 \rangle = x_j 2^j$

Extending to second order:

$$|x^2\rangle = 2 \sum_{j<k} |+\rangle^{\otimes j} |v_j^1\rangle |+\rangle^{\otimes k-j-1} |v_k^1\rangle |+\rangle^{\otimes n-k-1} + \sum_j |+\rangle^{\otimes j} |v_j^2\rangle |+\rangle^{\otimes n-j-1}$$

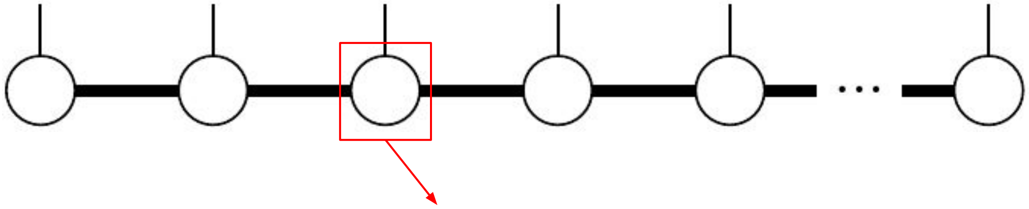
Corresponding to a $\chi = 3$ QTT format:

$$\begin{pmatrix} |+\rangle & 2|v_0^1\rangle & |v_0^2\rangle \end{pmatrix} \text{---} \begin{pmatrix} |+\rangle & 2|v_1^1\rangle & |v_1^2\rangle \\ 0 & |+\rangle & |v_1^1\rangle \\ 0 & 0 & |+\rangle \end{pmatrix} \text{---} \begin{pmatrix} |+\rangle & 2|v_2^1\rangle & |v_2^2\rangle \\ 0 & |+\rangle & |v_2^1\rangle \\ 0 & 0 & |+\rangle \end{pmatrix} \text{---} \cdots \text{---} \begin{pmatrix} |v_{n-1}^2\rangle \\ |v_{n-1}^1\rangle \\ |+\rangle \end{pmatrix}$$

Efficient QTT functions: polynomials (higher order)

Extending to higher order:

x^k



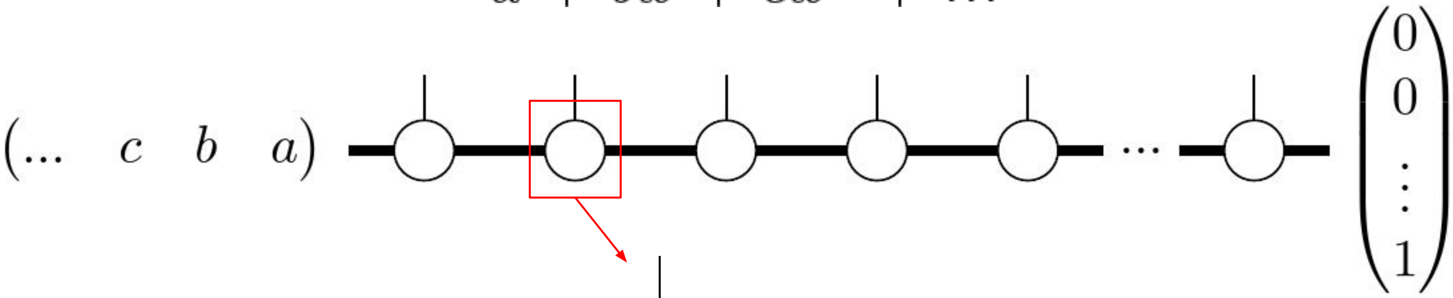
$$
\begin{pmatrix}
\cdots & & & & \\
|+\rangle & 4|v_j^1\rangle & 6|v_j^2\rangle & 4|v_j^3\rangle & |v_j^4\rangle \\
0 & |+\rangle & 3|v_j^1\rangle & 3|v_j^2\rangle & |v_j^3\rangle \\
0 & 0 & |+\rangle & 2|v_j^1\rangle & |v_j^2\rangle \\
0 & 0 & 0 & |+\rangle & |v_j^1\rangle \\
0 & 0 & 0 & 0 & |+\rangle
\end{pmatrix}
$$

χ = k+1

Efficient QTT functions: polynomials (general coefficients)

Boundary tensor determines polynomial coefficients:

$$a + bx + cx^2 + \ldots$$



$$\begin{pmatrix} \ldots & & & & \\ |+\rangle & 4|v_j^1\rangle & 6|v_j^2\rangle & 4|v_j^3\rangle & |v_j^4\rangle \\ 0 & |+\rangle & 3|v_j^1\rangle & 3|v_j^2\rangle & |v_j^3\rangle \\ 0 & 0 & |+\rangle & 2|v_j^1\rangle & |v_j^2\rangle \\ 0 & 0 & 0 & |+\rangle & |v_j^1\rangle \\ 0 & 0 & 0 & 0 & |+\rangle \end{pmatrix}$$

χ = k+1

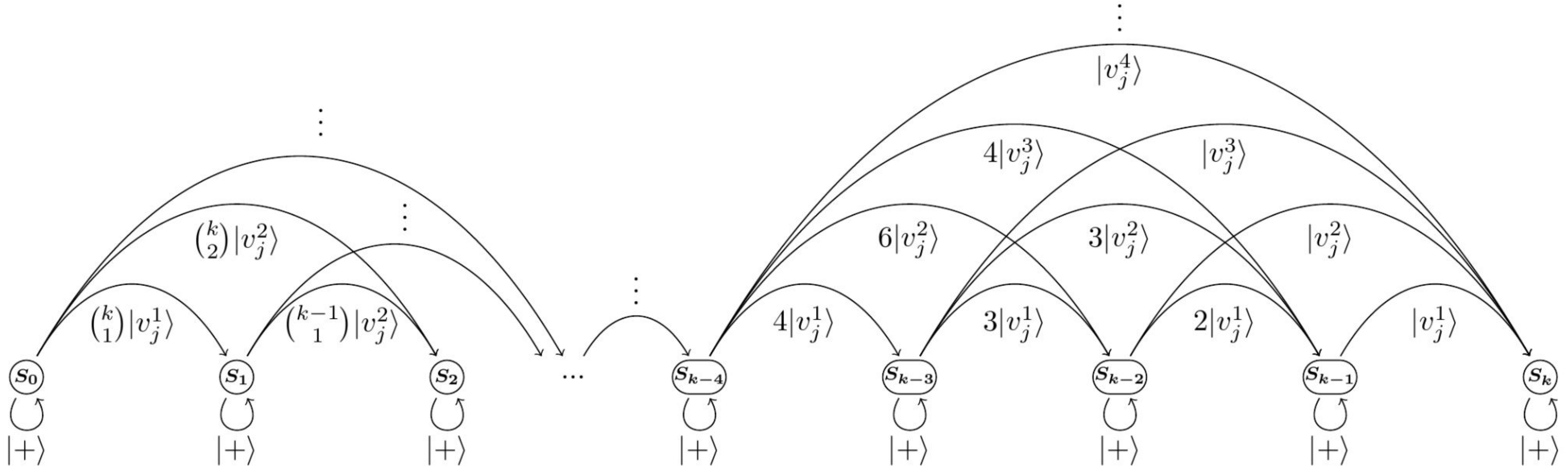
Efficient QTT functions: polynomials (finite state machine)

Can be viewed as a finite state machine.

Related: Hamiltonian MPO as FSM

Crosswhite, Bacon, Phys. Rev. A 78, 012356 (2008)
Crosswhite, Doherty, Vidal, Phys. Rev. B 78, 035116 (2008)
Motruk, Zaletel, Mong, Pollmann, Phys. Rev. B 93, 155139 (2016)
...

$$\begin{pmatrix} \cdots & & & & \\ |+\rangle & 4|v_j^1\rangle & 6|v_j^2\rangle & 4|v_j^3\rangle & |v_j^4\rangle \\ 0 & |+\rangle & 3|v_j^1\rangle & 3|v_j^2\rangle & |v_j^3\rangle \\ 0 & 0 & |+\rangle & 2|v_j^1\rangle & |v_j^2\rangle \\ 0 & 0 & 0 & |+\rangle & |v_j^1\rangle \\ 0 & 0 & 0 & 0 & |+\rangle \end{pmatrix}$$

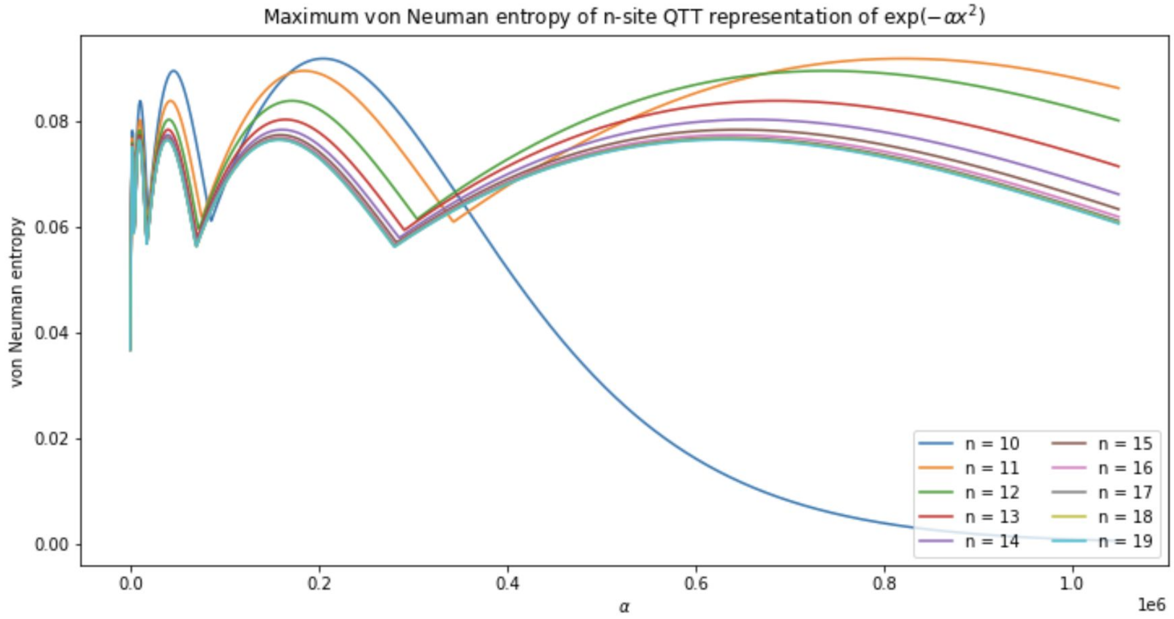


Efficient QTT functions: Gaussian

QTT for $e^{-\alpha x^2}$ has error upper-bounded by ~ $O(\chi e^{-\chi^2/\alpha})$

Dolgov, Khoromskij, Oseledets, SIAM (2012), 34, 6

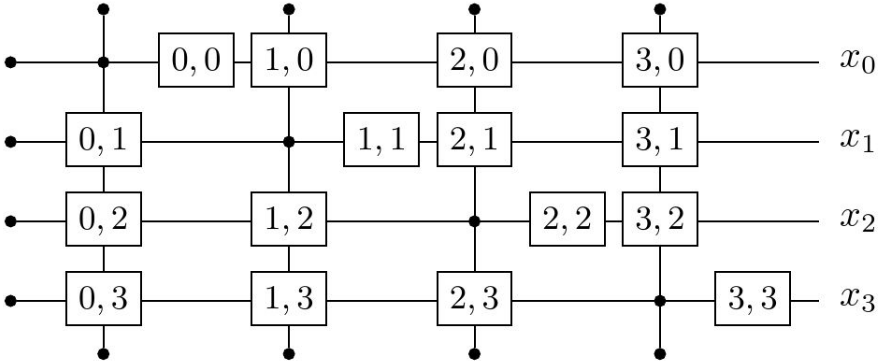
Numerical experiments showed for almost all α, $\chi = O(1)$



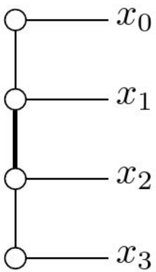
Efficient QTT functions: Gaussian

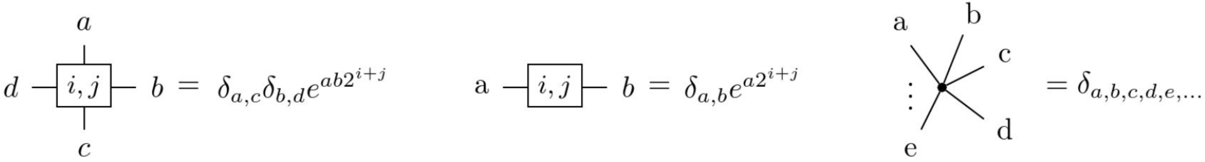
Hard to write out each QTT site, but can contract an *n* × *n* tensor network

$$e^{(x_0+2x_1+2^2x_2+2^3x_3)^2}$$



contract $\Rightarrow$





Quantized Tensor Train

Outline

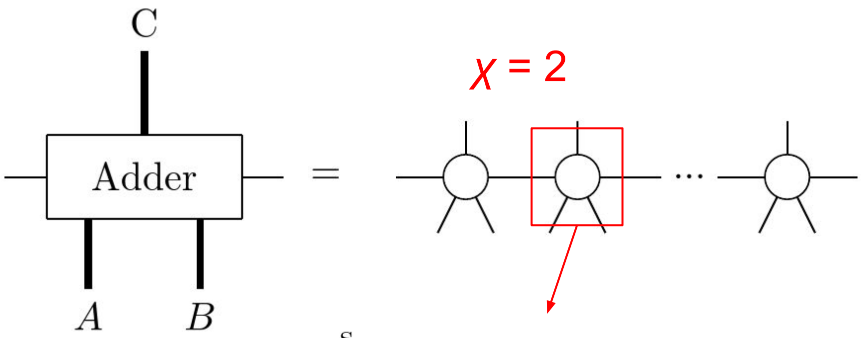
1. Notations
2. Introduction to QTT
3. Examples of efficient QTT (operators)
4. Applications of QTT
5. Summary & Discussion

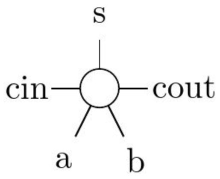
Efficient QTT operators: addition

Addition is defined to be the following linear map:

$$|A\rangle|B\rangle \rightarrow |A+B\rangle$$

The QTT can be obtained directly from a Ripple-carry adder circuit:





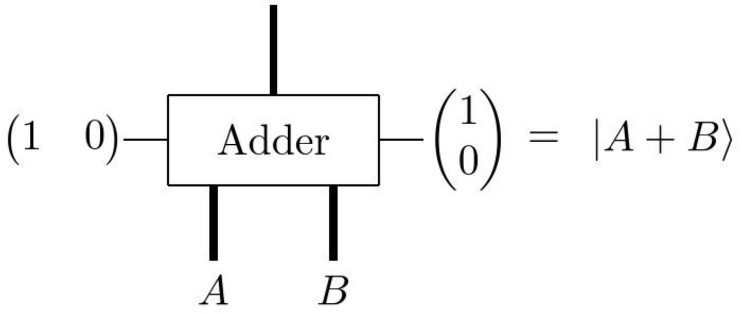
$$= \begin{cases} 1 & \text{a, b, cin, s, cout in the table} \\ 0 & \text{otherwise} \end{cases}$$

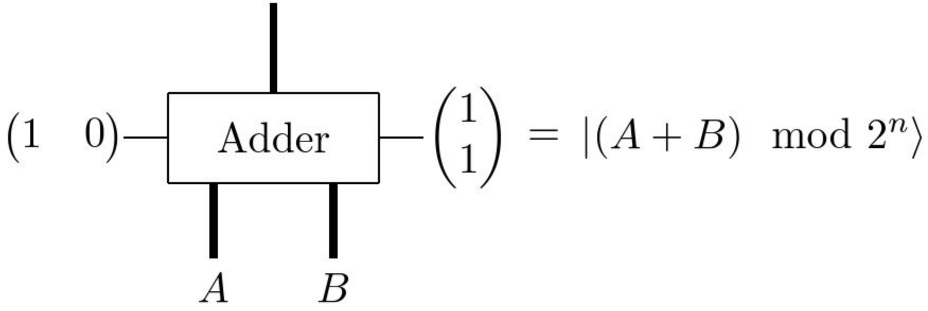
Full adder truth table

a	b	cin	s	cout
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

Efficient QTT operators: addition

Boundary tensors determines modulation:





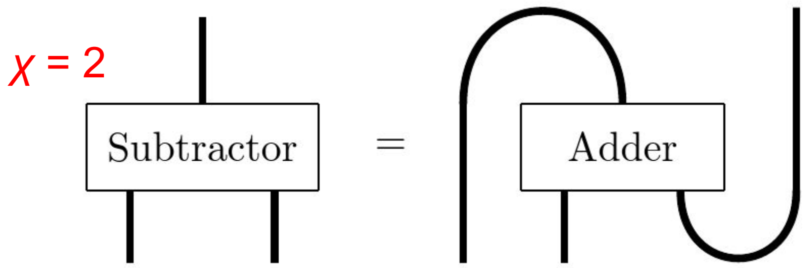
Building block for many operators!

Efficient QTT operators: subtraction

Subtraction is defined to be the following linear map:

$$|A\rangle|B\rangle \rightarrow |A - B\rangle$$

Subtractor in QTT = reshaped adder in QTT:

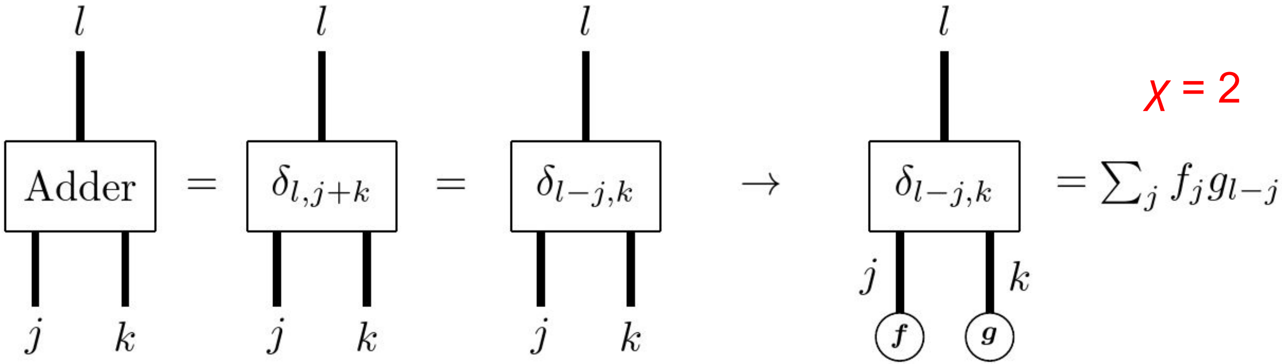


Efficient QTT operators: convolution

Convolution is defined as the linear map:

$$\sum_j f_j |j\rangle \otimes \sum_k g_k |k\rangle \rightarrow \sum_l \left(\sum_j f_j g_{l-j} \right) |l\rangle$$

It turns out convolution in QTT = addition in QTT:



$\chi = 2$

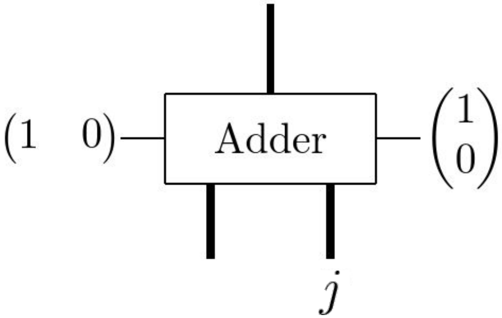
circular convolution = modulo adder

Efficient QTT operators: shift matrix

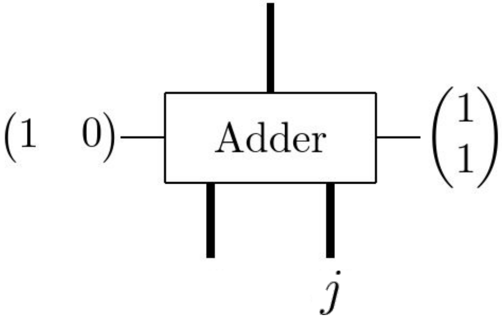
A (non-)circular shift matrix is defined as:

$$
\begin{array}{cc}
j\left\{\begin{array}{c} \\ \\ \\ \\ \end{array}\right. & \\[-6em]
 & \begin{bmatrix}
0 & 0 & 0 & 0 & 0 & \dots & 0 \\
\vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\
0 & 0 & 0 & 0 & 0 & \dots & 0 \\
1 & 0 & 0 & 0 & 0 & \dots & 0 \\
0 & 1 & 0 & 0 & 0 & \dots & 0 \\
\vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\
0 & 0 & 0 & 1 & 0 & \dots & 0
\end{bmatrix}
\end{array}
\qquad
\begin{array}{cc}
j\left\{\begin{array}{c} \\ \\ \\ \\ \end{array}\right. & \\[-6em]
 & \begin{bmatrix}
0 & 0 & 0 & 0 & 1 & \dots & 0 \\
\vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\
0 & 0 & 0 & 0 & 0 & \dots & 1 \\
1 & 0 & 0 & 0 & 0 & \dots & 0 \\
0 & 1 & 0 & 0 & 0 & \dots & 0 \\
\vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\
0 & 0 & 0 & 1 & 0 & \dots & 0
\end{bmatrix}
\end{array}
$$

Corresponding to adding index by *j*:



$\chi = 2$



Efficient QTT operators: Toeplitz matrix

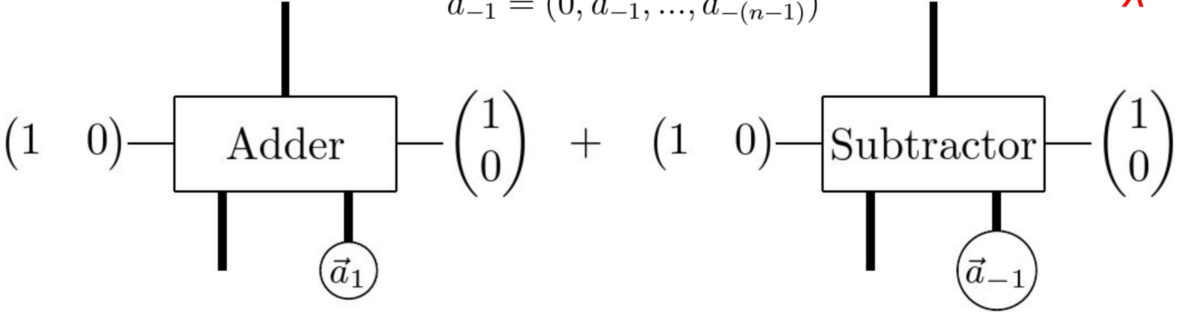
A Toeplitz matrix has the form:

$$\begin{bmatrix} a_0 & a_{-1} & a_{-2} & \cdots & \cdots & a_{-(n-1)} \\ a_1 & a_0 & a_{-1} & \ddots & & \vdots \\ a_2 & a_1 & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & a_{-1} & a_{-2} \\ \vdots & & \ddots & a_1 & a_0 & a_{-1} \\ a_{n-1} & \cdots & \cdots & a_2 & a_1 & a_0 \end{bmatrix}$$

Appear frequently in signal processing, numerical analysis, differential equations…

Corresponding to the sum:

$$\vec{a}_1 = (a_0, \ldots, a_{n-1})$$
$$\vec{a}_{-1} = (0, a_{-1}, \ldots, a_{-(n-1)})$$



$\chi \leq 2\chi(\vec{a}_1) + 2\chi(\vec{a}_{-1})$

Small χ for e.g. banded Toeplitz

Efficient QTT operators: circulant matrix

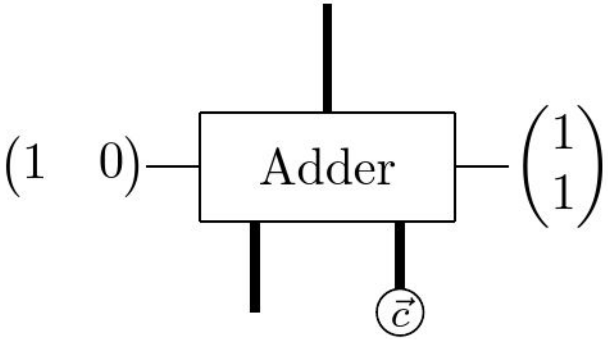
A circulant matrix has the form:

$$\begin{bmatrix} c_0 & c_{n-1} & \cdots & c_2 & c_1 \\ c_1 & c_0 & c_{n-1} & & c_2 \\ \vdots & c_1 & c_0 & \ddots & \vdots \\ c_{n-2} & & \ddots & \ddots & c_{n-1} \\ c_{n-1} & c_{n-2} & \cdots & c_1 & c_0 \end{bmatrix}$$

Special case of Toeplitz

Diagonalizable by discrete Fourier transform

Corresponding to circular convolution with vector $\vec{c} = (c_0, c_1, ..., c_{n-1})$:



$\chi \leq 2\chi(\vec{c})$

Efficient QTT operators: discrete Fourier transform

Discrete Fourier transform (DFT):

$$F_n = \frac{1}{\sqrt{2^n}} \begin{pmatrix} 1 & 1 & 1 & \dots & 1 \\ 1 & \omega & \omega^2 & \dots & \omega^{2^n-1} \\ 1 & \omega^2 & \omega^4 & \dots & \omega^{2(2^n-1)} \\ \dots & \dots & \dots & \dots & \dots \\ 1 & \omega^{2^n-1} & \omega^{2(2^n-1)} & \dots & \omega^{(2^n-1)(2^n-1)} \end{pmatrix} \qquad \omega = \exp(i2\pi/2^n)$$

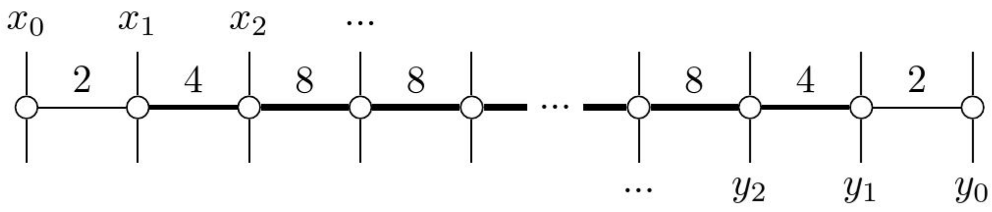
DFT is well-approximated by a QTT with error $O(ne^{-\chi \log(\chi/3)})$.

i.e. χ grows sub-logarithmically to maintain a constant global error.

Numerics suggest χ = 8 gives error below 10^{-15}.

JC, Stoudenmire, White, arXiv:2210.08468 (accepted to PRX quantum)

$F_n[x, y] \approx$



reversed ordering is important!

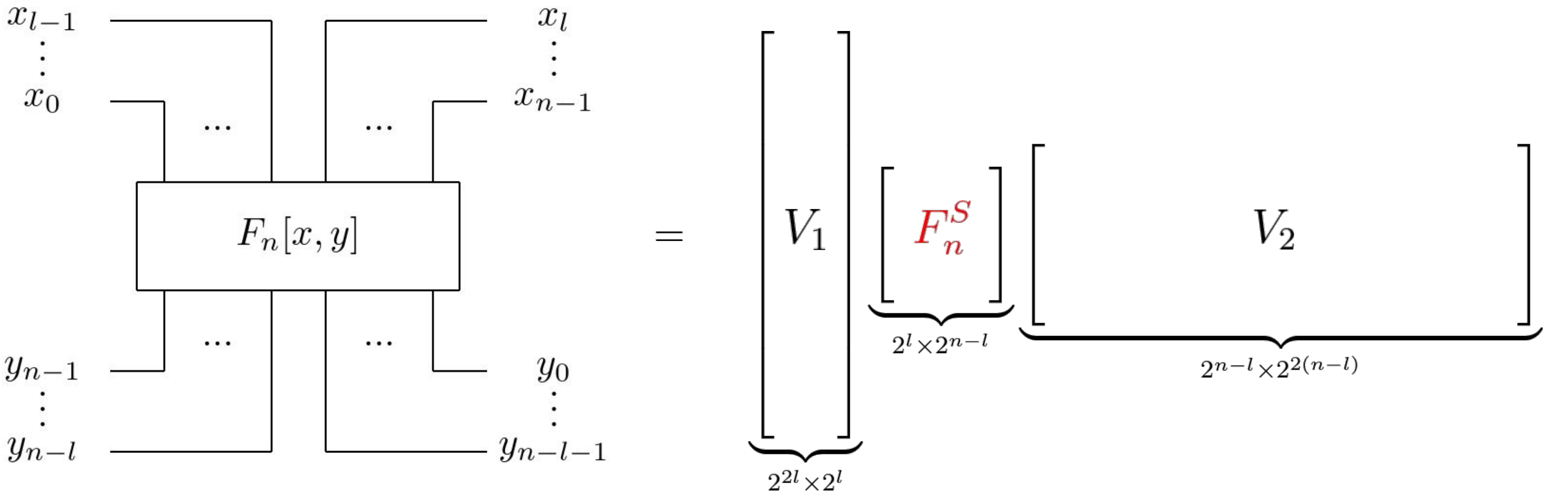
Efficient QTT operators: discrete Fourier transform

Why is DFT compressible in QTT:

$$F_n^S = 2^l \begin{array}{c} \overbrace{\hphantom{1 \quad \omega \quad \omega^2 \quad \cdots}}^{2^{n-l}} \\ \begin{pmatrix} 1 & 1 & 1 & \cdots & 1 \\ 1 & \omega & \omega^2 & \cdots & \omega^{2^n-1} \\ 1 & \omega^2 & \omega^4 & \cdots & \omega^{2(2^n-1)} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ 1 & \omega^{2^n-1} & \omega^{2(2^n-1)} & \cdots & \omega^{(2^n-1)(2^n-1)} \end{pmatrix} \end{array}$$

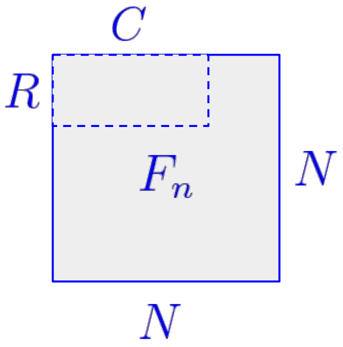
DFT's QTT rank

DFT's submatrix rank



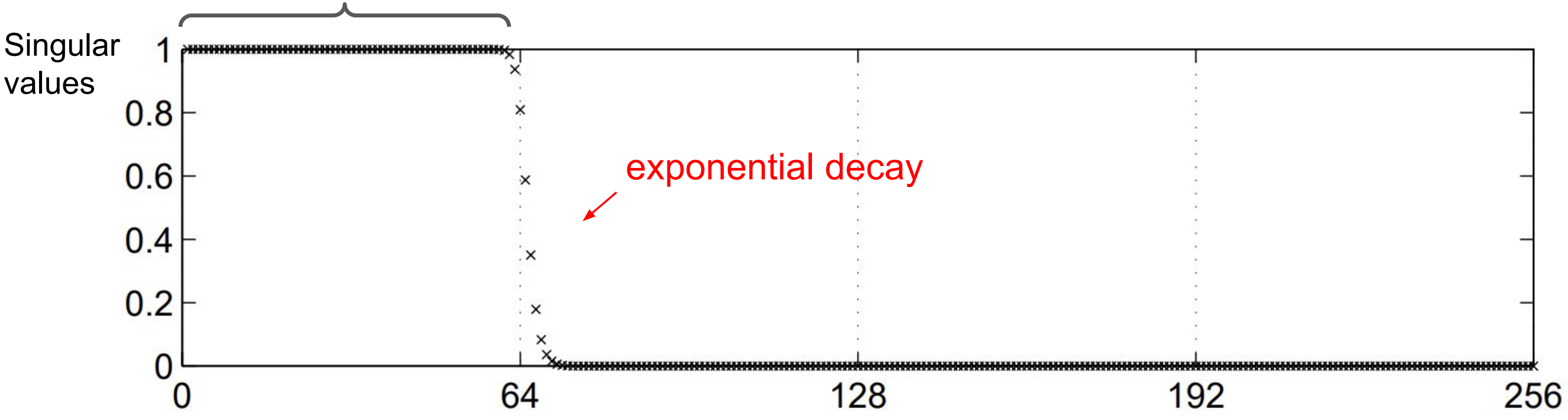
Efficient QTT operators: discrete Fourier transform

For an $R \times C$ submatrix of the $N \times N$ DFT, its effective rank is very small.



$\boxed{2^l \times 2^{n-l}/2^n = 1}$

$\sim RC/N$



[1]

[1] Edelman, A et. al., The Future Fast Fourier Transform?

Efficient QTT operators: derivatives

Option 1: finite difference method $\chi \leq 2$(FDM order + derivative order)

$$\frac{\partial^2}{\partial x^2} \sim \begin{pmatrix} -2 & 1 & 0 & \dots & 0 & 1 \\ 1 & -2 & 1 & \dots & 0 & 0 \\ 0 & 1 & -2 & \dots & 0 & 0 \\ \dots & & & & & \\ 1 & 0 & 0 & \dots & 1 & -2 \end{pmatrix} = -2I + \begin{pmatrix} 0 & 0 & 0 & \dots & 0 & 1 \\ 1 & 0 & 0 & \dots & 0 & 0 \\ 0 & 1 & 0 & \dots & 0 & 0 \\ \dots & & & & & \\ 0 & 0 & 0 & \dots & 1 & 0 \end{pmatrix} + \begin{pmatrix} 0 & 1 & 0 & \dots & 0 & 0 \\ 0 & 0 & 1 & \dots & 0 & 0 \\ 0 & 0 & 0 & \dots & 0 & 0 \\ \dots & & & & & \\ 1 & 0 & 0 & \dots & 0 & 0 \end{pmatrix}$$

Option 2: diagonalization by DFT $\chi \leq \chi(DFT)^2$ * (derivative order + 1)

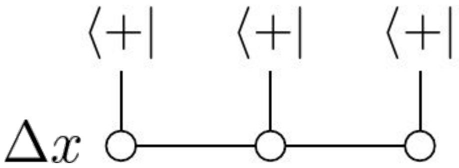
$$\frac{\partial^2}{\partial x^2} \sim \mathrm{DFT}^{-1} \cdot \mathrm{diag}(x^2) \cdot \mathrm{DFT}$$

Efficient QTT operators: integral

First order approximation to the integral:

$$\int_{x_0}^{x_N} f(x)dx \approx \sum_{j=0}^{N-1} f(x_j)\Delta x = \Delta_x \langle + |^{\otimes n} | f \rangle$$

Corresponding to inner product with $\chi = 1$ QTT:

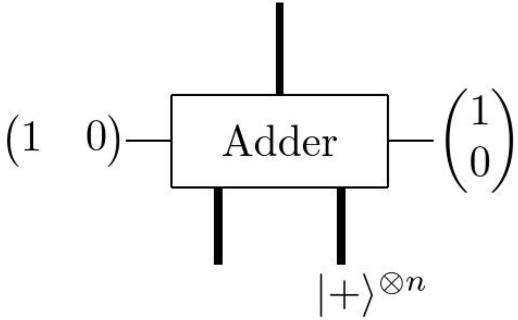


Efficient QTT operators: integral with variable range

Integral with variable range:

$$g(x) = \int_0^x f(x')dx \approx \Delta x \begin{bmatrix} 1 & 0 & \dots & 0 & 0 \\ 1 & 1 & \dots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 1 & 1 & \dots & 1 & 0 \\ 1 & 1 & \dots & 1 & 1 \end{bmatrix} \begin{bmatrix} f(x'_0) \\ f(x'_1) \\ \vdots \\ f(x'_{N-2}) \\ f(x'_{N-1}) \end{bmatrix}$$

The matrix corresponds to a $\chi = 2$ QTT:



Efficient QTT for e.g.

$$\operatorname{erf} z = \frac{2}{\sqrt{\pi}} \int_0^z e^{-t^2}\,\mathrm{d}t.$$

Other efficient QTT in literature

Wavelets as QTT

Oseledets Tyrtyshnikov, Algebraic
Wavelet Transform via Quantics
Tensor Train Decomposition

Image Compression with QTT

Latorre, Image compression and entanglement, arXiv:quant-ph/0510031, 2005

Green’s functions of quantum many-body systems as QTT

Shinaoka, Wallerberger, Murakami, Nogaki, Sakurai, Werner, and
Kauch, Multiscale Space-Time Ansatz for Correlation Functions
of Quantum Systems Based on Quantics Tensor Trains

…

Major open question: when is QTT efficient in general?

smoothness?

QTT can embed both very smooth or very sharp functions

Uniform distribution

Small rank

Lipschitz-continuous functions

Exponential cusps

Delta-function

Small rank

#Fourier coeffs

Smoothness

Outlier: Gaussian?

Some rigorous results, e.g. classical Besov smoothness implies QTT

Ali & Nouy, Constructive Approximation volume 58, pages463–544 (2023)

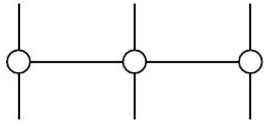
Major open question: when is QTT efficient in general?

Recursion & Fractal structure?

Recursive construction → QTT

How to formalize?

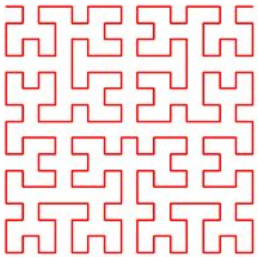
$$A = \left[\begin{array}{cc|cc|cc|cc}
a & f & & & & & & \\
b & a & f & & & & & \\
\hline
c & b & a & f & & & & \\
d & c & b & a & f & & & \\
\hline
 & d & c & b & a & f & & \\
 & & d & c & b & a & f & \\
\hline
 & & & d & c & b & a & f \\
 & & & & d & c & b & a
\end{array}\right]$$

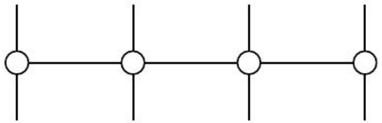


$\chi = 4$

Entropy of fractal systems

Zmeskal, Dzik, Vesely, Computers & Mathematics with Applications, Volume 66, Issue 2, 2013,





$\chi = 5$

Quantized Tensor Train

Outline

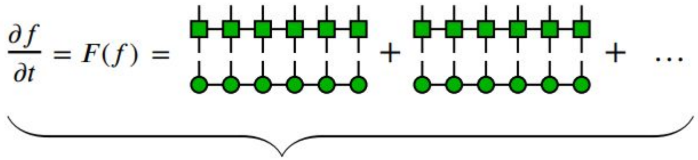
1. Notations
2. Introduction to QTT
3. Examples of efficient QTT
4. Applications of QTT
5. Summary & Discussion

Applications: plasma physics

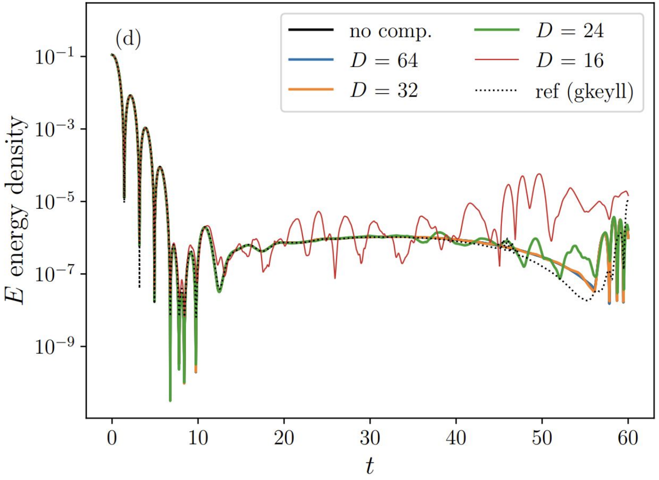
Ye, Loureiro, Phys. Rev. E 106, 035208 (2022)

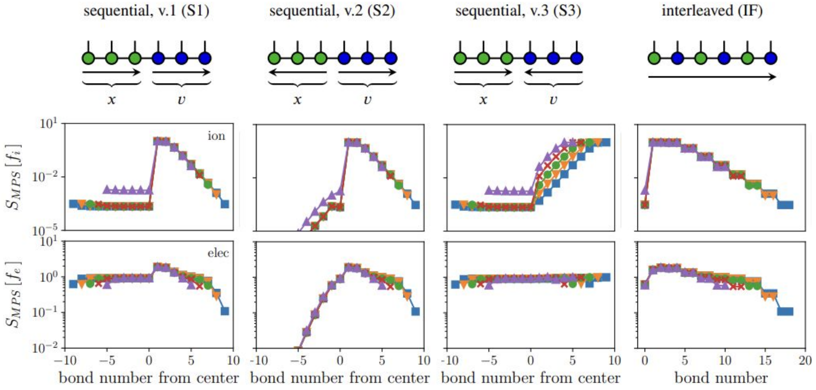
Solving the Vlasov-Poisson equation by time evolution in QTT:

$$\frac{\partial f_s}{\partial t} + \mathbf{v}_s \cdot \nabla_{\mathbf{r}} f_s + \frac{q_s}{m_s}\mathbf{E}\cdot\nabla_{\mathbf{v},s} f_s = \mathcal{C}[f_s]$$



$$f(t+\Delta t) \approx f(t) + \frac{\partial f}{\partial t}\Delta t$$





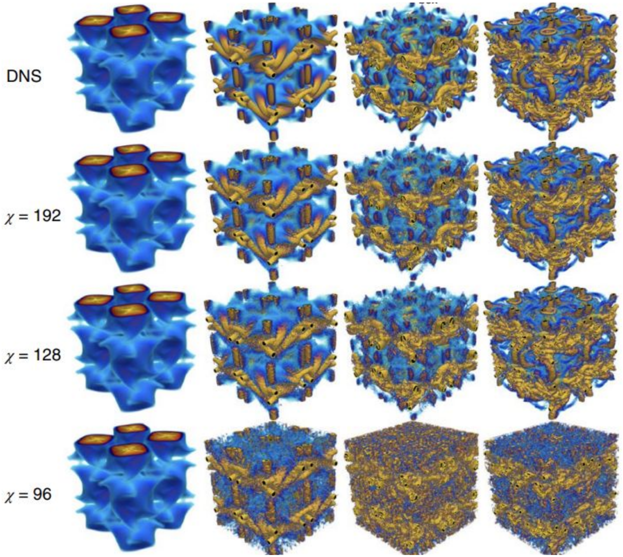
Applications: turbulence

Gourianov, Lubasch, Dolgov, van den Berg, Babaee, Givi, Kiffner, Jakscha, Nature Computational Science (2022)

Solving the incompressible Navier–Stokes equations iteratively in QTT :

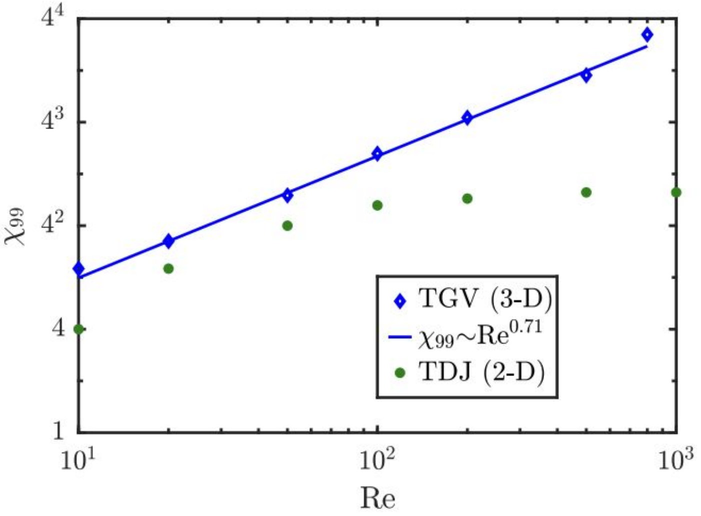
$$\nabla \cdot V = 0$$

$$\frac{\partial V}{\partial t} + (V \cdot \nabla)V = -\nabla p + \nu \nabla^2 V,$$



2D: QTT-rank saturates for Reynold number ≥ 200

3D: QTT-rank increases according to a power law



Applications: quantum chemistry

Physical orbitals tend to be smooth → efficient QTT

Jolly, Fernández, Waintal, arXiv:2308.03508

Solve Hartree-fock in QTT iteratively using DMRG (minimization):

$$\boxed{\left(-\frac{1}{2}\nabla^2 + V_{ion} + J[\rho] + K[\{\phi_j\}]\right)\phi_i = \epsilon_i \phi_i}$$

Work in progress with Sandeep Sharma & Garnet Chan

$$V_{ion} = \sum_A \frac{Z_A}{|R_A - r|}$$

$$J[\rho] = \int \frac{\rho(r')}{|r - r'|} dr'$$

$$K[\{\phi_j\}]\phi_i = \sum_j \phi_j(r) \int \frac{\phi_j^*(r')\phi_i(r')}{|r - r'|} dr'$$

$$\frac{1}{r} \approx \sum_i c_i e^{-\alpha_i r^2}$$

Applications: “superfast” Fourier transform

Assume an input vector v has length $N = 2^n$; want to compute DFT(v).

- Time complexity for the fast Fourier transform:

$$O(N \log N) = O(2^n n)$$

dominates time complexity

- Total time complexity for converting v to QTT with rSVD + DFT QTT:

$O(2^n \chi)$ if data can be compressed into an QTT with rank χ

χ	$O(1)$	$O(n)$	$O(2^{n/2})$
Total time	$O(2^n)$	$O(2^n n)$	$O(2^{3n/2})$

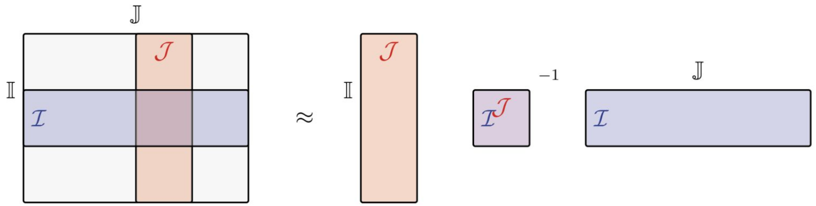
“superfast” Fourier transform

Connection to sparse Fourier transform?

Side note: convert vector into QTT

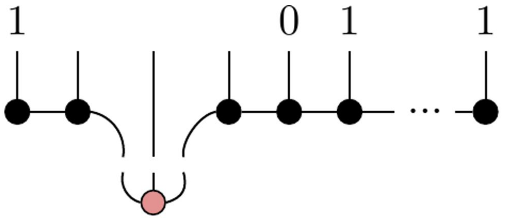
Converting an exponentially-long vector to QTT takes exponential time with SVD, even when QTT is efficient. What are some other methods?

Cross-interpolation (iterate through all cuts):



Dolgov, Savostyanov, Computer Physics Communications, Volume 246, 2020

DMRG-like method:
Initial guess → sampling environment → solve local LSE → sweep



$$= f(1, x_2, x_3, x_4, 0, 1, ..., 1)$$

Improve & Rigorous guarantee?

Summary & Discussion

- Efficient QTT construction for many important functions & operators

- Formalize efficient criteria for QTT?

- Directly connect to entanglement in quantum algorithms.

- Already been applied to many real-world differential equations.

Acknowledgement

Many thanks to Michael Lindsey for invitation

Thanks for discussions: Sandeep Sharma, Garnet Chan, Miles Stoudenmire, Steve White, Michael Lindsey, Lin Lin, Erika Ye, Aaron Szasz, Kevin Stubbs, Michael Kielstra