CSE 392: Matrix and Tensor Algorithms for Data

Instructor: Shashanka Ubaru

University of Texas, Austin

Spring 2024

Lecture 24: Tensor networks

Outline

What are Tensor Networks ?

- A **network** of **tensors**
- Alternative formulation to the standard, cumbersome algebraic tensor representation
- Conceived by Roger Penrose in 1971 *"It now ceases to be important to maintain a distinction between upper and lower indices"*
- **Instrumental** in tensor **computation** and **analysis**

Figure: Roger Penrose

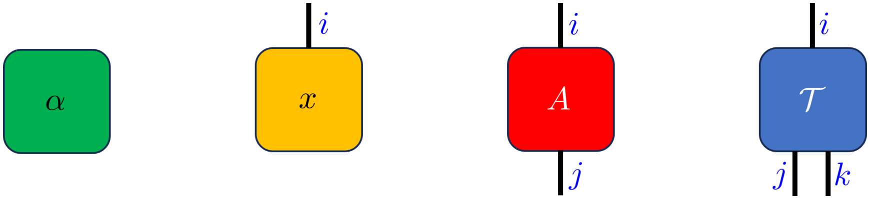
R Penrose. Applications of negative dimensional tensors. Combinatorial Mathematics and its Applications, Academic Press, 1971

What are Tensor Networks ?

- **Nodes** (or vertices) represent individual **tensors**
- **Edges** are (typically) non-directed and represent tensor index
- **Connected** (standard) edges represent (Einstein) **summation** over an index
- Free (dangling) indices depicted as edges attached to a single vertex
- Self-connecting edge (from a tensor to itself) represents **trace** operation
- Number of edges on nodes indicate the **order** of the tensor

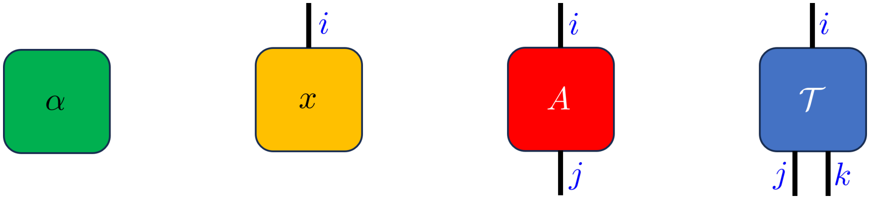
What are Tensor Networks ?

- What are these tensor networks objects ?



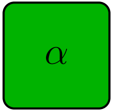
What are Tensor Networks ?

- What are these tensor networks objects ?



What are Tensor Networks ?

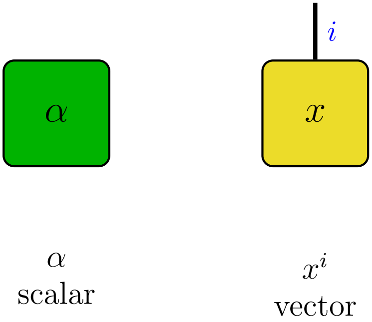
- What are these tensor network objects?



α
scalar

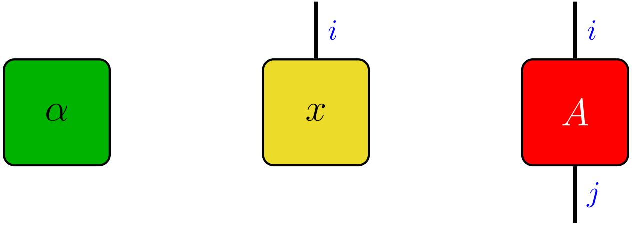
What are Tensor Networks ?

- What are these tensor network objects?



What are Tensor Networks ?

- What are these tensor network objects?



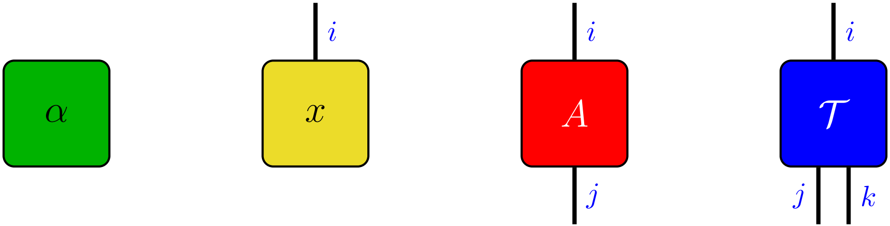
α
scalar

x^i
vector

A^i_j
matrix

What are Tensor Networks ?

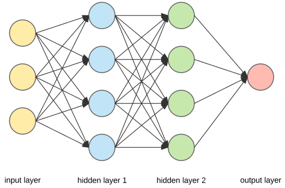
- What are these tensor network objects?



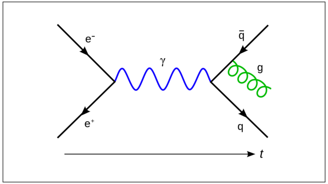
α	x^i	A^i_j	T^i_{jk}
scalar	vector	matrix	tensor

Tensor Network Applications

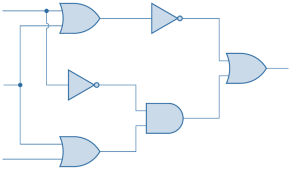
- Some examples of tensor networks



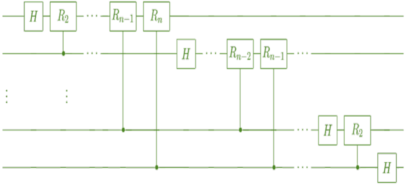
neural networks



Feynman diagrams



logical circuits



quantum circuits

How Powerful are Tensor Networks ?

- Tensor networks invariants / isomorphism offers means to analyze and identify (space and time complexity) structure in high dimensional computation

MENU **nature**

Article | Published: 23 October 2019

Quantum supremacy using a programmable superconducting processor

Frank Arute, Kunal Arya, [...] John M. Martinis

Nature **574**, 505–510(2019) | Cite this article

661k Accesses | **26** Citations | **6016** Altmetric | Metrics

Abstract

The promise of quantum computers is that certain computational tasks might be executed exponentially faster on a quantum processor than on a classical processor[1]. A fundamental challenge is to build a high-fidelity processor capable of running quantum algorithms in an exponentially large computational space. Here we report the use of a processor with programmable superconducting qubits[2,3,4,5,6,7] to create quantum states on 53 qubits, corresponding to a computational state-space of dimension 2^{53} (about 10^{16}). Measurements from repeated experiments sample the resulting probability distribution, which we verify using classical simulations. Our Sycamore processor takes about 200 seconds to sample one instance of a quantum circuit a million times—our benchmarks currently indicate that the equivalent task for a state-of-the-art classical supercomputer would take approximately 10,000 years. This dramatic increase in speed compared to all known classical algorithms is an experimental realization of quantum supremacy[8,9,10,11,12,13,14] for this specific computational task, heralding a much-anticipated computing paradigm.

Science Contents News Careers Journals

SHARE

Google researchers in Santa Barbara, California, say their advance may lead to near-term applications of quantum computers. ISTOCK.COM/JHVEPHOTO

IBM casts doubt on Google's claims of quantum supremacy

By Adrian Cho | Oct. 23, 2019 , 5:40 AM

Pareto-Efficient Quantum Circuit Simulation Using Tensor Contraction Deferral*

Edwin Pednault[†1], John A. Gunnels[‡1], Giacomo Nannicini[‡1], Lior Horesh[1], Thomas Magerlein[2], Edgar Solomonik[3], Erik W. Draeger[4], Eric T. Holland[4], and Robert Wisnieff[1]

[1]IBM T.J. Watson Research Center, Yorktown Heights, NY
[2]Tufts University, Medford, MA
[3]Dept. of Computer Science, University of Illinois at Urbana-Champaign, Champaign, IL
[4]Lawrence Livermore National Laboratory, Livermore, CA

The Power of Tensor Networks

- Such embarrassment can happen to anyone, unless one appreciates the power of tensor networks...

nature

Explore content ⌄ About the journal ⌄ Publish with us ⌄

nature > articles > article

Article | Open Access | Published: 14 June 2023

Evidence for the utility of quantum computing before fault tolerance

Youngseok Kim, Andrew Eddins, Sajant Anand, Ken Xuan Wei, Ewout van den Berg, Sami Rosenblatt, Hasan Nayfeh, Yantao Wu, Michael Zaletel, Kristan Temme & Abhinav Kandala

Nature **618**, 500–505 (2023) | Cite this article

Abstract

Quantum computing promises to offer substantial speed-ups over its classical counterpart for certain problems. However, the greatest impediment to realizing its full potential is noise that is inherent to these systems. The widely accepted solution to this challenge is the implementation of fault-tolerant quantum circuits, which is out of reach for current processors. Here we report experiments on a noisy 127-qubit processor and demonstrate the measurement of accurate expectation values for circuit volumes at a scale beyond brute-force classical computation. We argue that this represents evidence for the utility of quantum computing in a pre-fault-tolerant era. These experimental results are enabled by advances in the coherence and calibration of a superconducting processor at this scale and the ability to characterize[1] and controllably manipulate noise across such a large device. We establish the accuracy of the measured expectation values by comparing them with the output of exactly verifiable circuits. In the regime of strong entanglement, the quantum computer provides correct results for which leading classical approximations such as pure-state-based 1D (matrix product states, MPS) and 2D (isometric tensor network states, isoTNS) tensor network methods[2,3] break down. These experiments demonstrate a foundational tool for the realization of near-term quantum applications[4,5].

Fast classical simulation of evidence for the utility of quantum computing before fault tolerance

Tomislav Begušić and Garnet Kin-Lic Chan*
Division of Chemistry and Chemical Engineering, California Institute of Technology, Pasadena, California 91125, USA
(Dated: June 29, 2023)

We show that a classical algorithm based on sparse Pauli dynamics can efficiently simulate quantum circuits studied in a recent experiment on 127 qubits of IBM's Eagle processor [*Nature* **618**, 500 (2023)]. Our classical simulations on a single core of a laptop are orders of magnitude faster than the reported walltime of the quantum simulations, as well as faster than the estimated quantum hardware runtime without classical processing, and are in good agreement with the zero-noise extrapolated experimental results.

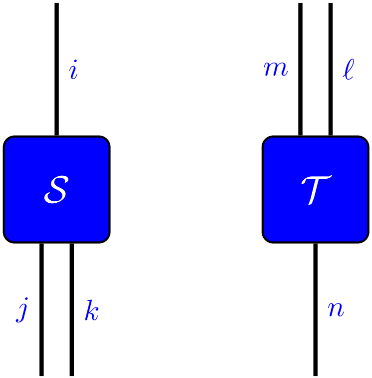
Efficient tensor network simulation of IBM's Eagle kicked Ising experiment

Joseph Tindall,[1] Matthew Fishman,[1] E. Miles Stoudenmire,[1] and Dries Sels[1,2]
[1] *Center for Computational Quantum Physics, Flatiron Institute, New York, New York 10010, USA*
[2] *Center for Quantum Phenomena, Department of Physics, New York University, 726 Broadway, New York, NY, 10003, USA*

We report an accurate, memory and time efficient classical simulation of a 127-qubit kicked Ising quantum system on the heavy-hexagon lattice. A simulation of this system on a quantum processor was recently performed using noise mitigation techniques to enhance accuracy (Nature volume 618, p. 500–505 (2023)). Here we show that, by adopting a tensor network approach that reflects the qubit connectivity of the device, we can perform a classical simulation that is significantly more accurate than the results obtained from the quantum device in the verifiable regime and comparable to the quantum simulation results for larger depths. The tensor network approach used will likely have broader applications for simulating the dynamics of quantum systems with tree-like correlations.

Tensor Network - Tensor Product

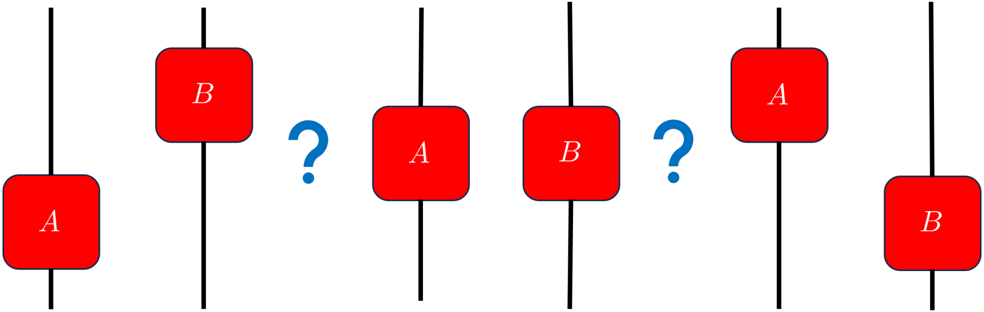
- Multiple disconnected tensors in the same diagram → multiplied by tensor product



$$\mathcal{S}^{i}_{jk} \otimes \mathcal{T}^{k\ell}_{m}$$

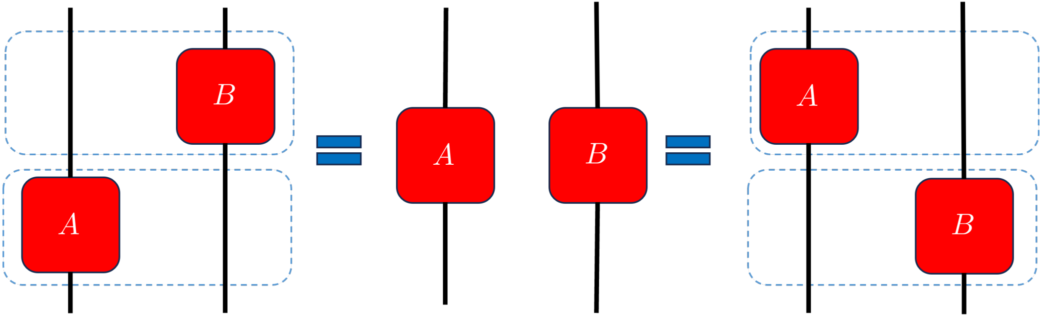
Tensor Network Invariants - Planner Deformation

- What is the difference between these networks ?



Tensor Network Invariants - Planar Deformation

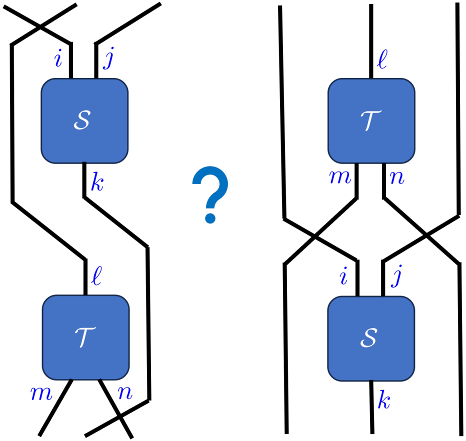
- These networks are **isomorphic**



- Tensors can freely roam past each other (planar deformation)

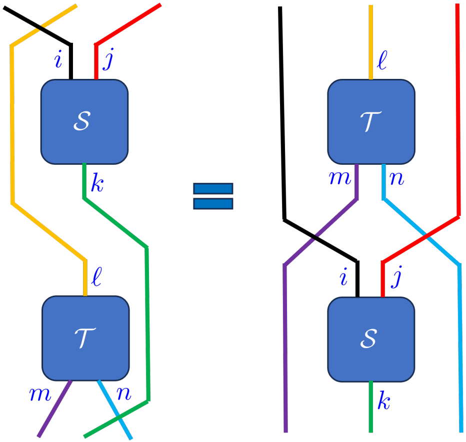
$$(\mathbb{1} \otimes B)(A \otimes \mathbb{1}) = A \otimes B = (A \otimes \mathbb{1})(\mathbb{1} \otimes B)$$

- Are these networks dissimilar ?



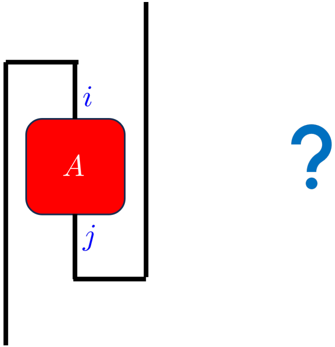
Tensor Network Invariants - Planar Deformation

- These networks are equivalent



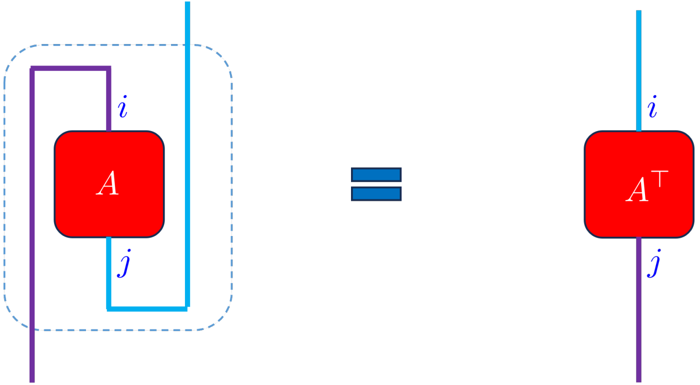
Tensor Network Relations

- What happens when we **swap edge directions** ?



Tensor Network Transposition

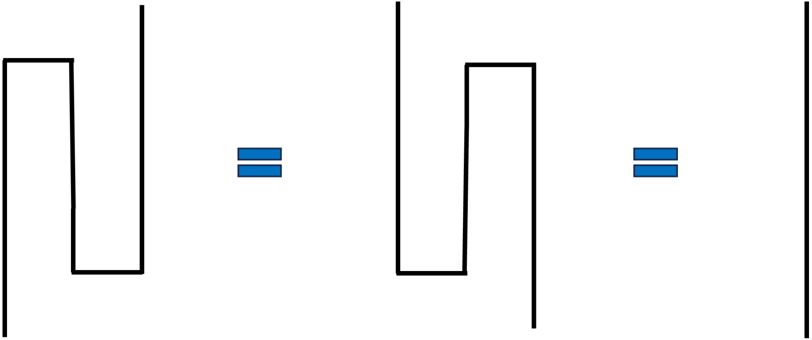
- Edge swapping is akin to index swap (transposition in matrices)



A_i^j $\qquad$ $(A^\top)_j^i$

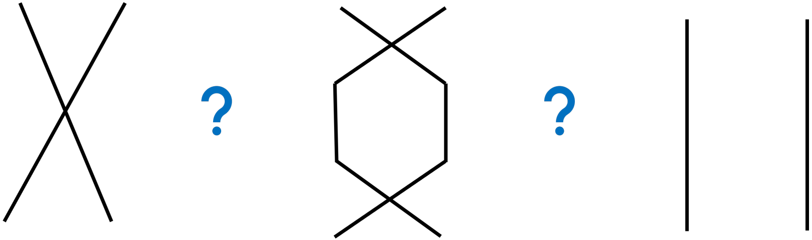
Tensor Network Invariants - Edge Detour

- Tensor networks are indifferent to edges "detours"



Tensor Network Swaps

- Are tensor networks indifferent to swaps ?

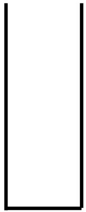


Tensor Networks - Penrose Duality

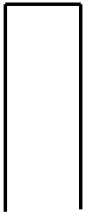
- **Penrose Duality** - bijection induced by bending wires

- Specific tensors (wire, cup, cap) play the role of **Kronecker's delta** and enable:
  - Tensor index **contraction** by diagrammatic connection
  - **Raising and lowering indices**
  - **Represent duality between maps, states and linear transformations**

$= \delta^i_j$



$= \delta^{ij}$



$= \delta_{ij}$

Anti-Symmetry

- A tensor is **fully anti-symmetric** if swapping any pair of indices changes its sign
- For example in 2D:

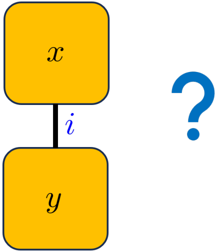
$$A_{ij} = -A_{ji}$$

- The ϵ_{ij} tensor is used to represent the **fully anti-symmetric Levi-Civita** symbol

$$\epsilon = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix}$$

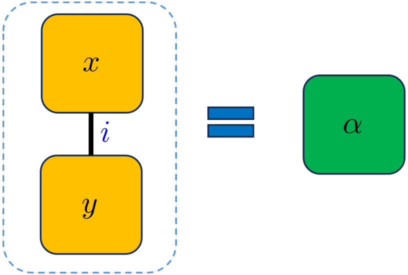
Tensor Network Contractions - Vector-Vector

- How do tensors interact ?



Tensor Network Contractions - Vector-Vector

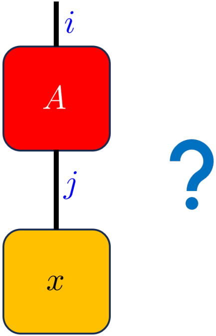
- Represents a dot-product between two vectors which entails a scalar
- **Edge contraction** implies summation over the joint index



$$\sum_i x_i y^i = \alpha$$

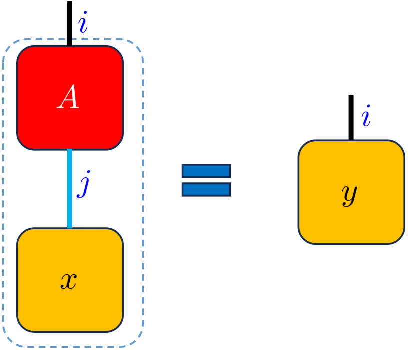
Tensor Network Contractions - Matrix-Vector

- How does a matrix and a vector contract ?



Tensor Network Contractions - Matrix-Vector

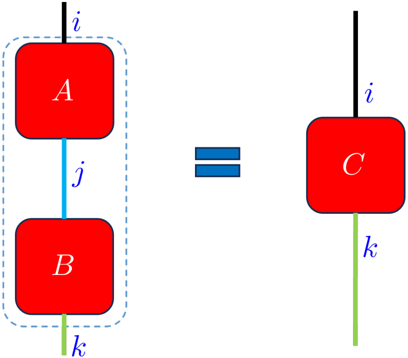
- Matrix-vector from a tensor network perspective is effectively a vector



$$\sum_j A^i_j x^j = y^i$$

Tensor Network Contractions - Matrix-Matrix

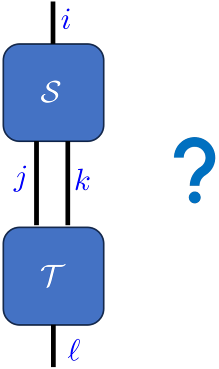
- Similarly, matrix-matrix contraction over a single edge entails a matrix (matrix-matrix product)



$$\sum_j A^i_j B^j_k = C^i_k$$

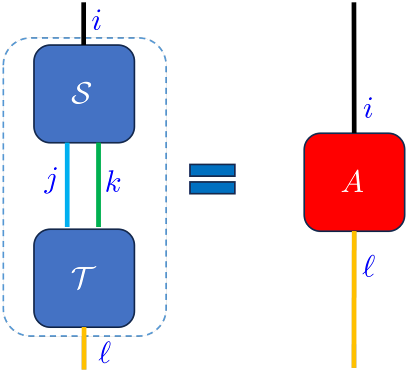
Tensor Network Contractions - Tensor-Tensor

- How do tensors interact with other tensors?



Tensor Network Contractions - Tensor-Tensor

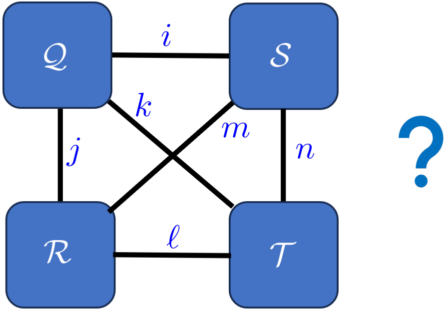
- Two 3^{rd} degree tensors contracted by 2 indices form a matrix



$$\sum_{jk} \mathcal{S}^i_{jk} \mathcal{T}^{jk}_{\ell} = A^i_{\ell}$$

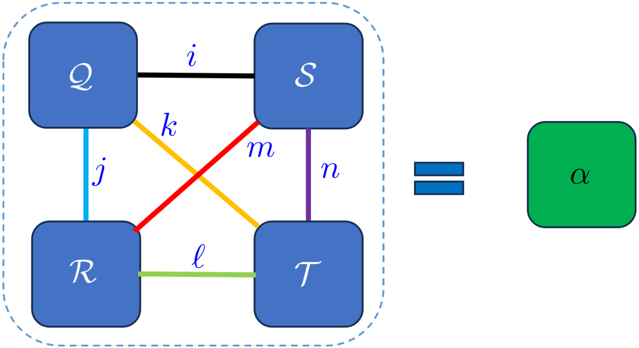
Tensor Network Contractions - Tensor-Tensor

- What would such contraction yield ?



Tensor Network Contractions - Tensor-Tensor

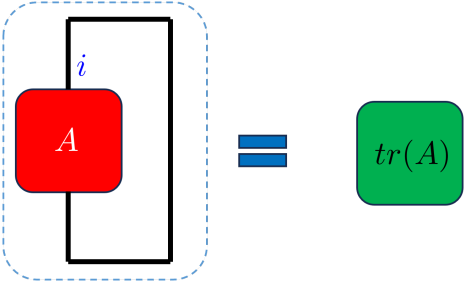
- Four 3^{rd} order tensors, where all edges contracted, entails a scalar



$$\sum_{ijk\ell mn} \mathcal{Q}^{i}_{jk}\mathcal{R}^{j}_{\ell m}\mathcal{S}^{mn}_{i}\mathcal{T}^{k\ell}_{n} = \alpha$$

Tensor Network Contractions - Trace

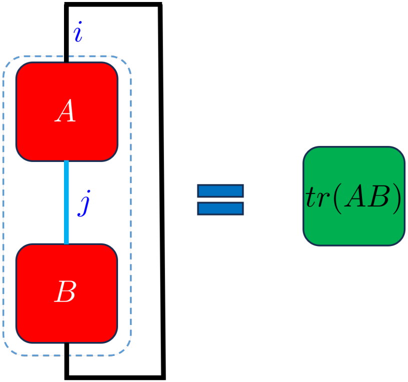
- What contraction of a tensor to **itself** means ?



$$\sum_i A_i^i = tr(A)$$

Tensor Network Contractions - Trace

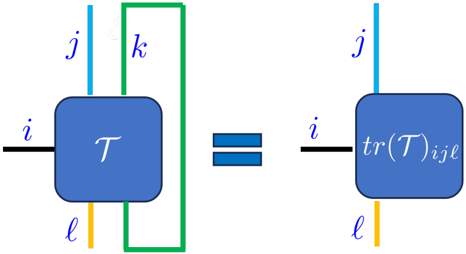
- What contraction of a matrix product to **itself** means ?

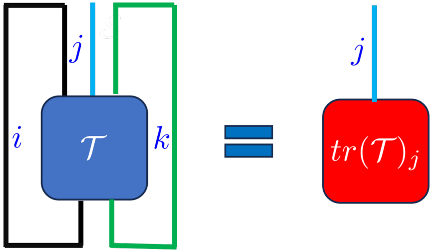


$$\sum_{ij} A^i_j B^j_i = tr(AB)$$

Tensor Network Contractions - Partial Trace

- What **partial contraction** of a tensor product to **itself** means ?



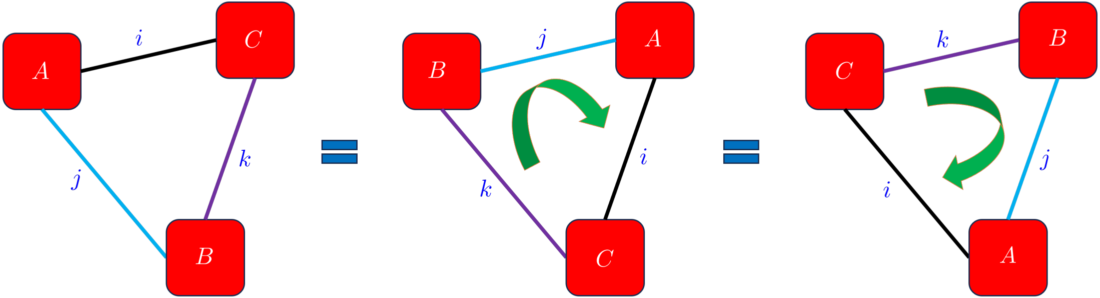


$$\sum_k \mathcal{T}_\ell^{ijk} = tr(\mathcal{T})_\ell^{ij}$$

$$\sum_{ik} \mathcal{T}^{ijk} = tr(\mathcal{T})^j$$

Tensor Network Contractions - Trace Cyclicity

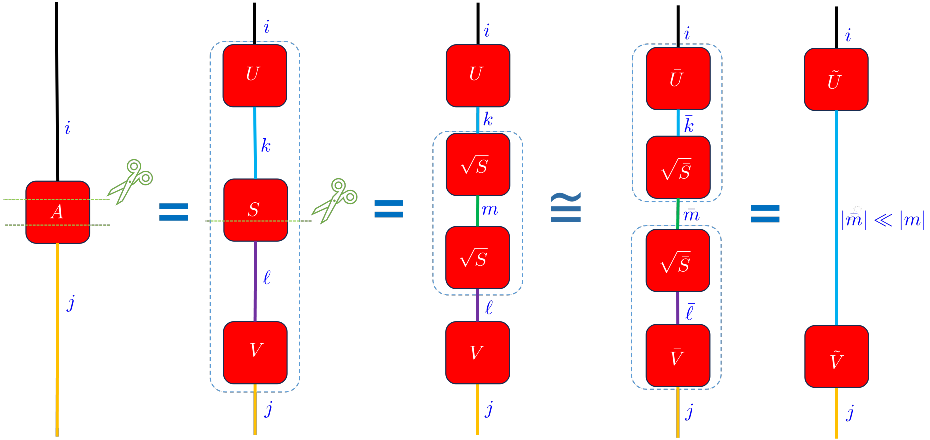
- How can we prove **trace cyclicity** ?
- Trivially proven with tensor networks due to **rotational invariance** of the network (**graph isomorphism**)



$tr(ABC)$ $tr(BCA)$ $tr(CAB)$

Tensor Networks - Splits and Low-Rank Approximation

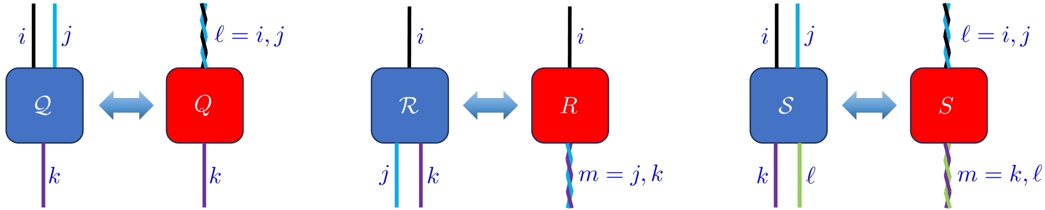
- How is it related to **sketching** and **low rank approximations** and **tensor algebra**?
- **Split** - inverse form of tensor contraction



Tensor Networks - Splits and Low-Rank Approximation

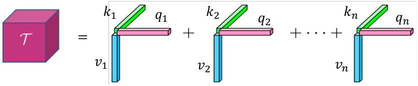
- Can we **always split** ?
  - Can always compute SVD on **matrices**

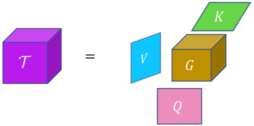
- How do we extend this to **tensors**?
  - **Vectorize** and then employ **matrix SVD**

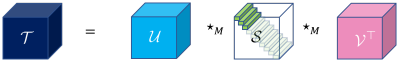


Tensor Networks - Splits and Low-Rank Approximation

- Can we **always split** ?
  - Can always compute SVD on **matrices**
- How do we extend this to **tensors**?
  - **Vectorize** and then employ **matrix SVD**
  - **Native** tensor decompositions ...







References

- Applications of negative dimensional tensors, Penrose, R., Combinatorial mathematics and its applications 1, 221-244 (1971)
- Bridgeman J.C., Chubb, C.T., Hand-waving and interpretive dance: an introductory course on tensor networks, Journal of Physics A: Mathematical and Theoretical 50, 223001 (2017), arxiv:1603.03039
- Biamonte, J., Bergholm, V., Tensor networks in a nutshell. (2017), arXiv:1708.00006
- Stoudenmire, E.M., Learning relevant features of data with multi-scale tensor networks, Quantum Science and Technology (2018): 034003