Tensor Networks

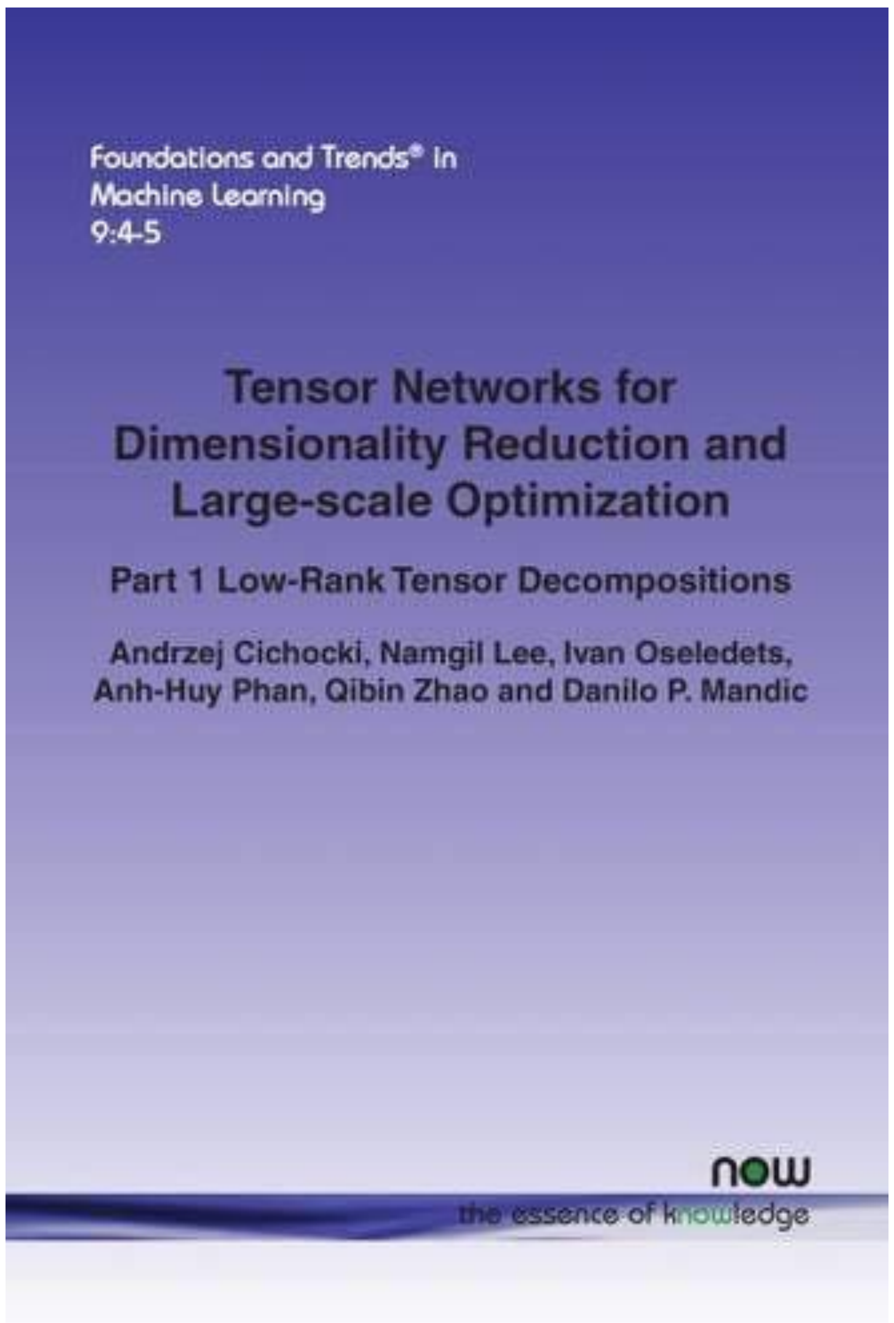
Qibin Zhao
Tensor Learning Unit
RIKEN AIP

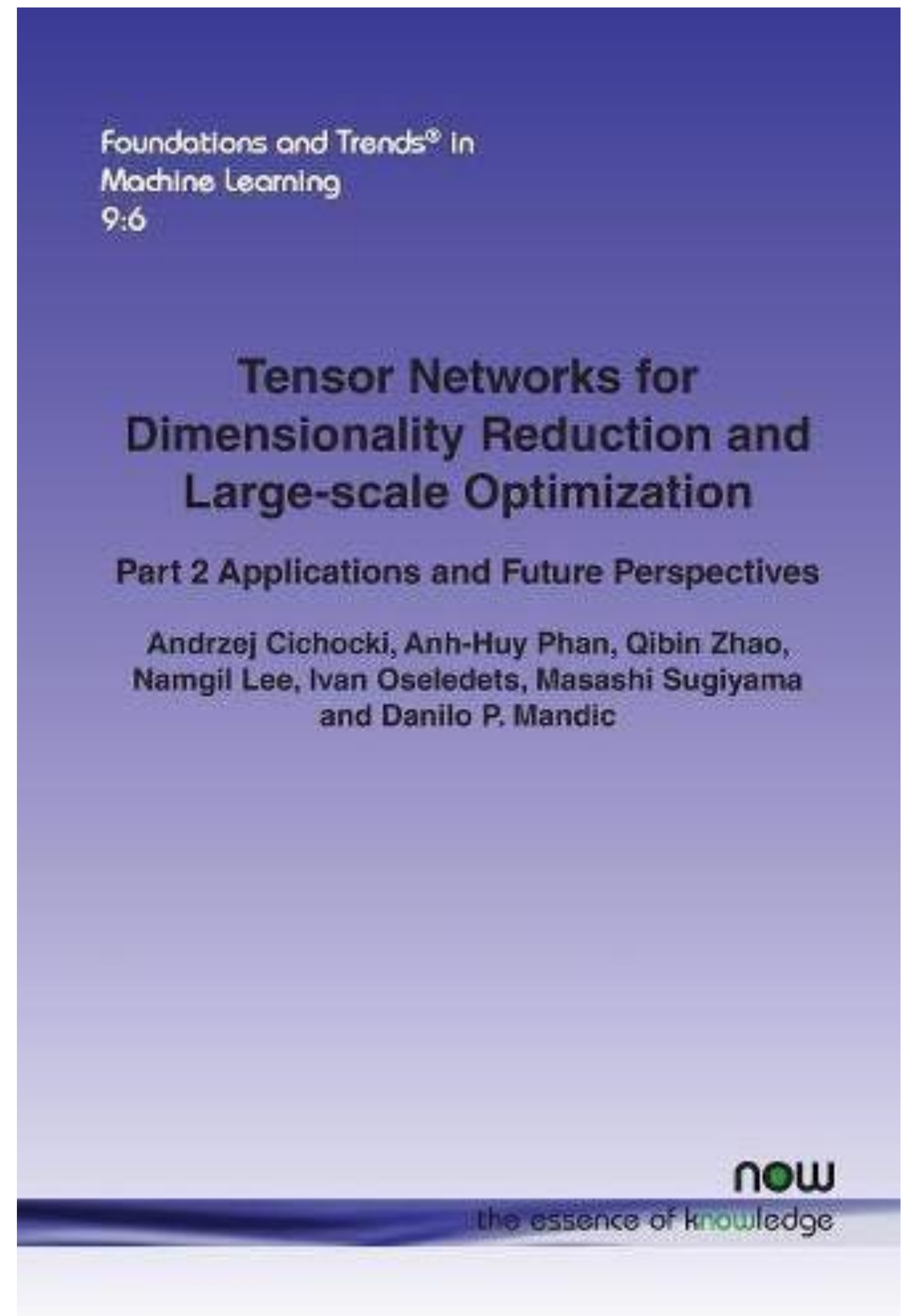
2018-6-2 @ Waseda University

Monographs

Tensor networks for dimensionality reduction and large optimization

Andrzej Cichocki, Namgil Lee, Ivan Oseledets, Anh-Huy Phan, **Qibin Zhao** and Danilo P.Mandic





Outline

- Why tensor network
- Tensor network diagrams
- Tensor networks and decompositions
- TT decomposition: graph interpretation and algorithm

Background

- Multidimensional data of exceedingly huge volume, variety and structural richness become ubiquitous across disciplines in engineering and data science
  - ✓ multimedia data like speech and video
  - ✓ remote sensing data
  - ✓ medical and biological data
- Standard machine learning methods and algorithms prohibitive to analysis of large-scale, multi-modal, multi-relational big data due to curse of dimensionality
- Machine learning and data analytic require a paradigm shift to efficiently process massive datasets within tolerable time
- Tensor networks emerges as very useful tools for dimensionality reduction and large-scale optimization problems

Curse of Dimensionality

- Curse of dimensionality (COD) an exponentially increasing of number of parameters required to describe a system or an extremely large number of degrees of freedom
- For tensor, COD means the number of elements I^N of an Nth-order tensor of size $I \times I \times \cdots \times I$ grows exponentially with tensor order N
- Tensor volumes become prohibitively huge if order is high, thus requiring enormous computational and storage resources

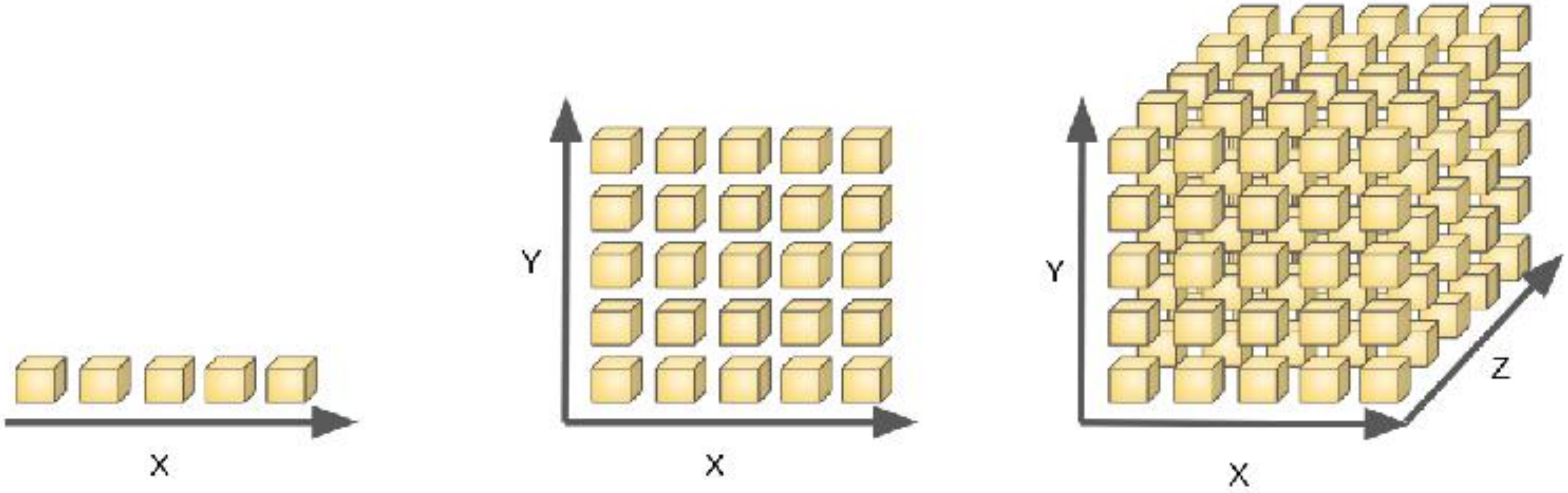


image credit Peter Gleeson

Challenges addressed by Tensor Networks

Tensor networks address two main challenges in big data analysis:

(i) Find a low-rank approximate representation for huge data tensor or a specific cost function while maintaining the desired accuracy of approximation, thus alleviating the curse of dimensionality

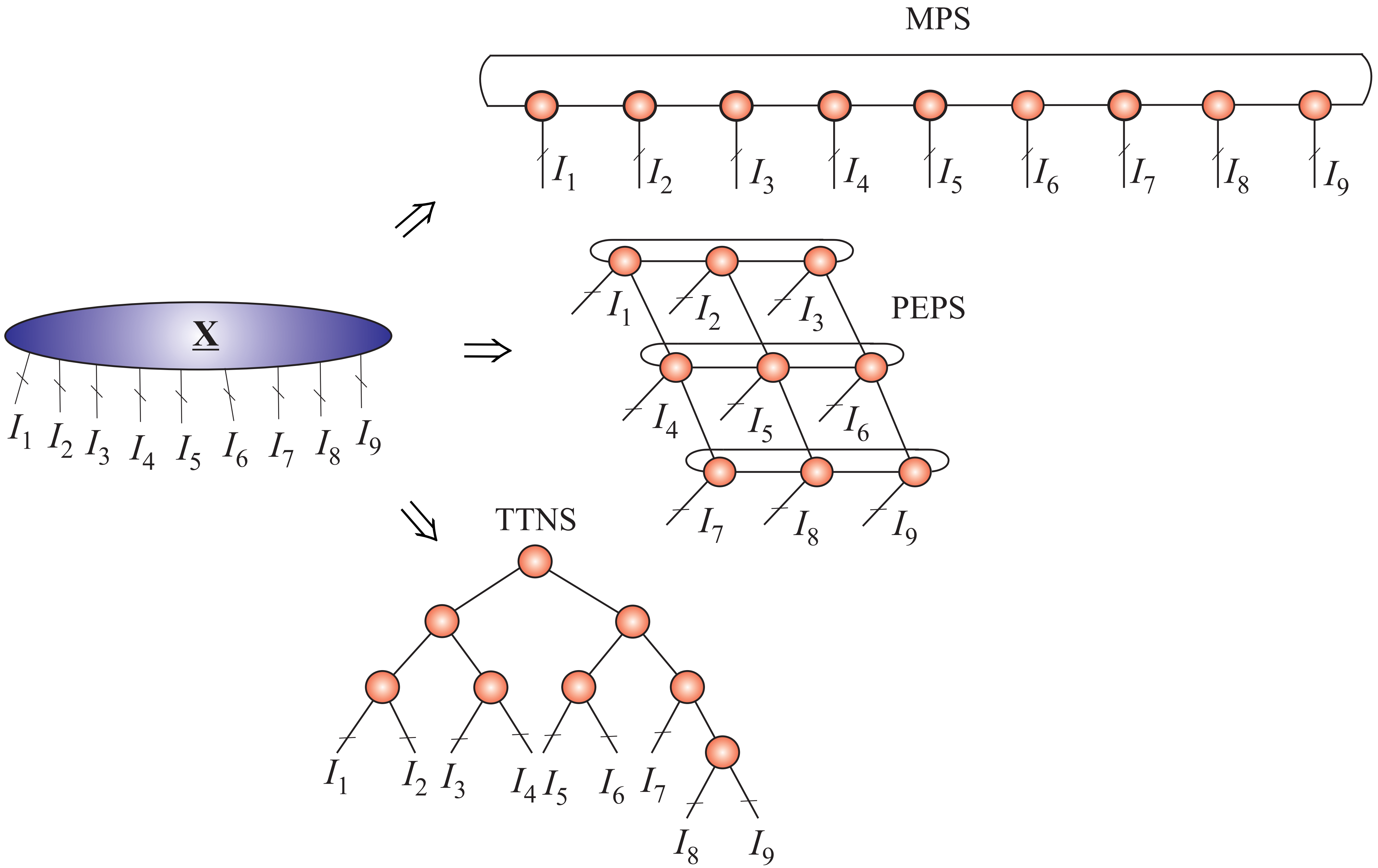
(ii) Extract physically meaningful latent variables from data in a sufficiently accurate and computationally afford way

What are Tensor Networks (TN)?

- Tensor decompositions (TD) decompose higher-order tensors into factor tensors and matrices
- Tensor networks (TN) decompose higher-order tensors into sparsely interconnected small-scale factor matrices or low-order core tensors
- TD and TN are treated in a united way by considering TD as a simple TN
- TN can be thought of as special graph structures representing high-order tensors via a set of sparsely interconnected, distributed low-order core tensors
- TN enjoys both enhanced interpretation and computational advantages, and allows for super-compression of big datasets
  - ✓ e.g. compute eigenvalues, eigenvectors of high-dimensional linear/nonlinear operators

TN Examples

TN decompose high-order tensors into a set of sparsely interconnected and distributed small-scale low-order core tensors



Advantages of TN

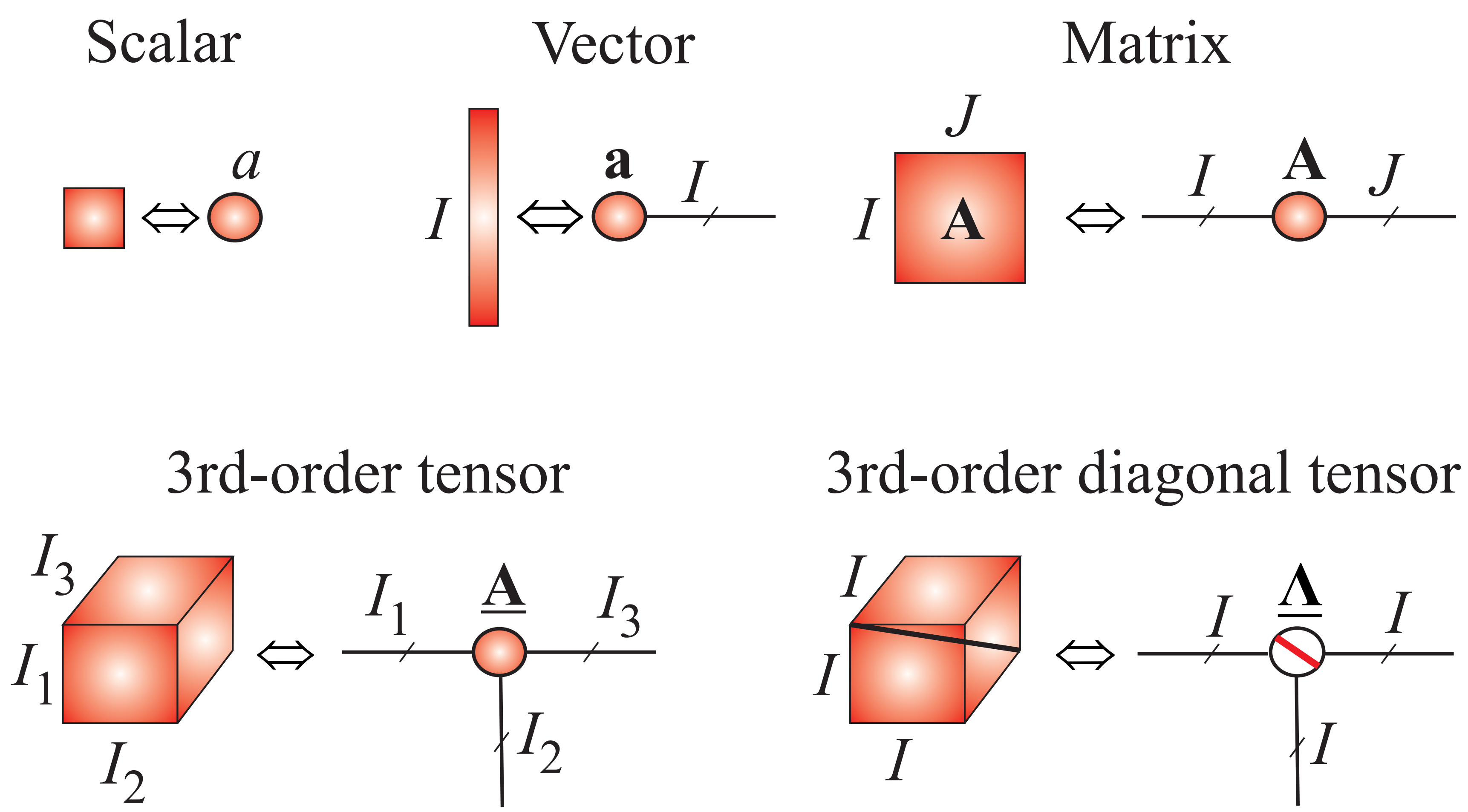
- Ability to perform all math operations in tractable formats
- Sparse and distributed formats of both the structurally rich data and complex optimization tasks
- Efficient compressed formats of large multidimensional data via tensorization and low-rank tensor decomposition into low-order factor core tensors
- Possibility to analyze linked blocks of large-scale tensors in order to separate correlated from uncorrelated components in observed raw data
- Graphical representations express math operations on tensors in an intuitive way, without the explicit use of complex math expressions

Outline

- Why tensor network
- Tensor network diagrams
- Tensor networks and decompositions
- TT decomposition: graph interpretation and algorithm

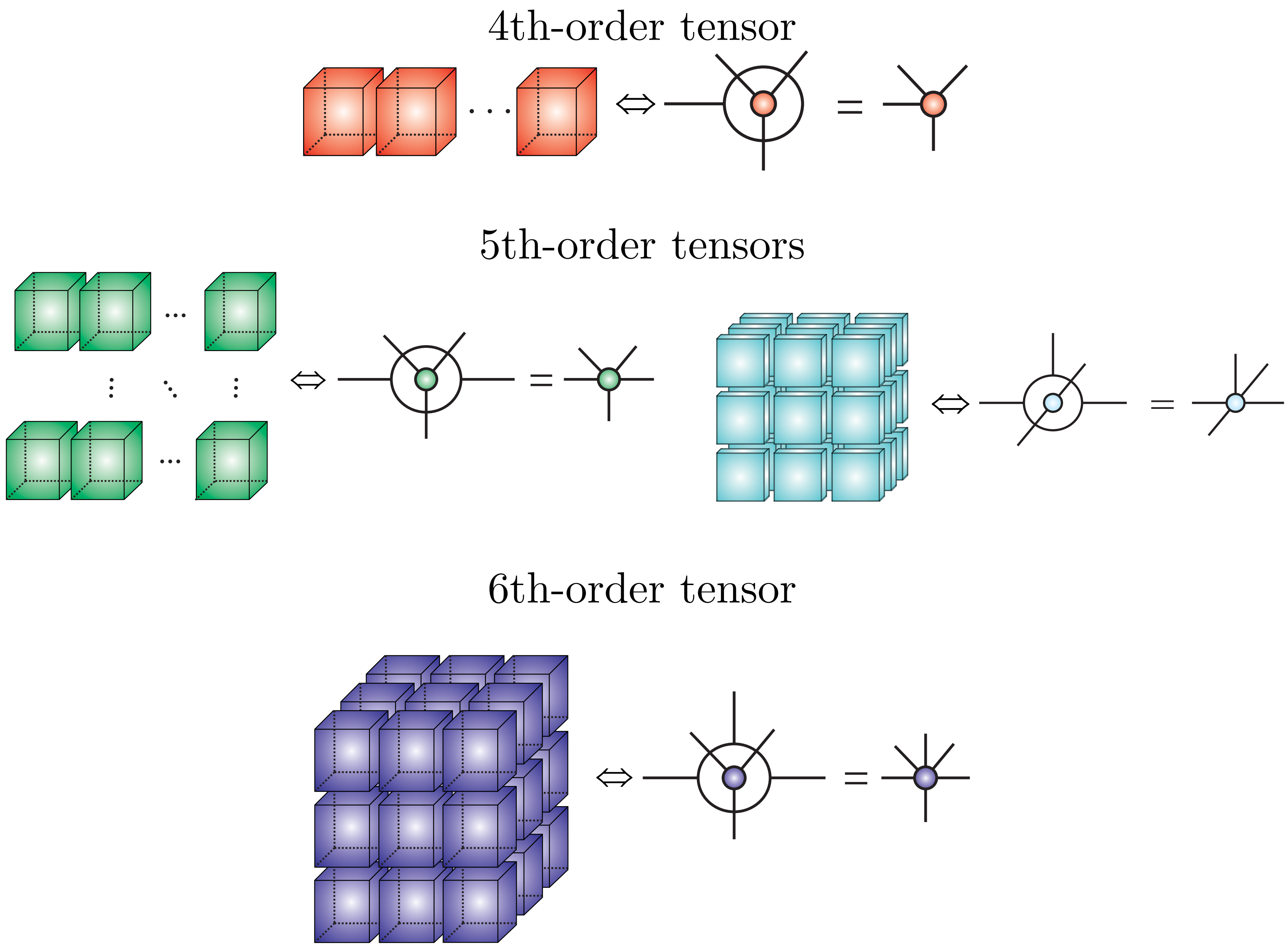
Tensors

Basic building blocks for TN diagrams



Block Tensors

TN diagrams for representing high-order block tensors, with each entry is an individual sub-tensor

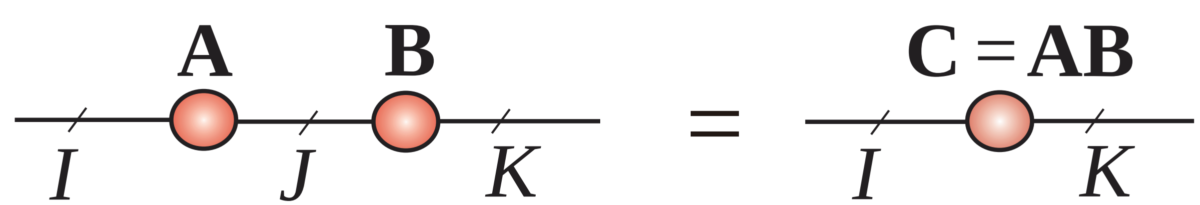


Basic Operations

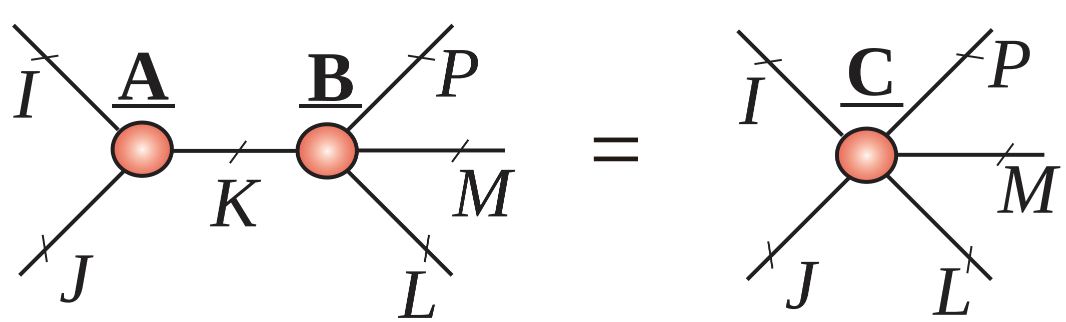
TN diagram for representing multi-linear operations

- Matrix-vector multiplication

- Matrix-matrix multiplication



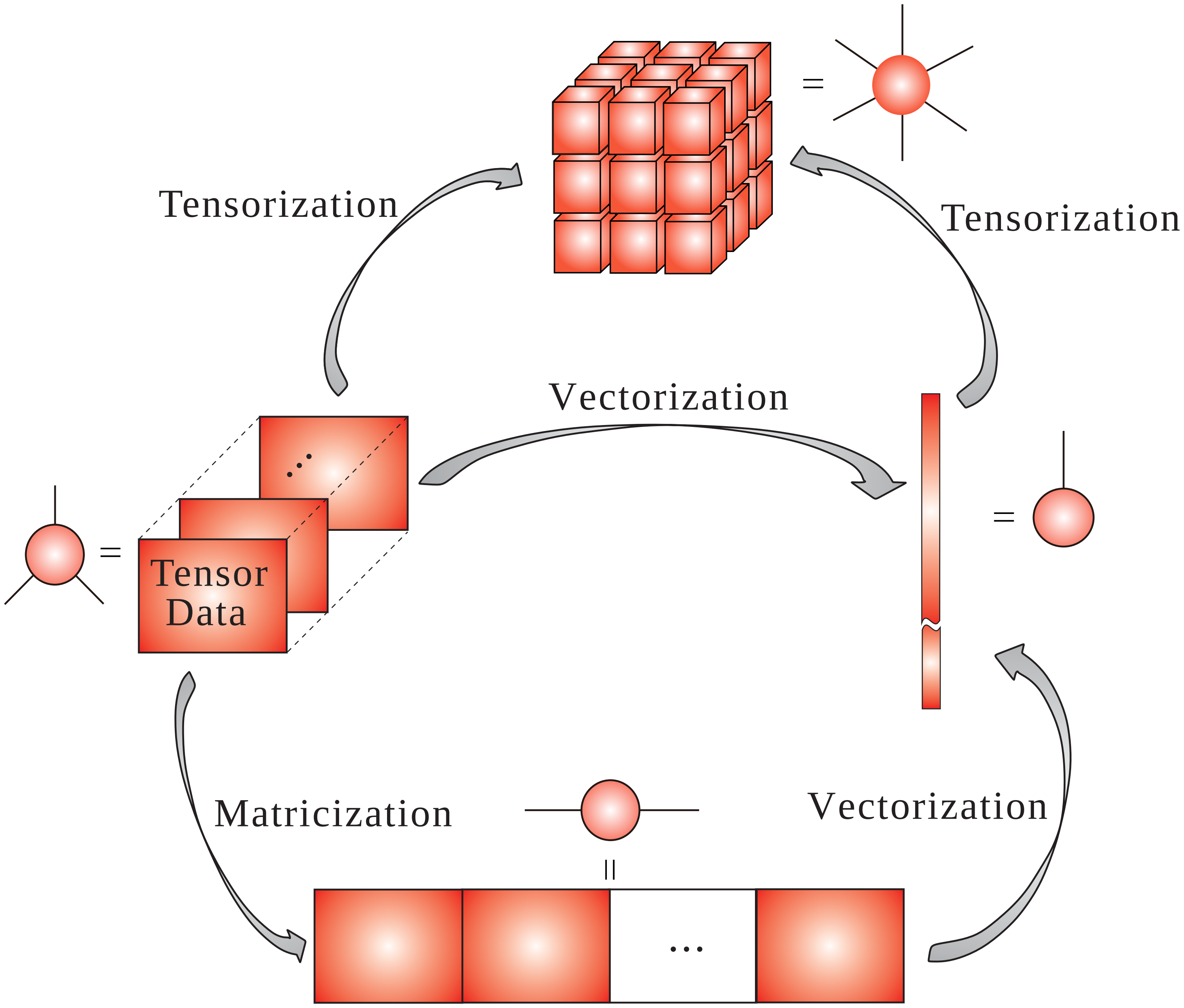
- Tensor contraction



$$\sum_{k=1}^{K} a_{i,j,k}\, b_{k,l,m,p} = c_{i,j,l,m,p}$$

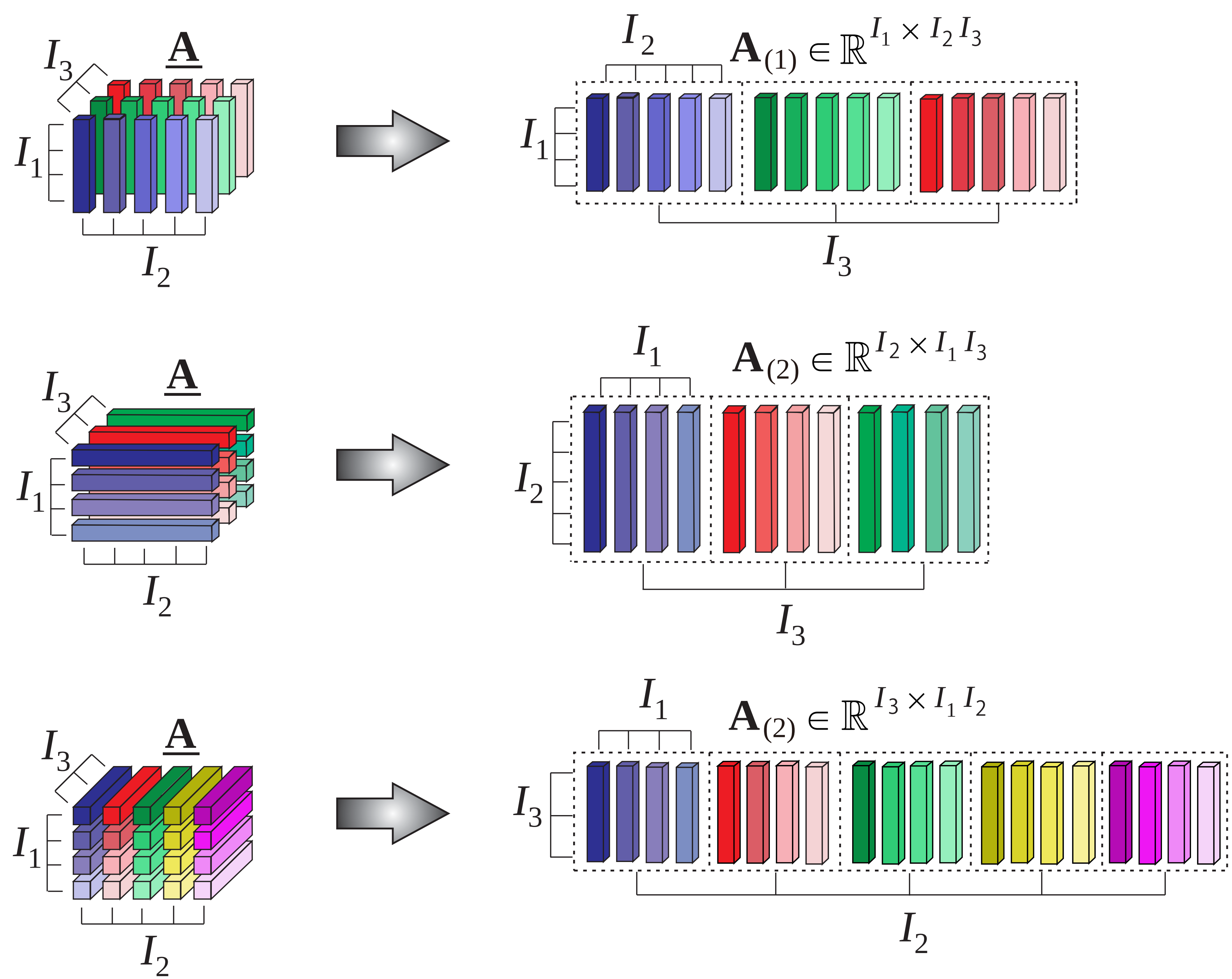
Tensor Reshaping Operations

Relationship between matricization, vectorization and tensorization



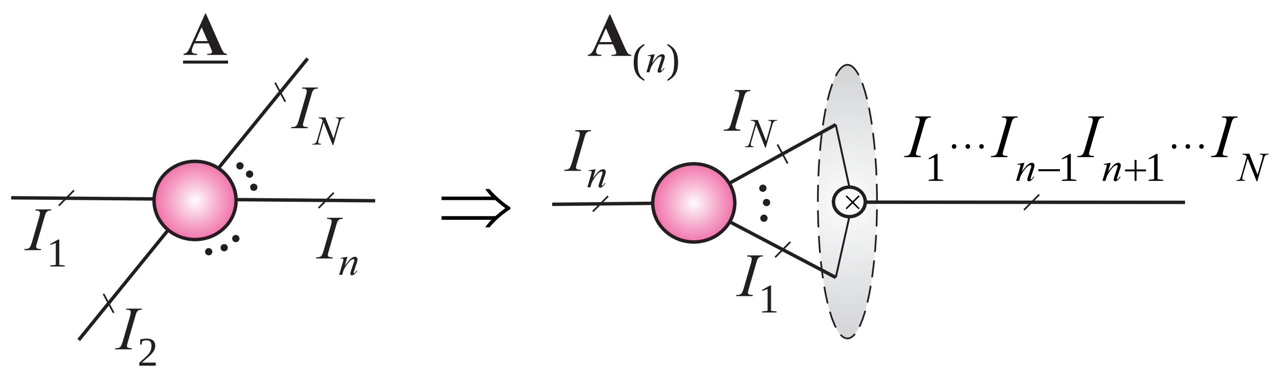
Matricization (Unfolding)

Illustration of mode-1, mode-2, mode-3 matricization of a 3rd-order tensor

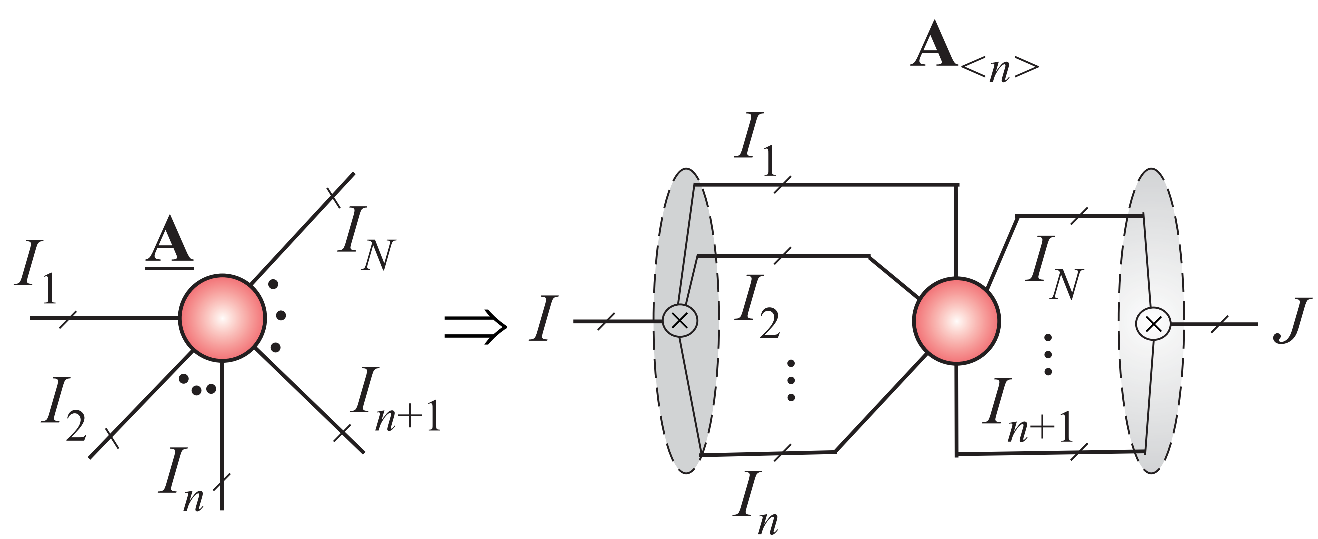


Matricization (Unfolding)

- TN Diagram of mode-n matricization of Nth-order tensor $\underline{\mathbf{A}} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ into a matrix $\mathbf{A}_{(n)} \in \mathbb{R}^{I_n \times I_1 \cdots I_{n-1} I_{n+1} \cdots I_N}$

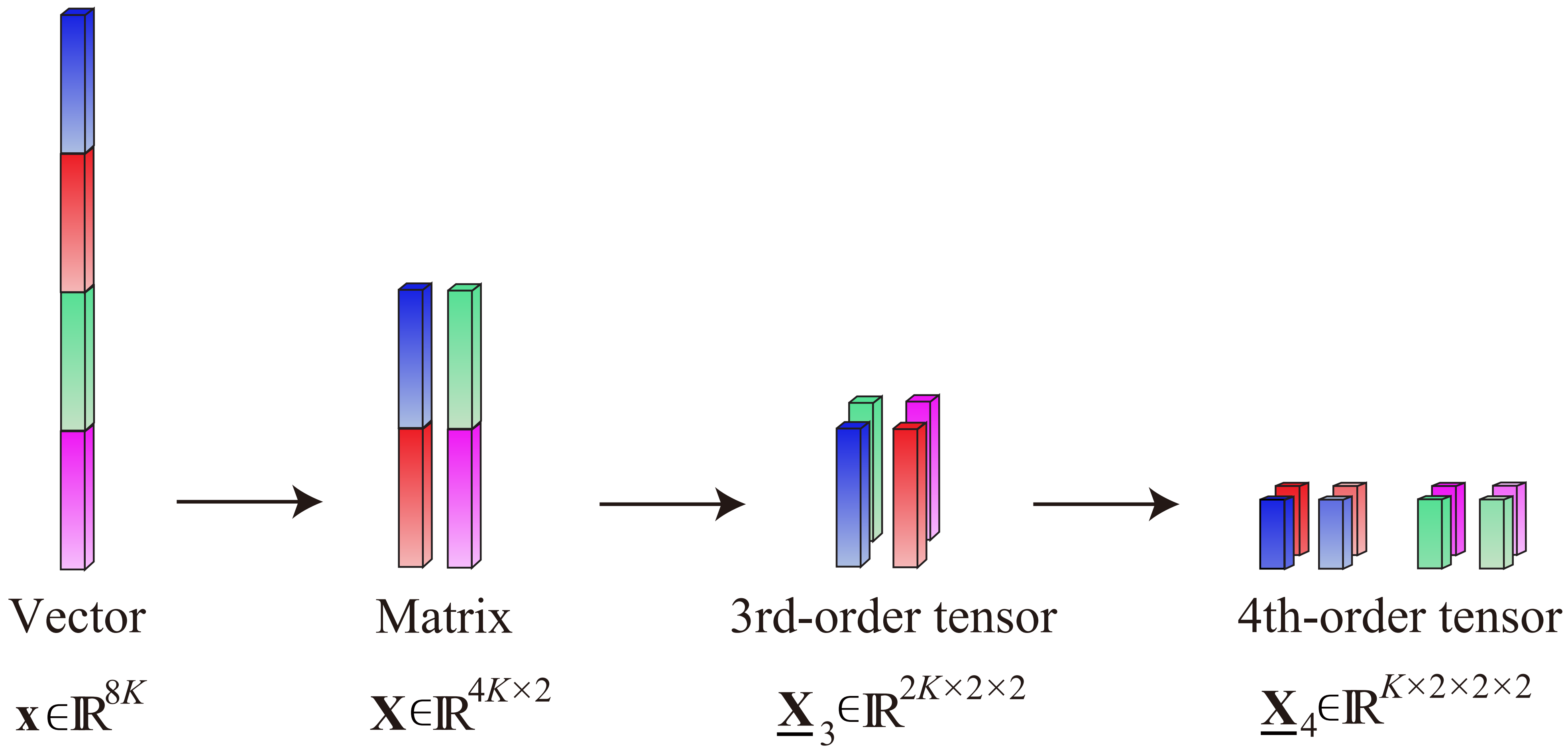


- TN Diagram of mode-{1,2,…,n} canonical matricization of a Nth-order tensor into a matrix $\mathbf{A}_{<n>} = \mathbf{A}_{(\overline{i_1 \cdots i_n}\,;\,\overline{i_{n+1} \cdots i_N})} \in \mathbb{R}^{I_1 I_2 \cdots I_n \times I_{n+1} \cdots I_N}$



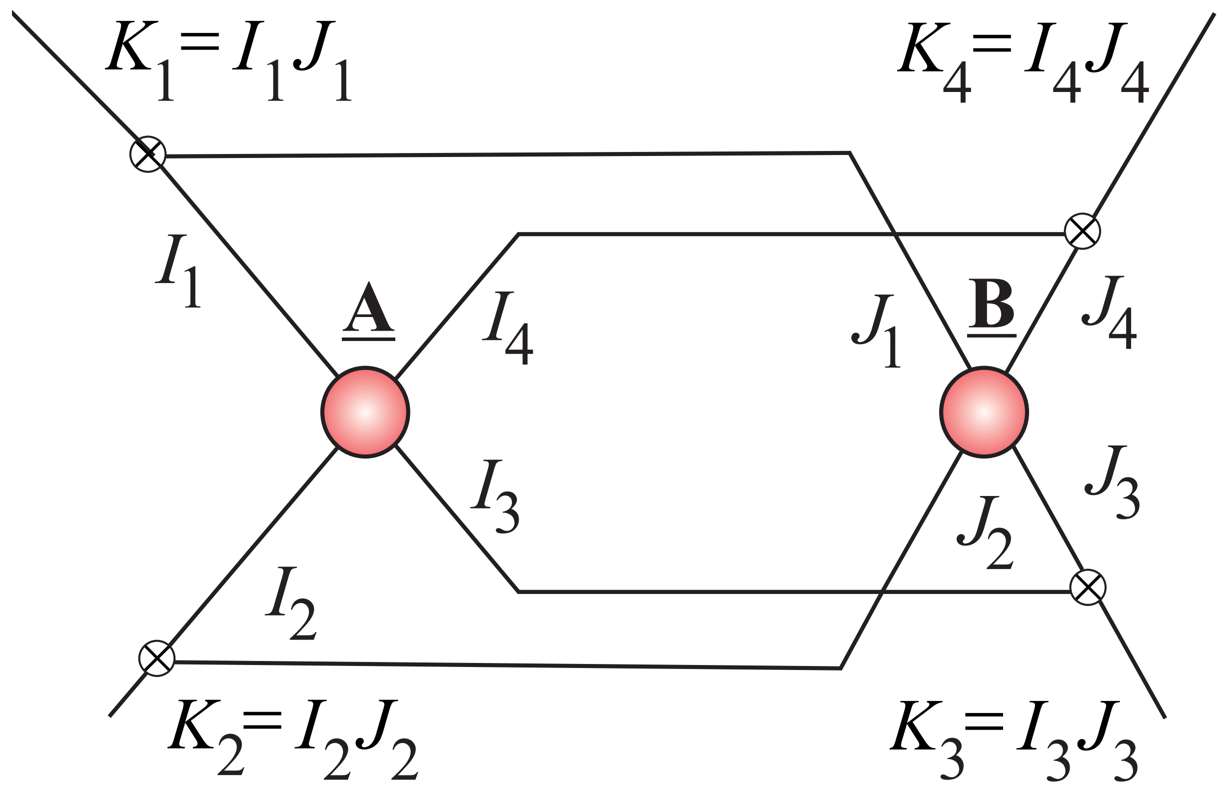
Tensorization

Tensorization of a vector or a matrix can be considered as a reverse process to the vectorization or matricization



Tensor Kronecker Product

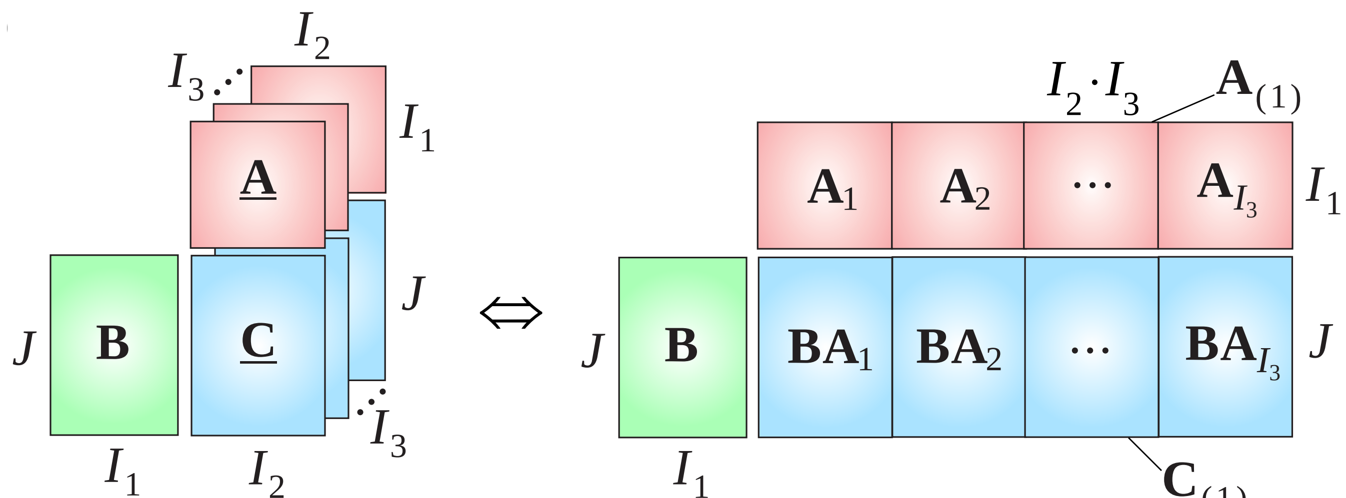
The kronecker product of two Nth-order tensors $\underline{\mathbf{A}} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ and $\underline{\mathbf{B}} \in \mathbb{R}^{J_1 \times J_2 \times \cdots \times J_N}$ yields tensor $\underline{\mathbf{C}} = \underline{\mathbf{A}} \otimes_L \underline{\mathbf{B}} \in \mathbb{R}^{I_1 J_1 \times \cdots \times I_N J_N}$ with entries $c_{\overline{i_1 j_1}, \ldots, \overline{i_N j_N}} = a_{i_1, \ldots, i_N} \, b_{j_1, \ldots, j_N}$



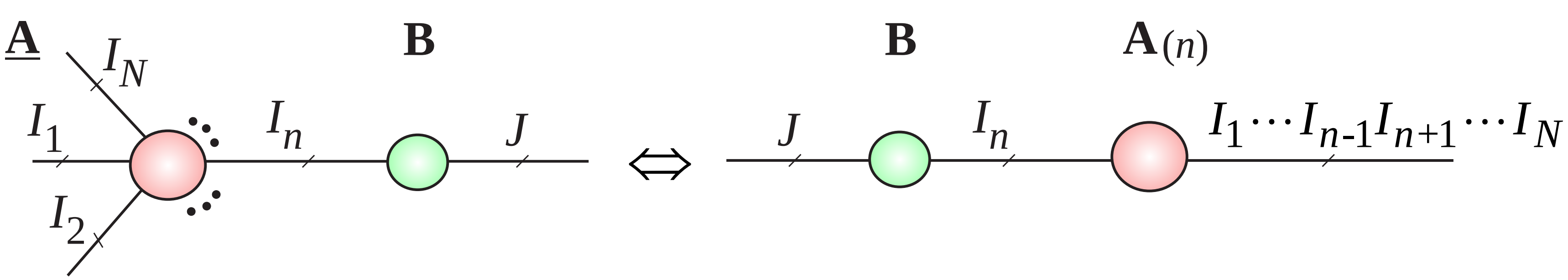
The mode-n product also called tensor-times-matrix (TTM) product of a tensor $\underline{\mathbf{A}} \in \mathbb{R}^{I_1 \times \cdots \times I_N}$ and matrix $\mathbf{B} \in \mathbb{R}^{J \times I_n}$ is defined as

$$\underline{\mathbf{C}} = \underline{\mathbf{A}} \times_n \mathbf{B} \in \mathbb{R}^{I_1 \times \cdots \times I_{n-1} \times J \times I_{n+1} \times \cdots \times I_N}$$

$$c_{i_1,i_2,\ldots,i_{n-1},j,i_{n+1},\ldots,i_N} = \sum_{i_n=1}^{I_n} a_{i_1,i_2,\ldots,i_N}\, b_{j,i_n}$$



$\underline{\mathbf{C}} = \underline{\mathbf{A}} \times_1 \mathbf{B}$ $\qquad$ $\mathbf{C}_{(1)} = \mathbf{B}\,\mathbf{A}_{(1)}$



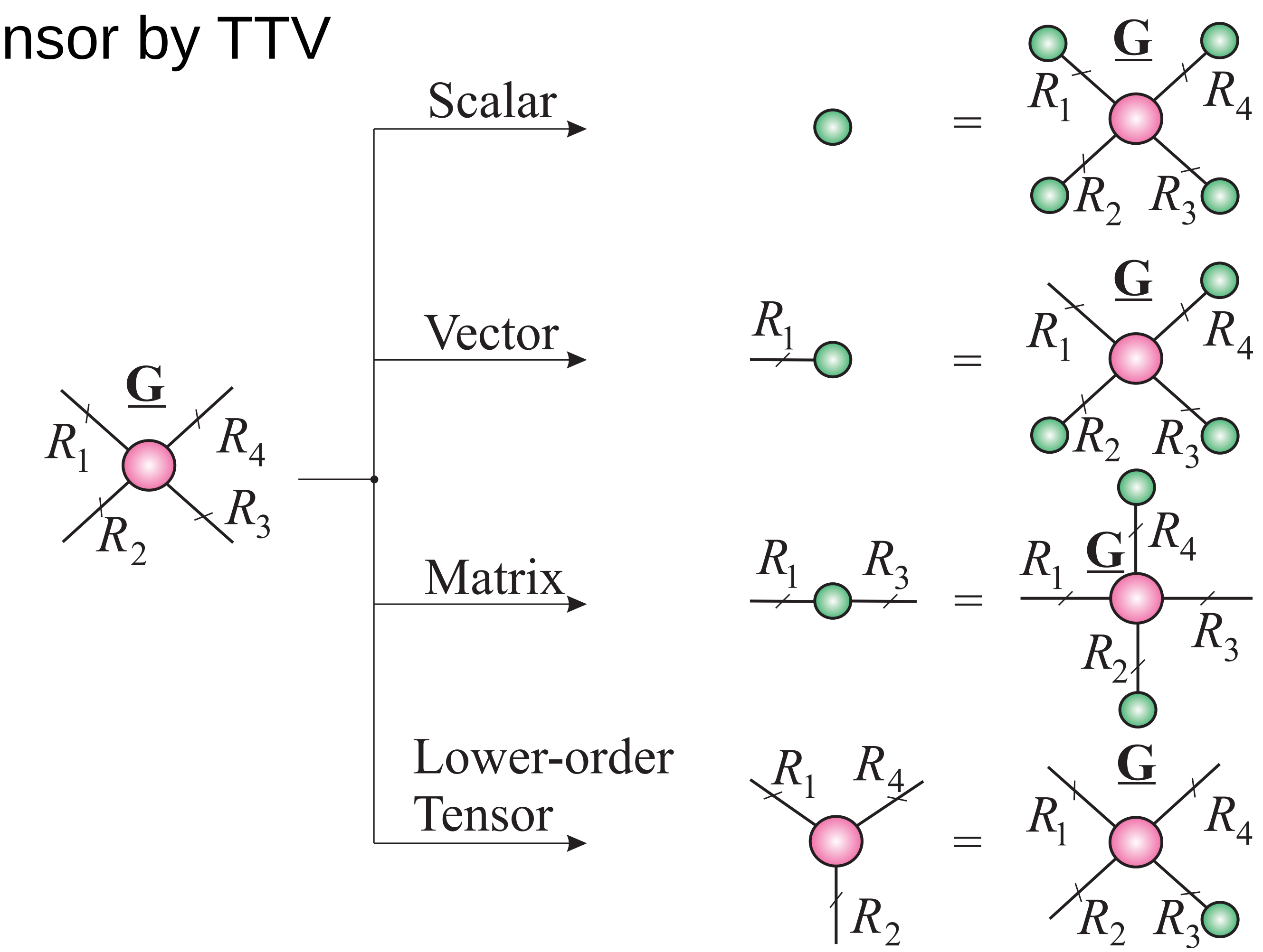
$\underline{\mathbf{C}} = \underline{\mathbf{A}} \times_n \mathbf{B}$ $\qquad$ $\mathbf{C}_{(n)} = \mathbf{B}\,\mathbf{A}_{(n)}$

Multilinear Product—TTV

The tensor-times-vector (TTV) product of a tensor $\underline{\mathbf{A}} \in \mathbb{R}^{I_1 \times \cdots \times I_N}$ and a vector $\mathbf{b} \in \mathbb{R}^{I_n}$ yields tensor $\underline{\mathbf{C}} = \underline{\mathbf{A}} \,\bar{\times}_n \mathbf{b} \in \mathbb{R}^{I_1 \times \cdots \times I_{n-1} \times I_{n+1} \times \cdots \times I_N}$ with entries

$$c_{i_1,\ldots,i_{n-1},i_{n+1},\ldots,i_N} = \sum_{i_n=1}^{I_n} a_{i_1,\ldots,i_{n-1},i_n,i_{n+1},\ldots,i_N} \, b_{i_n}$$

✓ an Illustration of compressing a 4th-order tensor into a scaler, vector, matrix or 3rd-order tensor by TTV

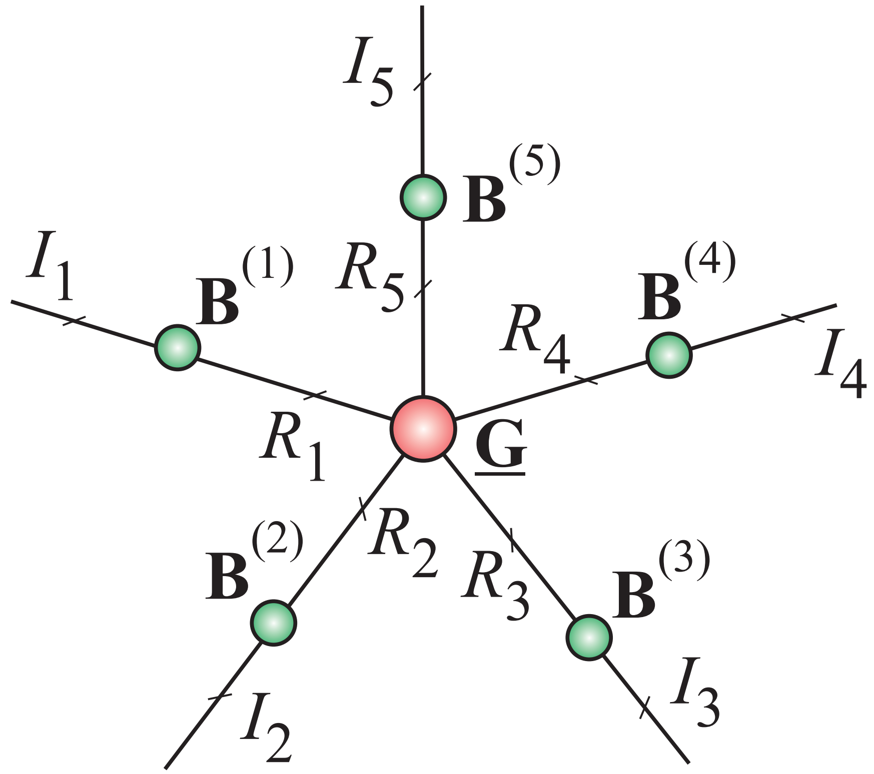


Full Multilinear Product—Tucker

The full multilinear (Tucker) product of a tensor $\underline{\mathbf{G}} \in \mathbb{R}^{R_1 \times R_2 \times \cdots \times R_N}$ and a set of factor matrices $\underline{\mathbf{B}}^{(n)} \in \mathbb{R}^{I_n \times R_n}$ perform multiplication in all the modes

$$\underline{\mathbf{C}} = \underline{\mathbf{G}} \times_1 \mathbf{B}^{(1)} \times_2 \mathbf{B}^{(2)} \cdots \times_N \mathbf{B}^{(N)}$$

✓ an Illustration of Tucker product a 5th-order tensor and five factor matrices



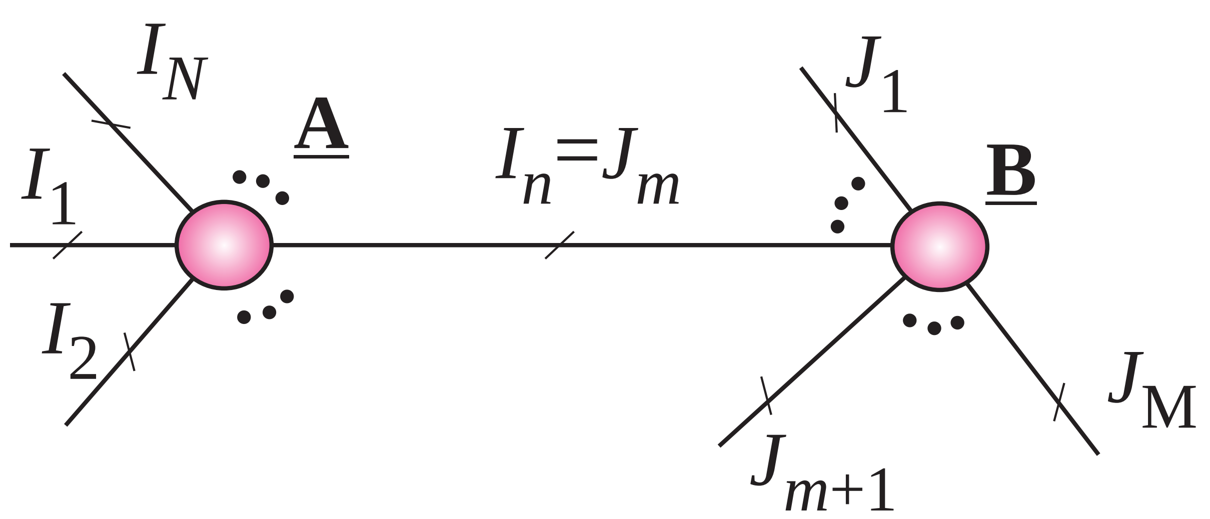
The tensor contraction of tensors $\underline{\mathbf{A}} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ and $\underline{\mathbf{B}} \in \mathbb{R}^{J_1 \times J_2 \times \cdots \times J_M}$

with common modes $I_n = J_m$, yields an (N+M-2)-order tensor as

$$\underline{\mathbf{C}} = \underline{\mathbf{A}} \times_n^m \underline{\mathbf{B}} \in \mathbb{R}^{I_1 \times \cdots \times I_{n-1} \times I_{n+1} \times \cdots \times I_N \times J_1 \times \cdots \times J_{m-1} \times J_{m+1} \times \cdots \times J_M}$$

with entires

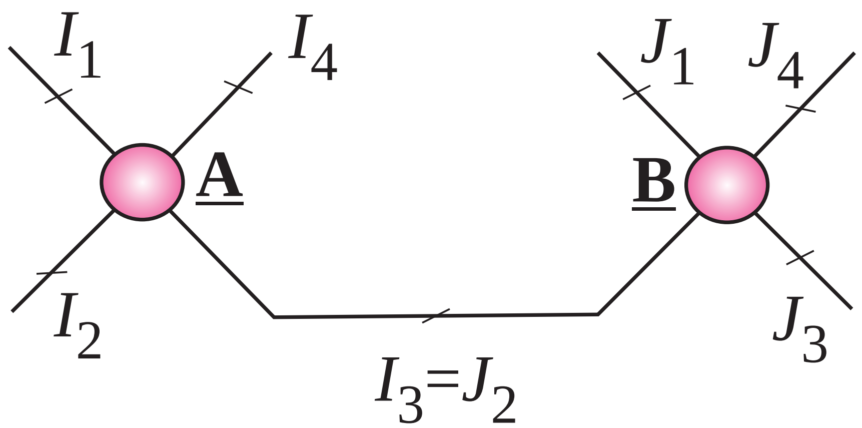
$$c_{i_1, \ldots, i_{n-1}, i_{n+1}, \ldots, i_N, j_1, \ldots, j_{m-1}, j_{m+1}, \ldots, j_M} =$$
$$= \sum_{i_n=1}^{I_n} a_{i_1, \ldots, i_{n-1}, i_n, i_{n+1}, \ldots, i_N} \; b_{j_1, \ldots, j_{m-1}, i_n, j_{m+1}, \ldots, j_M}$$



Tensor Contraction Examples

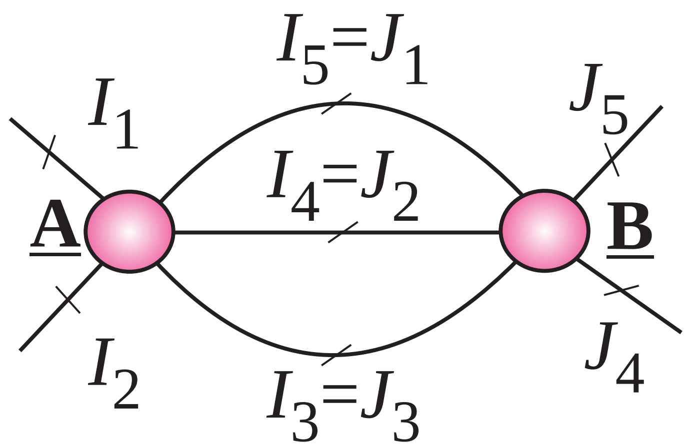
- Tensor contraction of two 4th-order tensors along mode-3 in $\underline{\mathbf{A}}$ and mode-2 in $\underline{\mathbf{B}}$ yield a 6th-order tensor

$$\underline{\mathbf{C}} = \underline{\mathbf{A}} \times_3^2 \underline{\mathbf{B}} \in \mathbb{R}^{I_1 \times I_2 \times I_4 \times J_1 \times J_3 \times J_4}$$



- Tensor contraction of two 5th-order tensors along modes 3,4,5 in $\underline{\mathbf{A}}$ and 1,2,3 in $\underline{\mathbf{B}}$ yield a 4th-order tensor

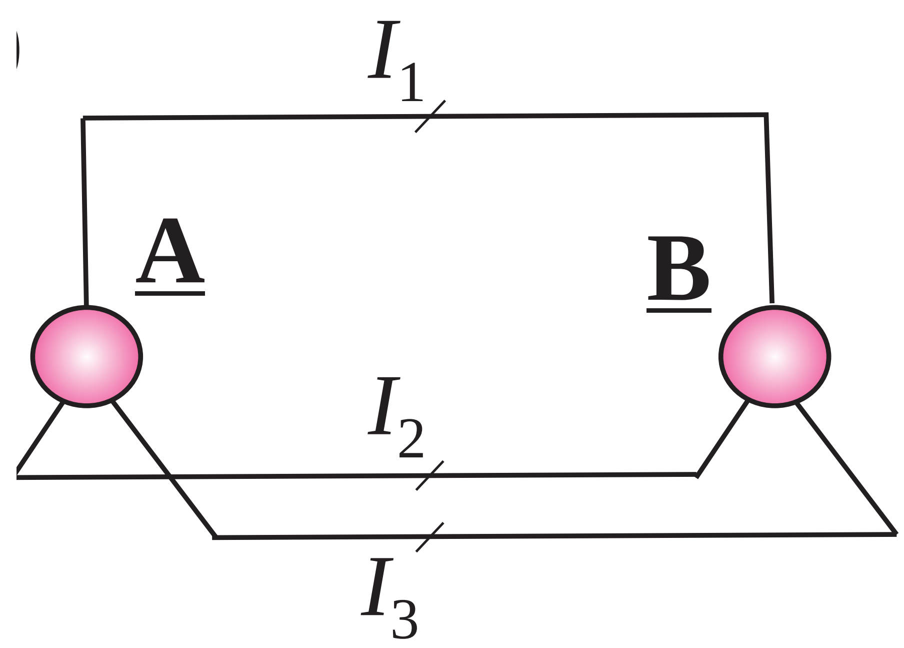
$$\underline{\mathbf{C}} = \underline{\mathbf{A}} \times_{5,4,3}^{1,2,3} \underline{\mathbf{B}} \in \mathbb{R}^{I_1 \times I_2 \times J_4 \times J_5}$$



Tensor Contraction Examples Cont

- Tensor contraction along all the modes (or Inner product) of two 3rd-order tensors yield a scaler

$$c = \langle \underline{\mathbf{A}}, \underline{\mathbf{B}} \rangle = \underline{\mathbf{A}} \times^{1,2,3}_{1,2,3} \underline{\mathbf{B}} = \underline{\mathbf{A}} \bar{\times} \underline{\mathbf{B}} = \sum_{i_1,i_2,i_3} a_{i_1,i_2,i_3} b_{i_1,i_2,i_3}$$

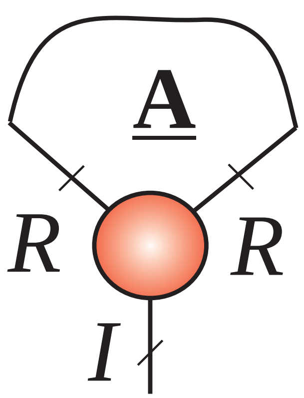


Multilinear Product—Tensor Trace

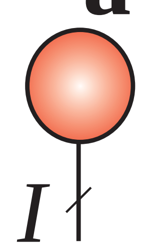
The tensor trace consider a tensor with partial self-contraction modes, where the outer indices represent physical modes, inner indices represent contraction modes. The tensor trace performs the summation of all inner indices of tensor

- ✓ e.g., a tensor $\underline{\mathbf{A}}$ of size $R \times I \times R$ has two inner indices: mode 1 and 3 of size R, and one outer index: mode 2 of size I, tensor trace yields a vector

$$\mathbf{a} = \mathrm{Tr}(\underline{\mathbf{A}}) = \sum_r \underline{\mathbf{A}}(r, :, r)$$



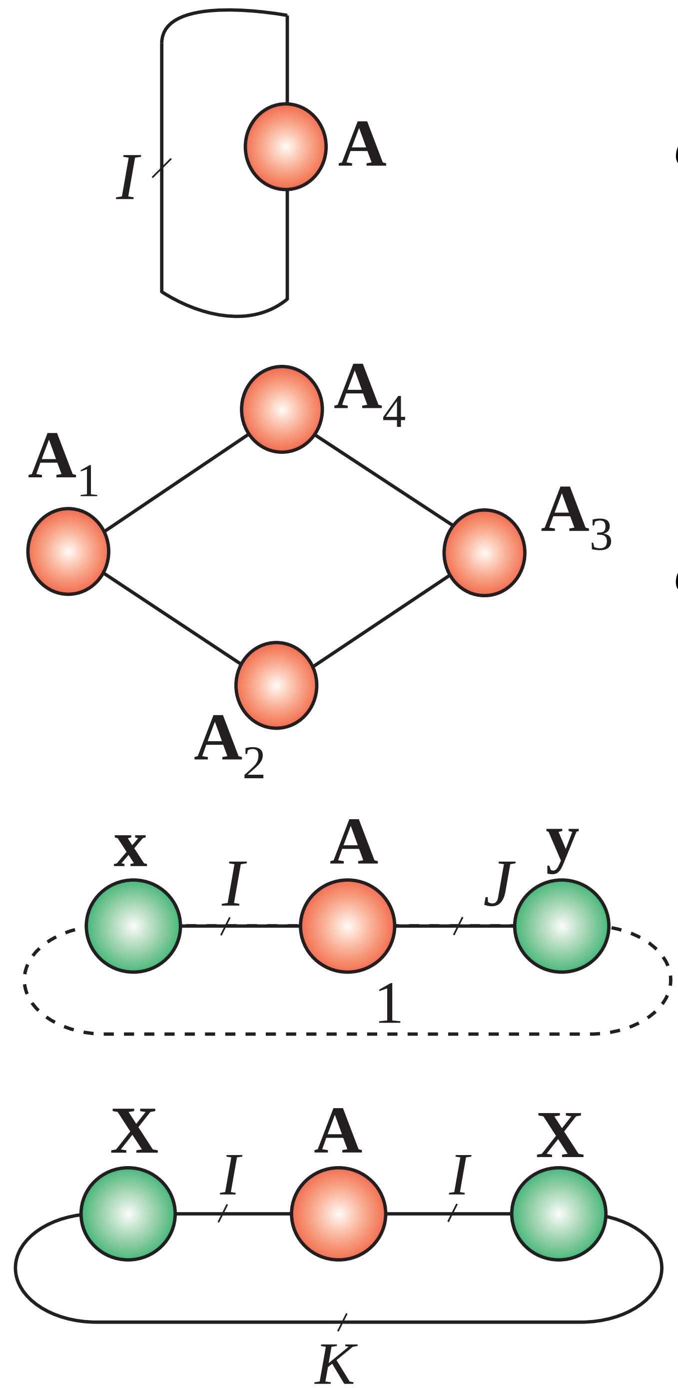
=



$$\mathbf{a} = [a_1, a_2, \ldots, a_I]^{\mathrm{T}}$$

$$a_i = \sum_r \underline{\mathbf{A}}(r, i, r)$$

- TN diagrams of tensor trace of matrices



$$c = \mathrm{tr}(\mathbf{A}) = \sum_i a_{ii}$$

$$c = \mathrm{tr}(\mathbf{A}_1\mathbf{A}_2\mathbf{A}_3\mathbf{A}_4)$$

$$\mathrm{tr}(\mathbf{A}\mathbf{y}\mathbf{x}^{\mathrm{T}}) = \mathbf{x}^{\mathrm{T}}\mathbf{A}\mathbf{y}$$

$$\mathrm{tr}(\mathbf{X}^{\mathrm{T}}\mathbf{A}\mathbf{X})$$

Transformation of TN structures

TN graphical representation has benefits to

- perform complex math operations on core tensors in an intuitive way, without resorting to math expressions
- modify, simplify and optimize the topology of TN, while keeping the original physical model intact
  - ✓ modify topology to tree structured TN like HT/TT can reduce computational complexity (through sequential contraction of cores) and enhance stability of algorithms
  - ✓ often advantageous to modify TN with circles to TN with tree structure by eliminating circles

Transformation of TN structures Cont

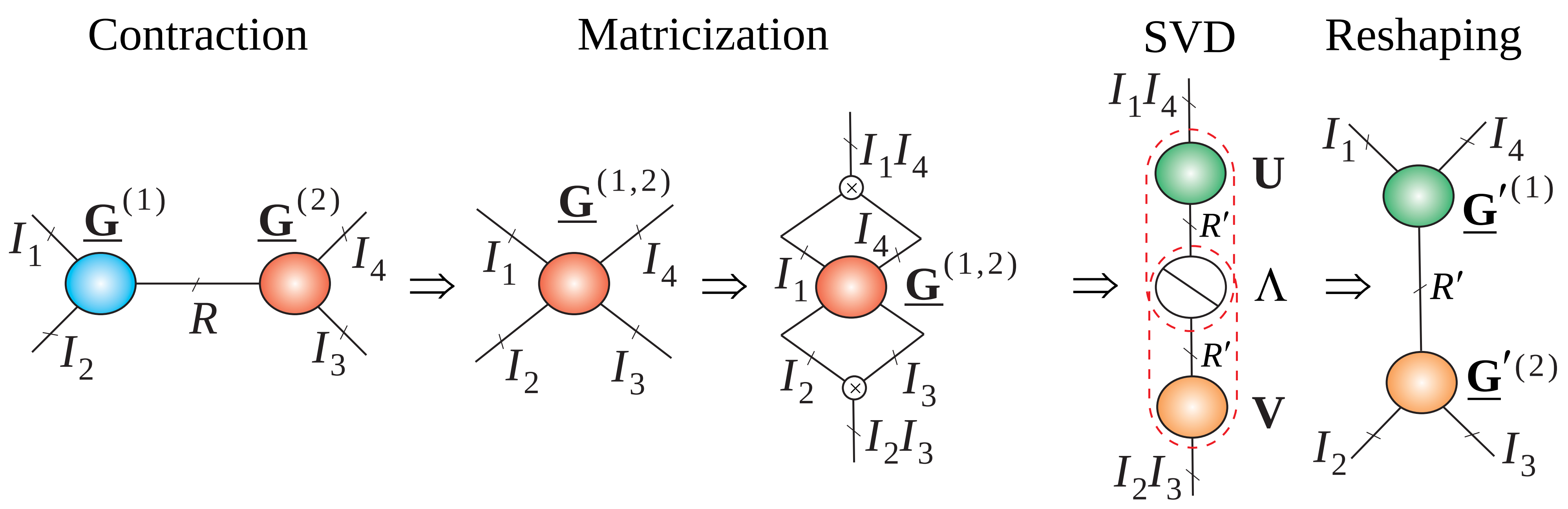
A general procedure of the basic transformation on TN structure:

i) perform sequential core tensors

ii) unfold these contracted tensors into matrices

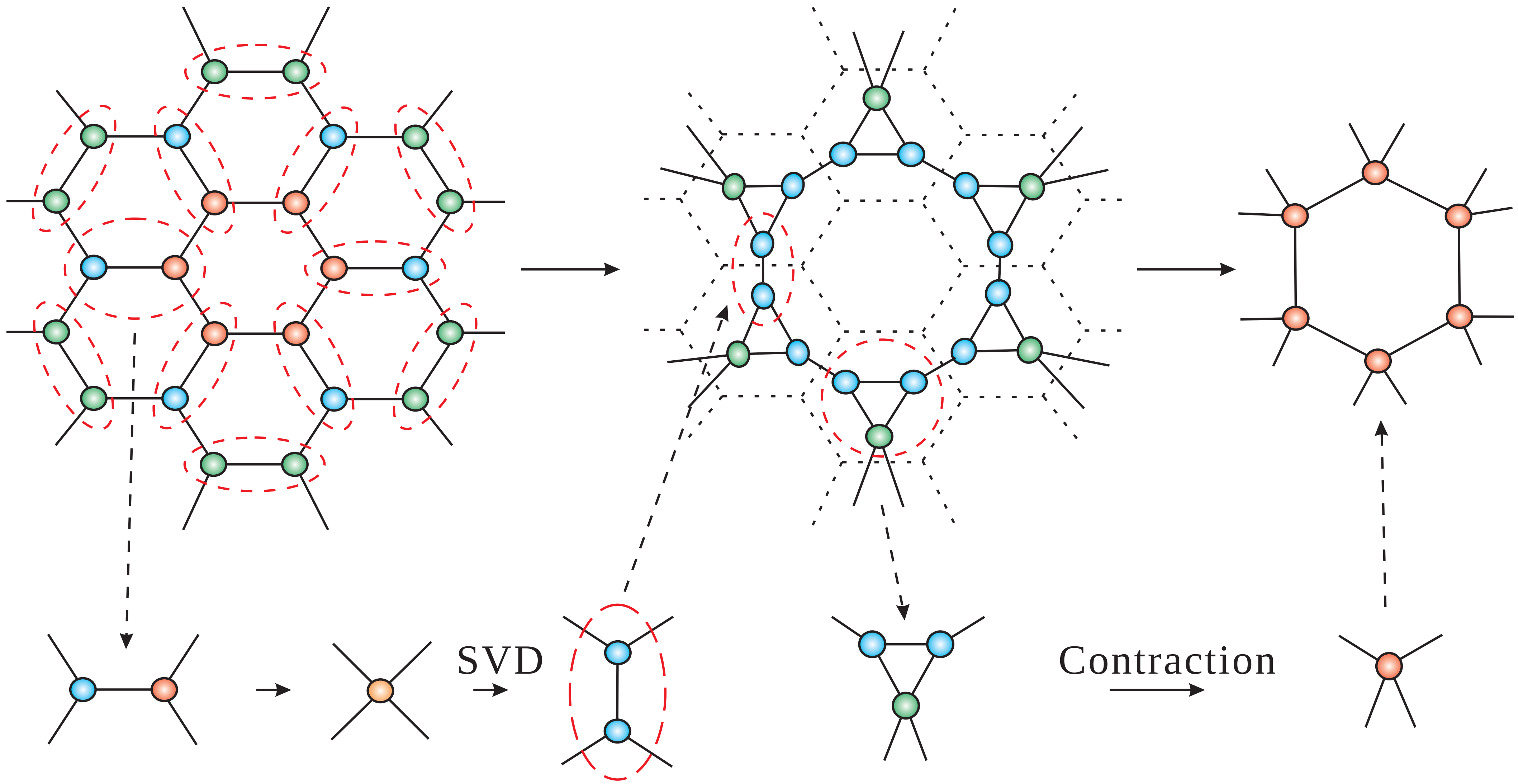
iii) factorize the unfolded matrices typically via truncated SVD

iv) reshape matrices back into new core tensors



Transformation of TN structures Cont

✓ e.g. an illustration of transformation honey-comb lattice (HCL) into tensor ring (TR) via tensor contraction and SVD

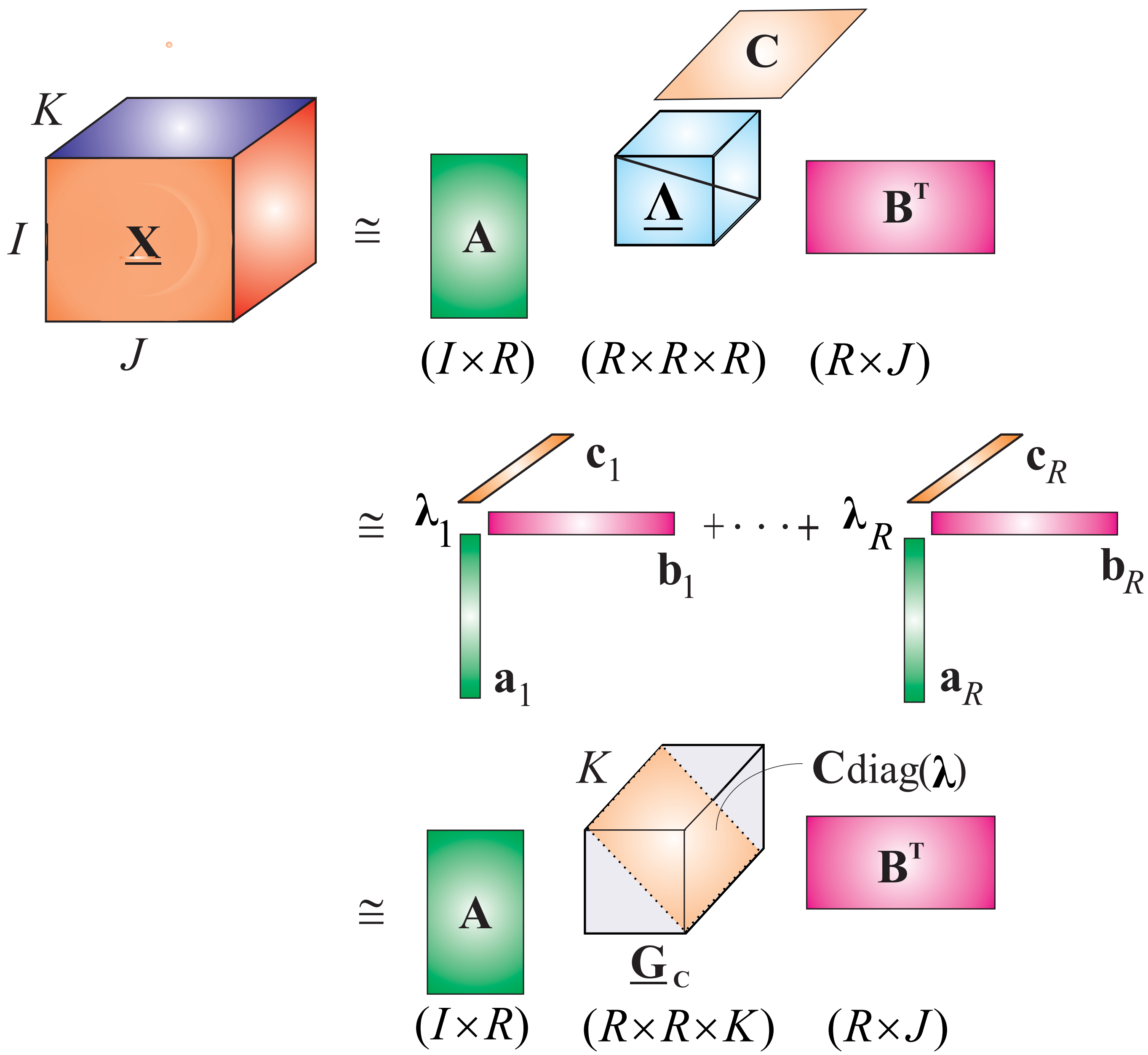


Outline

- Why tensor network
- Tensor network diagrams
- Tensor networks and decompositions
- TT decomposition: graph interpretation and algorithm

CP Decomposition

Recall CP decomposition can be expressed as a finite sum of rank-1 tensors which are formed through outer product of vectors

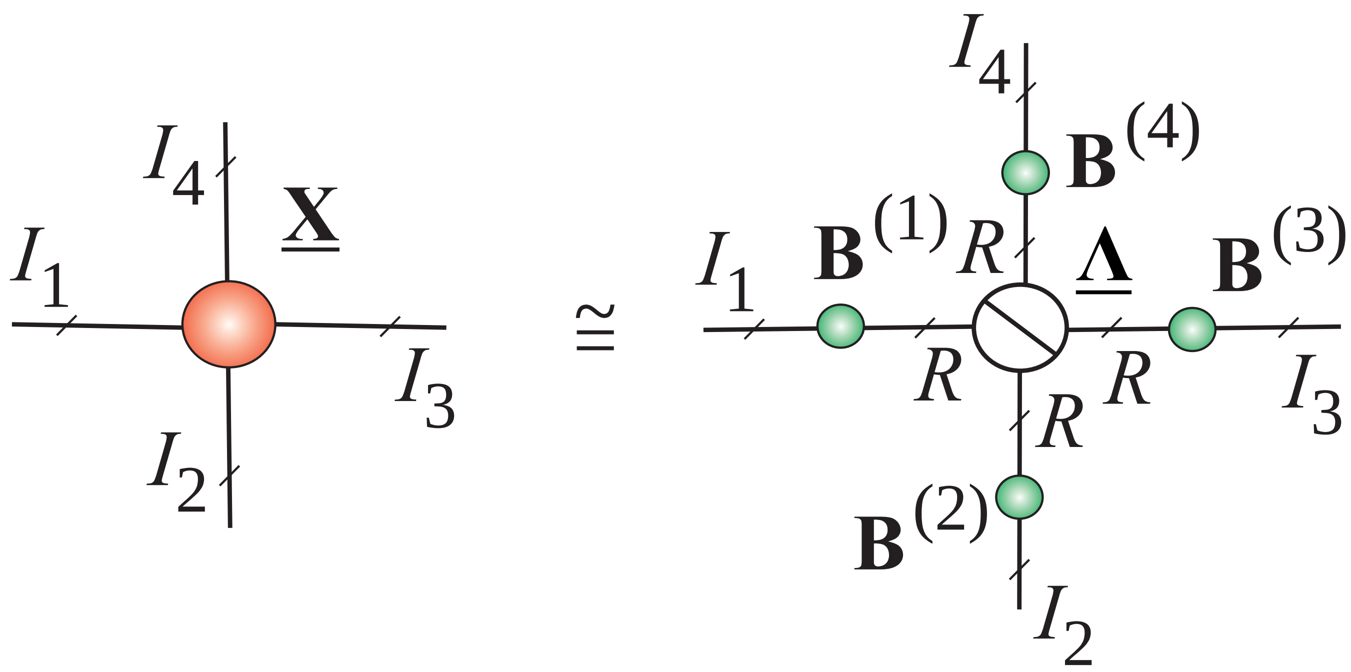


$$
\begin{aligned}
\underline{\mathbf{X}} &\cong \sum_{r=1}^{R} \lambda_r \, \mathbf{b}_r^{(1)} \circ \mathbf{b}_r^{(2)} \circ \cdots \circ \mathbf{b}_r^{(N)} \\
&= \underline{\mathbf{\Lambda}} \times_1 \mathbf{B}^{(1)} \times_2 \mathbf{B}^{(2)} \cdots \times_N \mathbf{B}^{(N)} \\
&= [\![\underline{\mathbf{\Lambda}}; \mathbf{B}^{(1)}, \mathbf{B}^{(2)}, \ldots, \mathbf{B}^{(N)}]\!],
\end{aligned}
$$

Recall CP decomposition can be expressed as a finite sum of rank-1 tensors which are formed through outer product of vectors

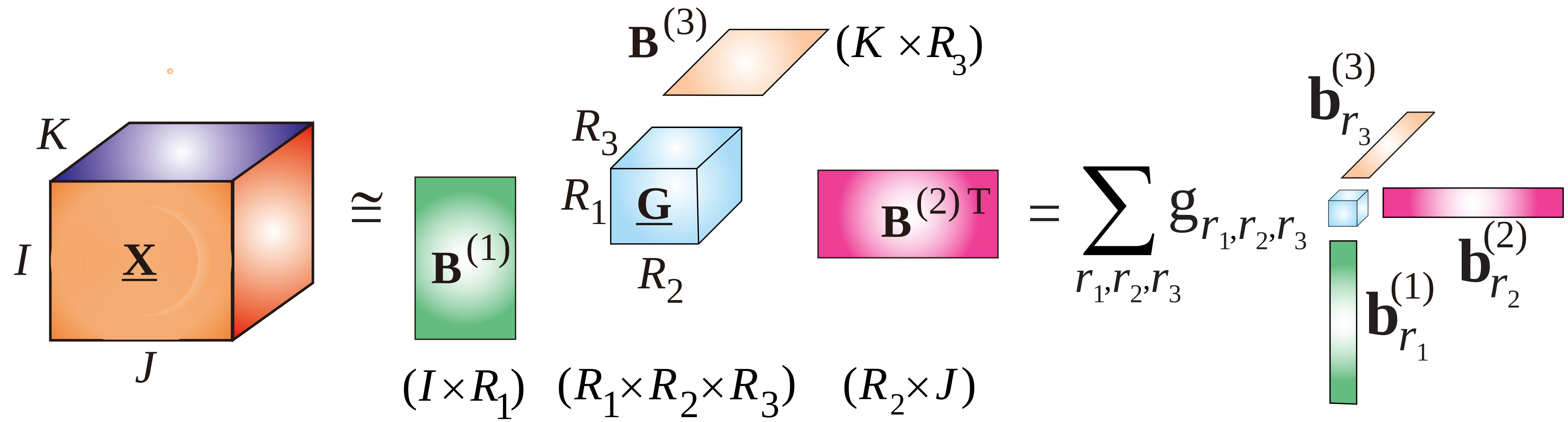
- ✓ e.g., TN diagram of a CP format of 4th-order tensor

$$\underline{\mathbf{X}} \cong \underline{\mathbf{\Lambda}} \times_1 \mathbf{B}^{(1)} \times_2 \mathbf{B}^{(2)} \times_3 \mathbf{B}^{(3)} \times_4 \mathbf{B}^{(4)} = \sum_{r=1}^{R} \lambda_r \, \mathbf{b}_r^{(1)} \circ \mathbf{b}_r^{(2)} \circ \mathbf{b}_r^{(3)} \circ \mathbf{b}_r^{(4)}$$



Recall Tucker decomposition performs the full multi-linear product in all the modes

$$\begin{aligned}\underline{\mathbf{X}} &\cong \sum_{r_1=1}^{R_1} \cdots \sum_{r_N=1}^{R_N} g_{r_1 r_2 \cdots r_N} \left(\mathbf{b}_{r_1}^{(1)} \circ \mathbf{b}_{r_2}^{(2)} \circ \cdots \circ \mathbf{b}_{r_N}^{(N)} \right) \\ &= \underline{\mathbf{G}} \times_1 \mathbf{B}^{(1)} \times_2 \mathbf{B}^{(2)} \cdots \times_N \mathbf{B}^{(N)} \\ &= [\![\underline{\mathbf{G}}; \mathbf{B}^{(1)}, \mathbf{B}^{(2)}, \ldots, \mathbf{B}^{(N)}]\!],\end{aligned}$$

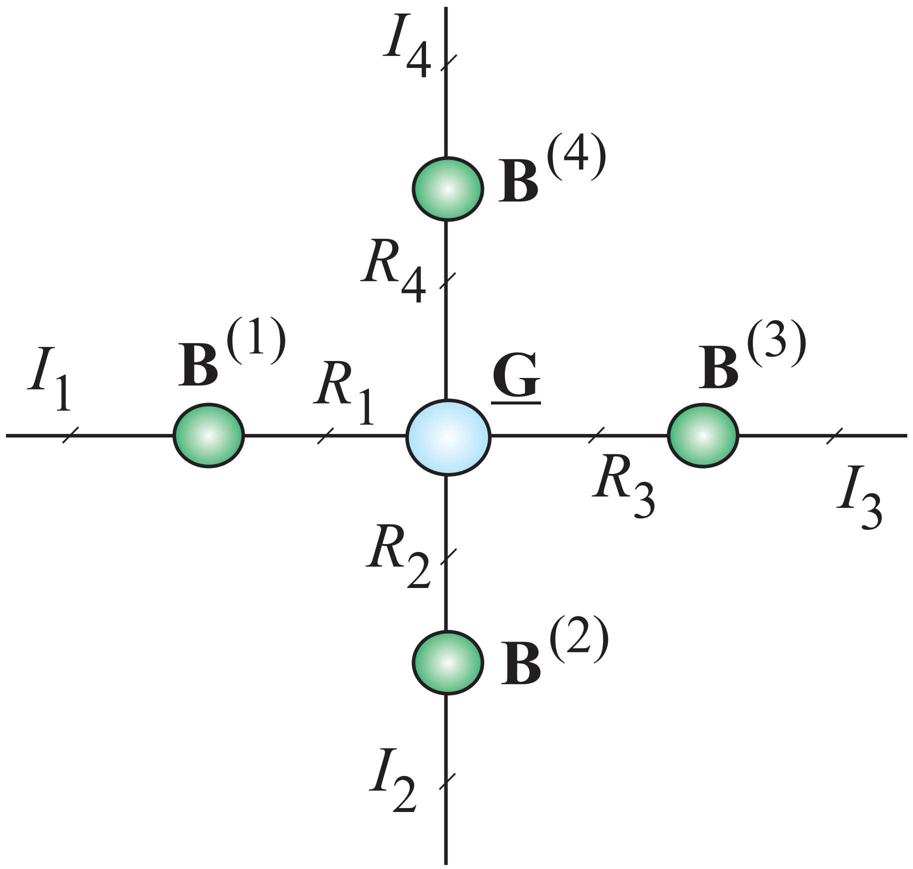


Tucker Decomposition Cont

Recall Tucker decomposition performs the full multi-linear product in all the modes

✓ e.g., TN diagram of a Tucker format of 4th-order tensor

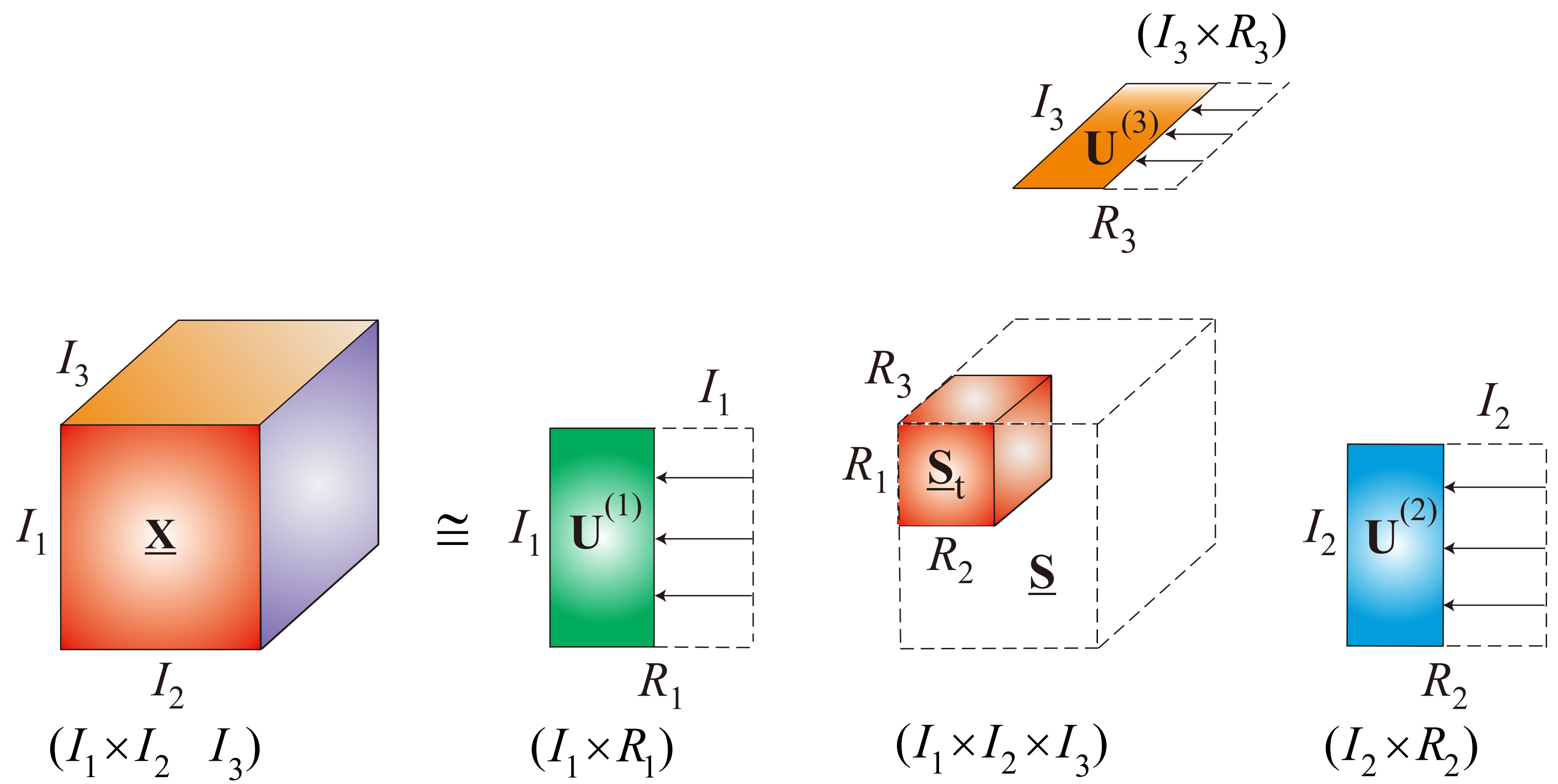
$$\underline{\mathbf{X}} \cong \underline{\mathbf{G}} \times_1 \mathbf{B}^{(1)} \times_2 \mathbf{B}^{(2)} \times_3 \mathbf{B}^{(3)} \times_4 \mathbf{B}^{(4)}$$



HOSVD

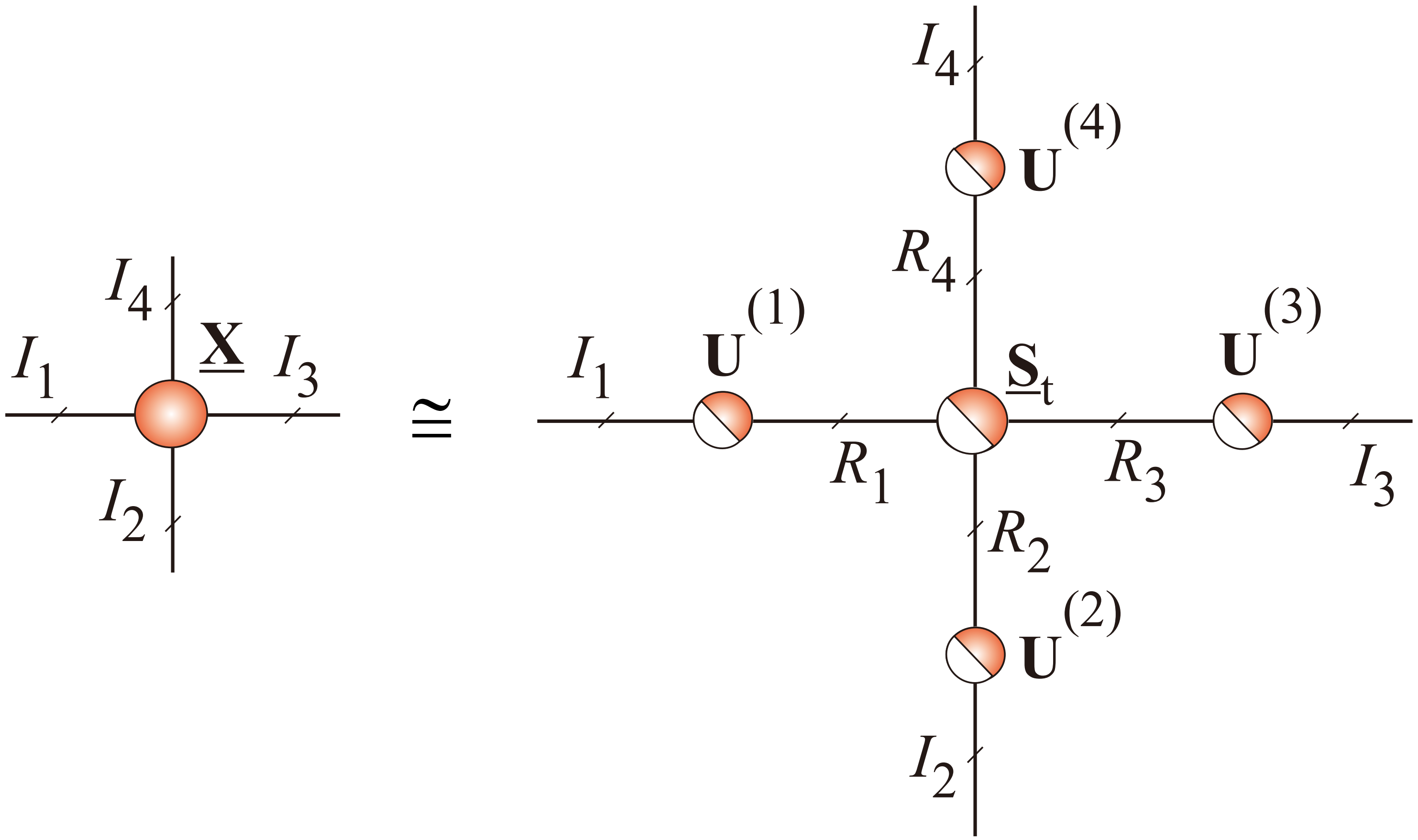
Recall high-order SVD (HOSVD) a special form of constrained Tucker decomposition with $\mathbf{B}^{(n)} = \mathbf{U}^{(n)} \in \mathbb{R}^{I_n \times I_n}$ are orthogonal factor matrices and $\underline{\mathbf{G}} = \underline{\mathbf{S}} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ is all-orthogonal core tensor

$$\underline{\mathbf{X}} = \underline{\mathbf{S}} \times_1 \mathbf{U}^{(1)} \times_2 \mathbf{U}^{(2)} \cdots \times_N \mathbf{U}^{(N)}$$



✓ e.g., TN diagram of a HOSVD of 4th-order tensor

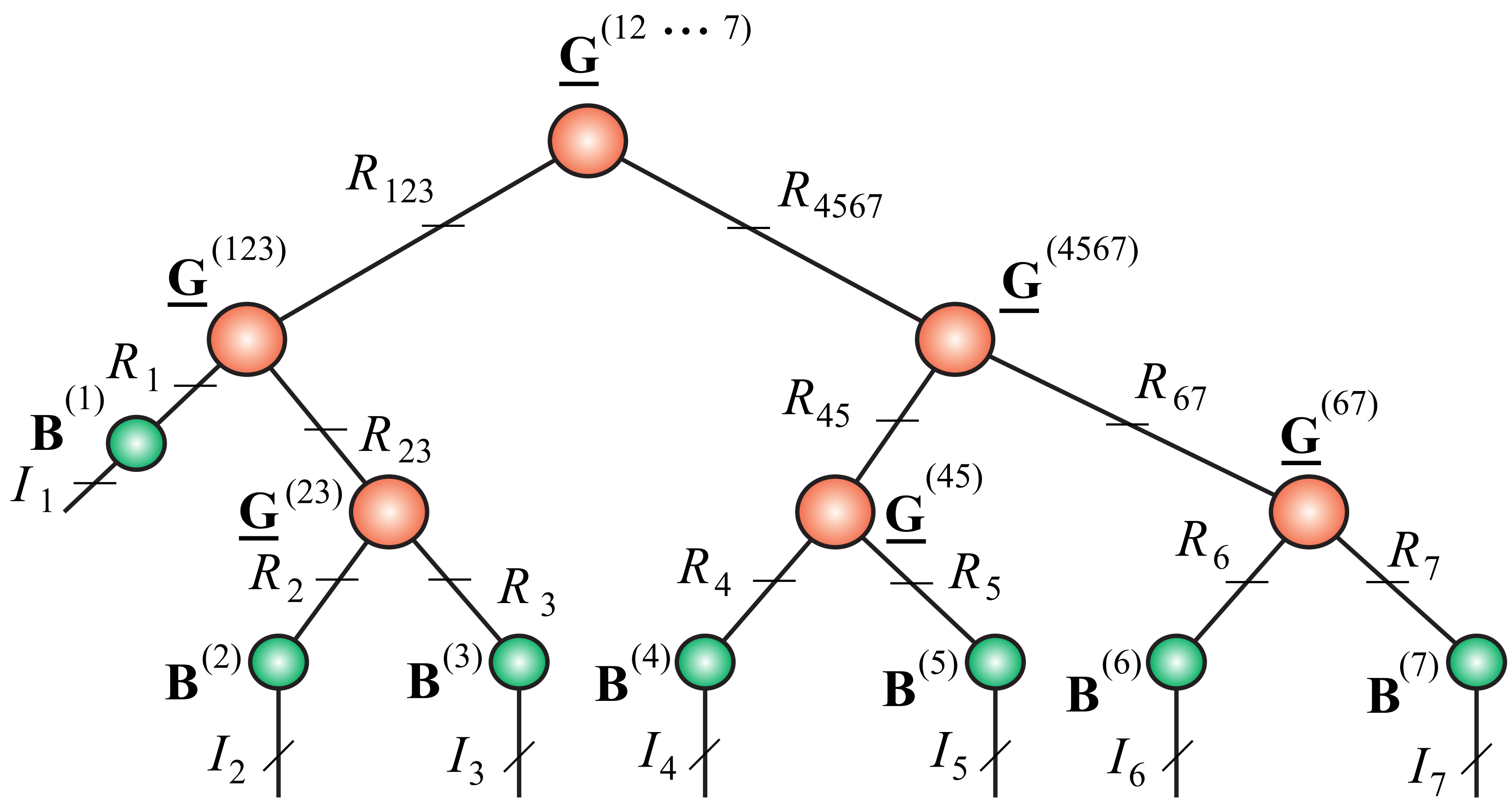
$$\underline{\mathbf{X}} \cong \underline{\mathbf{S}}_t \times_1 \mathbf{U}^{(1)} \times_2 \mathbf{U}^{(2)} \times_3 \mathbf{U}^{(3)} \times_4 \mathbf{U}^{(4)}$$



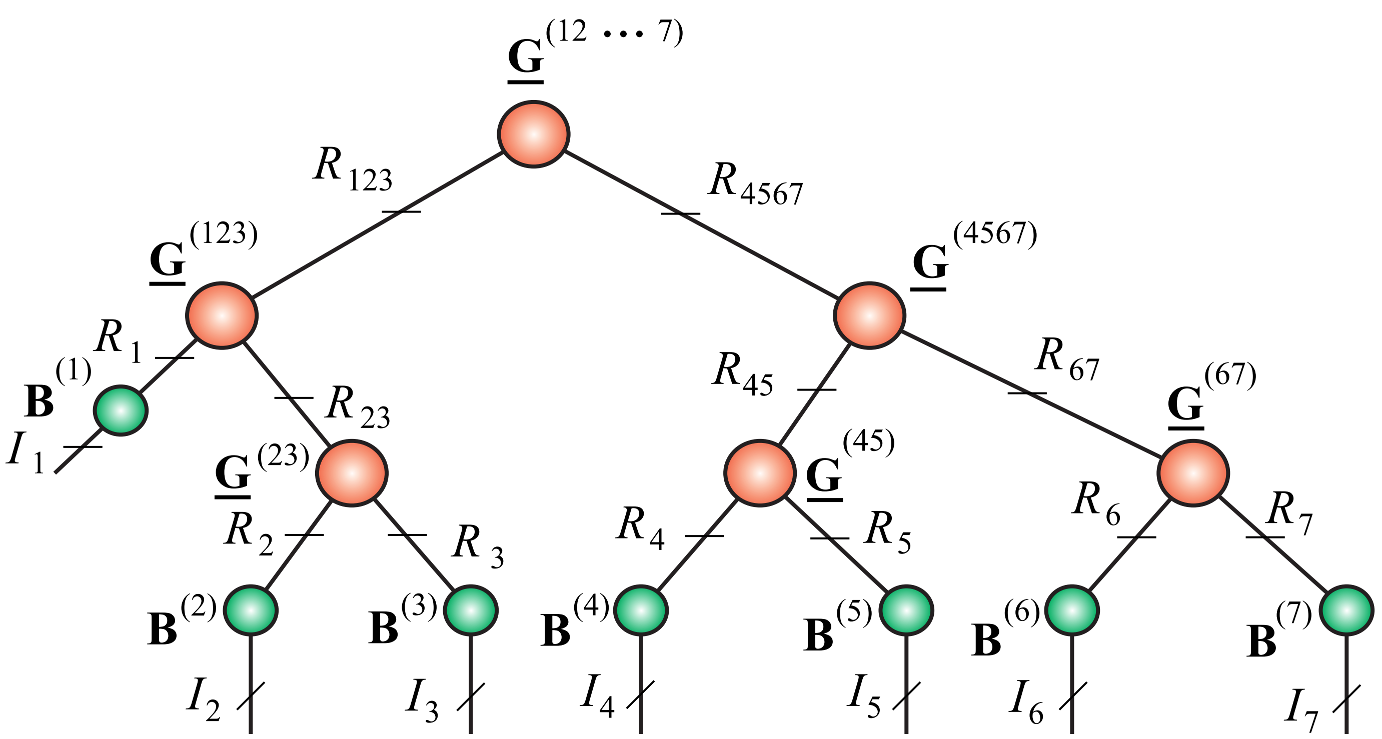
Hierarchical Tucker Decomposition

- The hierarchical Tucker decomposition (HT) requires splitting the set of modes of a tensor in a hierarchical way
- HT results in a binary tree containing a subset of modes at each branch called a dimension tree $T_N,\ N > 1$ which satisfies
  - ✓ all nodes $t \in T_N$ are non-empty subsets of $\{1,\ 2, \ldots,\ \mathrm{N}\}$
  - ✓ the set $t_{root} = \{1, 2, \ldots, N\}$ is the root node of T_N
  - ✓ each non-leaf node has two children $u, v \in T_N$ such that t is a disjoint union $t = u \cup v$

- An illustration of HT decomposition of $\underline{\mathbf{X}} \in \mathbb{R}^{I_1 \times \cdots \times I_7}$ with a given set of integers $\{R_t\}_{t \in T_7}$, i.e. HT ranks



- Let intermediate tensors $\underline{\mathbf{X}}^{(t)}$ with node $t = \{n_1, \ldots, n_k\} \subset \{1, \ldots, 7\}$ have the size $I_{n_1} \times I_{n_2} \times \cdots \times I_{n_k} \times R_t$
- Let $\mathbf{X}^{(t)} \equiv \mathbf{X}^{(t)}_{<k>} \in \mathbb{R}^{I_{n_1} I_{n_2} \cdots I_{n_k} \times R_t}$ denotes unfolded of $\underline{\mathbf{X}}^{(t)}$
- Let $\underline{\mathbf{G}}^{(t)} \in \mathbb{R}^{R_u \times R_v \times R_t}$ be the core tensor linking left and right child of t, HT can be expressed recursively



$$\mathrm{vec}(\underline{\mathbf{X}}) \cong (\mathbf{X}^{(123)} \otimes_L \mathbf{X}^{(4567)})\ \mathrm{vec}(\mathbf{G}^{(12\cdots7)})$$

$$\mathbf{X}^{(123)} \cong (\mathbf{B}^{(1)} \otimes_L \mathbf{X}^{(23)})\ \mathbf{G}^{(123)}_{<2>}$$

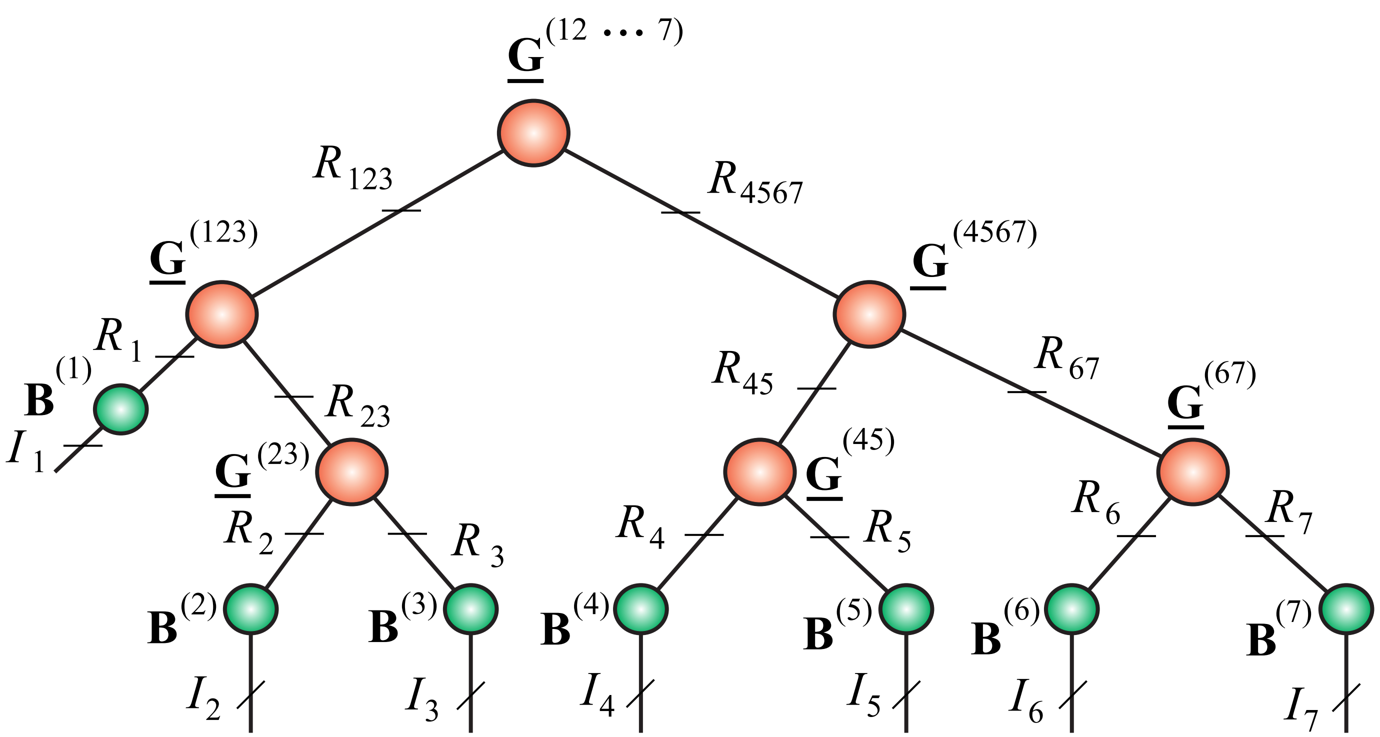
$$\mathbf{X}^{(4567)} \cong (\mathbf{X}^{(45)} \otimes_L \mathbf{X}^{(67)})\ \mathbf{G}^{(4567)}_{<2>}$$

$$\mathbf{X}^{(23)} \cong (\mathbf{B}^{(2)} \otimes_L \mathbf{B}^{(3)})\ \mathbf{G}^{(23)}_{<2>}$$

$$\mathbf{X}^{(45)} \cong (\mathbf{B}^{(4)} \otimes_L \mathbf{B}^{(5)})\ \mathbf{G}^{(45)}_{<2>}$$

$$\mathbf{X}^{(67)} \cong (\mathbf{B}^{(6)} \otimes_L \mathbf{B}^{(7)})\ \mathbf{G}^{(67)}_{<2>}$$

Equivalently, with tensor notations HT expression becomes



$$\underline{\mathbf{X}} \cong \sum_{r_{123}=1}^{R_{123}} \sum_{r_{4567}=1}^{R_{4567}} g^{(12\cdots7)}_{r_{123},r_{4567}} \, \underline{\mathbf{X}}^{(123)}_{r_{123}} \circ \underline{\mathbf{X}}^{(4567)}_{r_{4567}}$$

$$\underline{\mathbf{X}}^{(123)}_{r_{123}} \cong \sum_{r_1=1}^{R_1} \sum_{r_{23}=1}^{R_{23}} g^{(123)}_{r_1,r_{23},r_{123}} \mathbf{b}^{(1)}_{r_1} \circ \mathbf{X}^{(23)}_{r_{23}}$$

$$\underline{\mathbf{X}}^{(4567)}_{r_{4567}} \cong \sum_{r_{45}=1}^{R_{45}} \sum_{r_{67}=1}^{R_{67}} g^{(4567)}_{r_{45},r_{67},r_{4567}} \mathbf{X}^{(45)}_{r_{45}} \circ \mathbf{X}^{(67)}_{r_{67}}$$

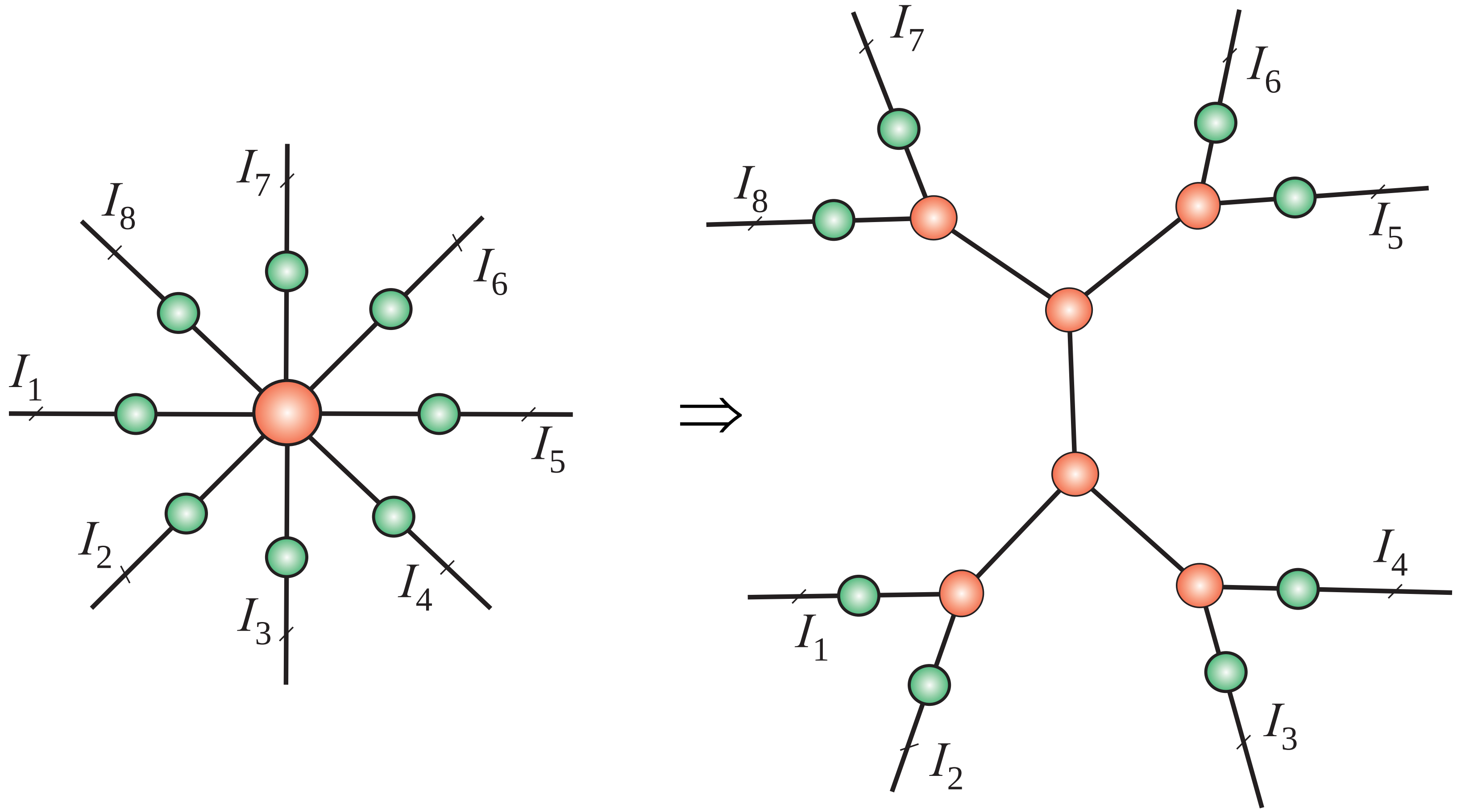
$$\mathbf{X}^{(23)}_{r_{23}} \cong \sum_{r_2=1}^{R_2} \sum_{r_3=1}^{R_3} g^{(23)}_{r_2,r_3,r_{23}} \, \mathbf{b}^{(2)}_{r_2} \circ \mathbf{b}^{(3)}_{r_3}$$

$$\mathbf{X}^{(45)}_{r_{45}} \cong \sum_{r_4=1}^{R_4} \sum_{r_5=1}^{R_5} g^{(45)}_{r_4,r_5,r_{45}} \, \mathbf{b}^{(4)}_{r_4} \circ \mathbf{b}^{(5)}_{r_5}$$

$$\mathbf{X}^{(67)}_{r_{67}} \cong \sum_{r_6=1}^{R_6} \sum_{r_7=1}^{R_7} g^{(67)}_{r_6,r_7,r_{67}} \, \mathbf{b}^{(6)}_{r_6} \circ \mathbf{b}^{(7)}_{r_7}$$

Links between Tucker and HT

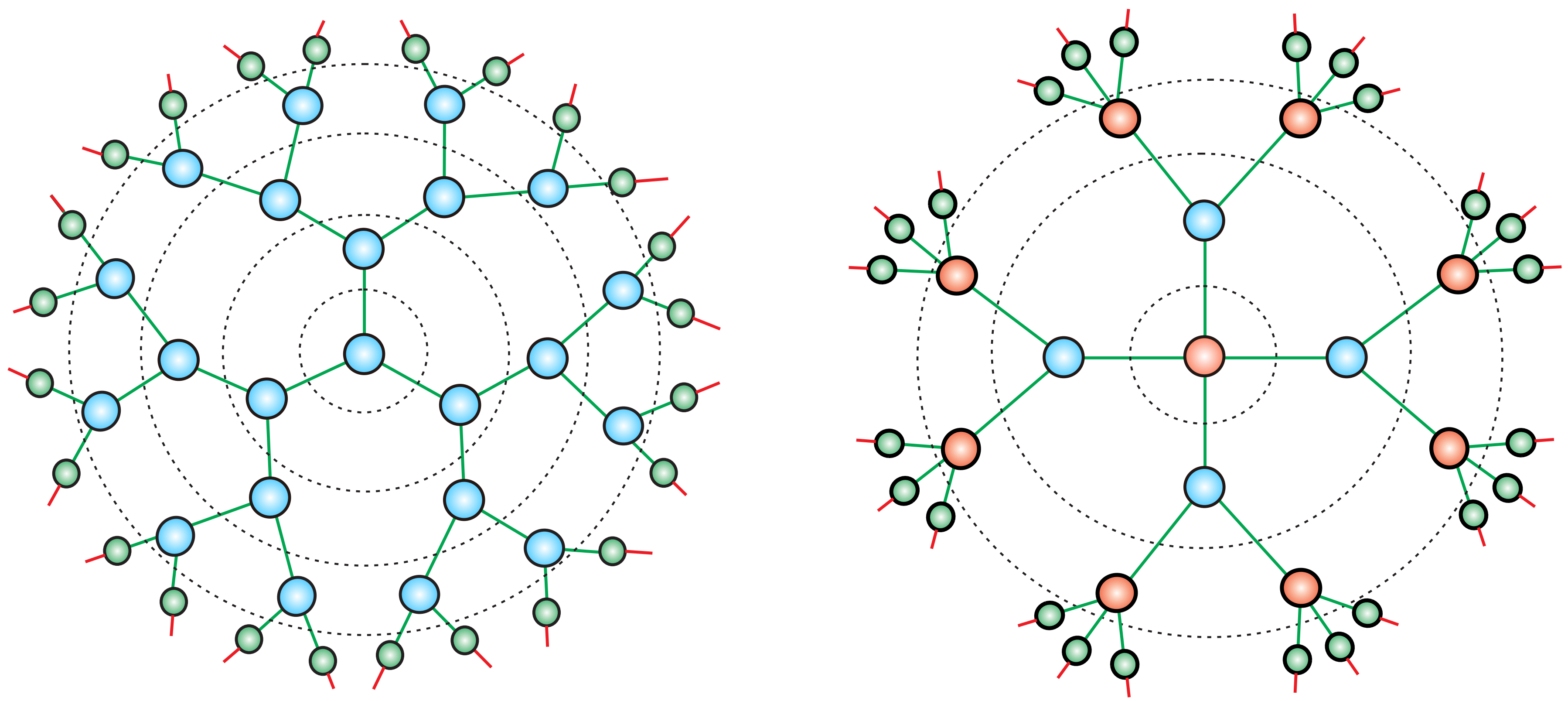
- HT leads naturally to a distributed Tucker decomposition
- A single core in Tucker is replaced by interconnected cores of low-order in HT
- In such distributed network some cores are connected directly with some of factor matrices



Tree Tensor Network State

Tree tensor network state (TTNS) can be considered as a generalization of HT (TT), and as a distributed model for Tucker-N decomposition

✓ e.g. TN diagram of TTNS 3rd-order and 4th-order tensor cores for the representation of 24th-order tensors

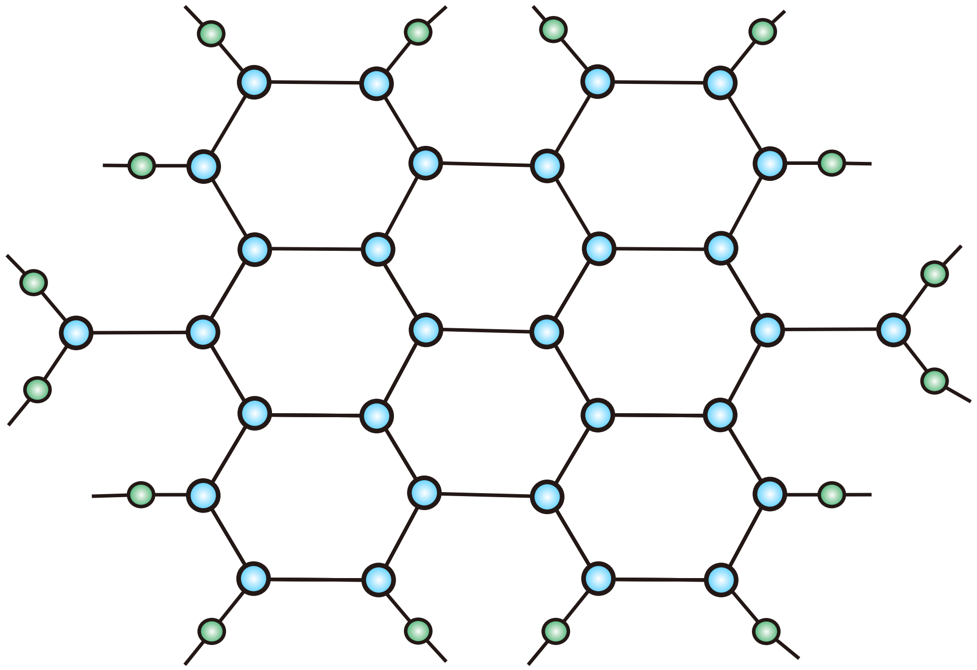


TN with Circles

- TN dramatically reduces computational cost and provide distributed storage through low-rank TN approximation
- However, the ranks of HT (or TT) increase rapidly with the data order and desired approximation accuracy
- The ranks can be kept considerably small through special architectures of TN with circles
  - ✓ e.g. projected entangled pair states (PEPS)
  - ✓ honey-comb lattice (HCL)
  - ✓ multi-scale entanglement renormalization ansatz (MERA)
- TN with circles pays the price of higher computational complexity w.r.t. tensor contraction due to many circles

Honey-comb lattice (HCL) consists of only 3rd-order core tensors

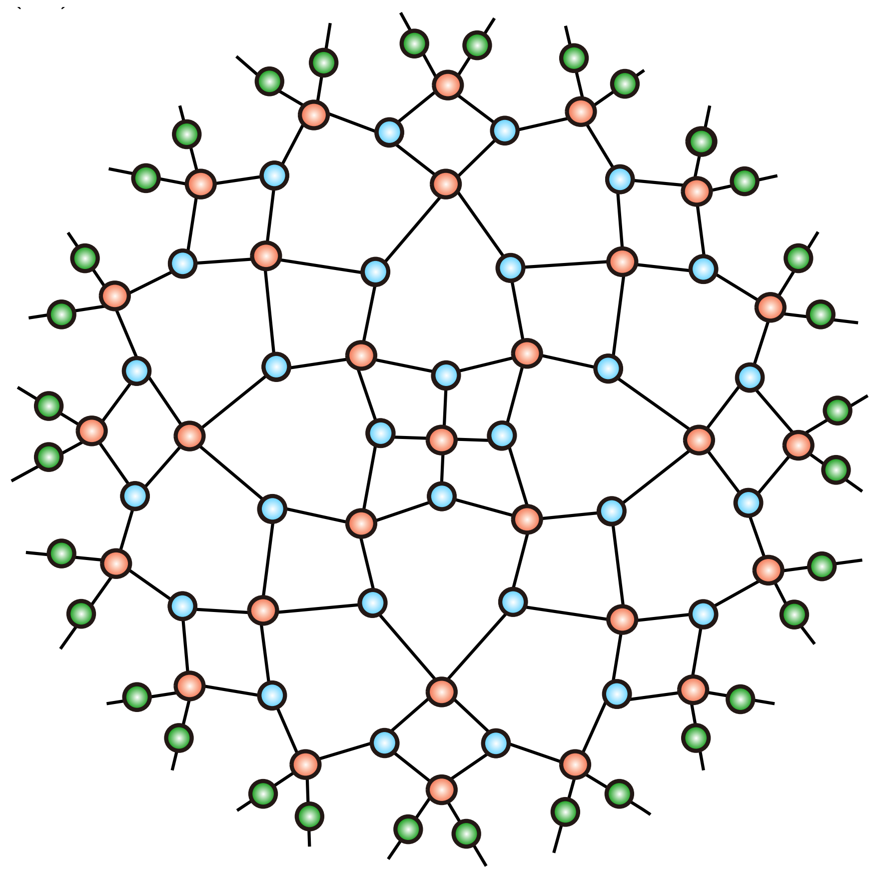
✓ e.g. TN diagram of HCL of a 16th-order tensor



TN with Circles—MERA

Multi-scale entanglement renormalization ansatz (MERA) consists of both 3rd-order and 4th-order core tensors

- ✓ MERA core tensors are much smaller, which dramatically reduce number of free parameters and provide more efficient storage of huge-scale data tensors
- ✓ MERA allows to model complex functions and interactions between variables
- ✓ e.g. TN diagram of MERA of a 32th-order tensor



Outline

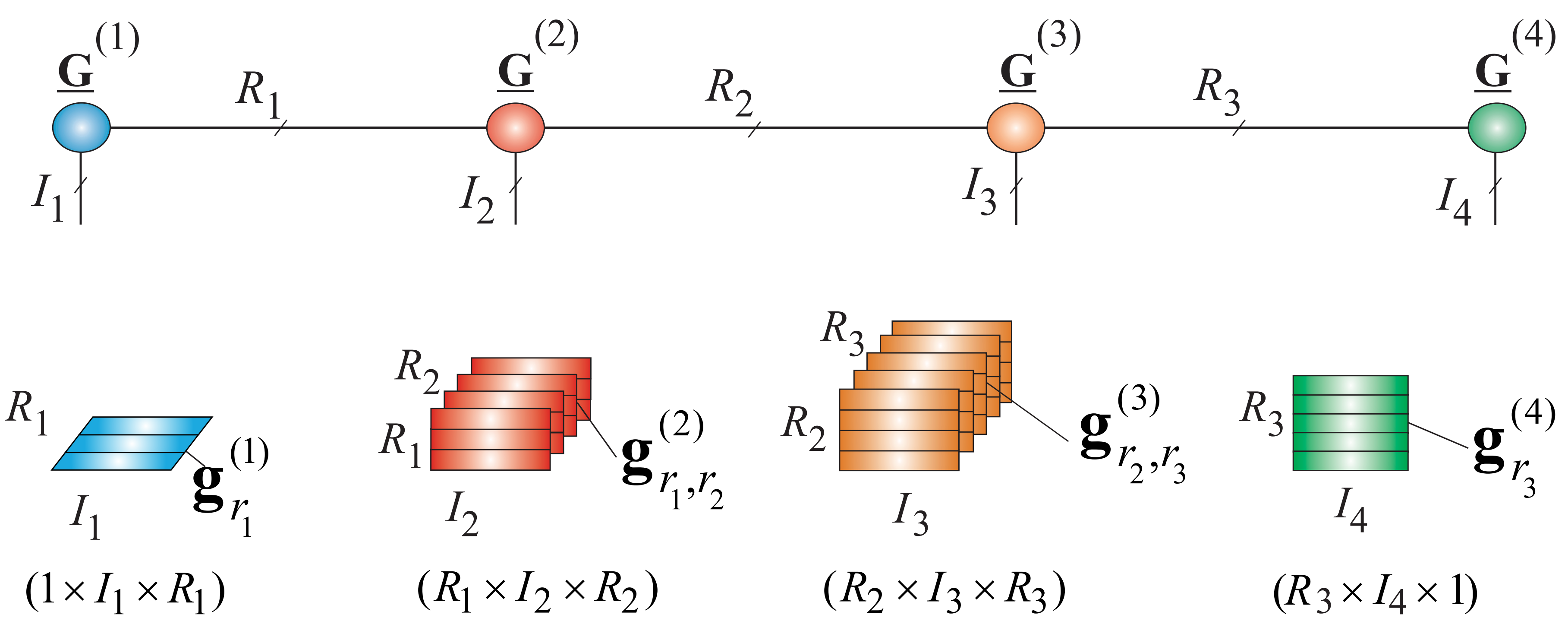
- Why tensor network
- Tensor network diagrams
- Tensor networks and decompositions
- TT decomposition: graph interpretation and algorithm

Tensor Train Decomposition

- Tensor train decomposition (TT) or matrix product state (MPS) is a special case of tree structured TN
- All the nodes (TT-cores) of the underlying TN are connected in cascade or train
- Each tensor entry can be computed as a cascade multiplication of appropriate matrices (slices of TT-cores)

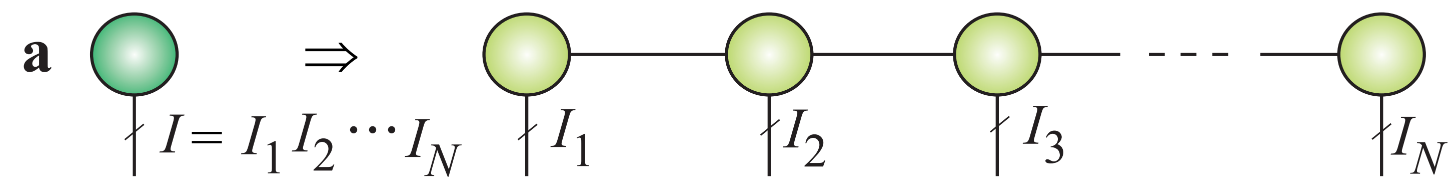
$$x_{i_1, i_2, \ldots, i_N} = \mathbf{G}^{(1)}_{i_1} \mathbf{G}^{(2)}_{i_2} \cdots \mathbf{G}^{(N)}_{i_N} \text{ where } \mathbf{G}^{(n)}_{i_n} = \underline{\mathbf{G}}^{(n)}(:, i_n, :) \in \mathbb{R}^{R_{n-1} \times R_n}$$

$$\underline{\mathbf{X}} = \underline{\mathbf{G}}^{(1)} \times^1 \underline{\mathbf{G}}^{(2)} \times^1 \cdots \times^1 \underline{\mathbf{G}}^{(N)} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$$

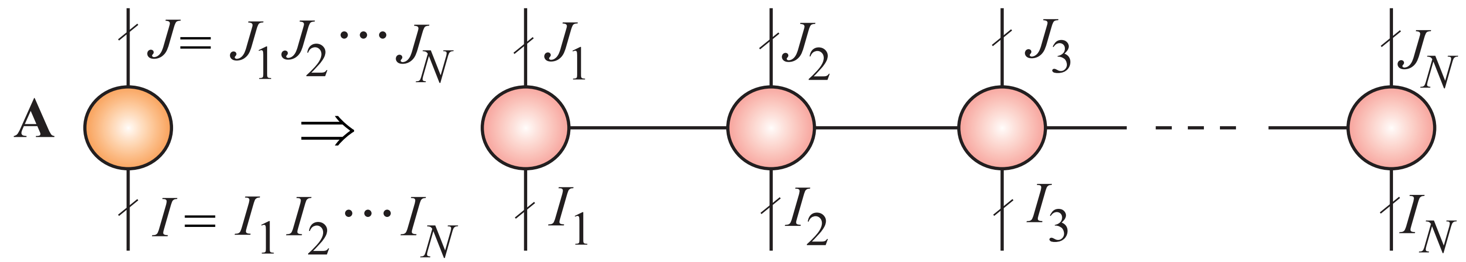


TT Example

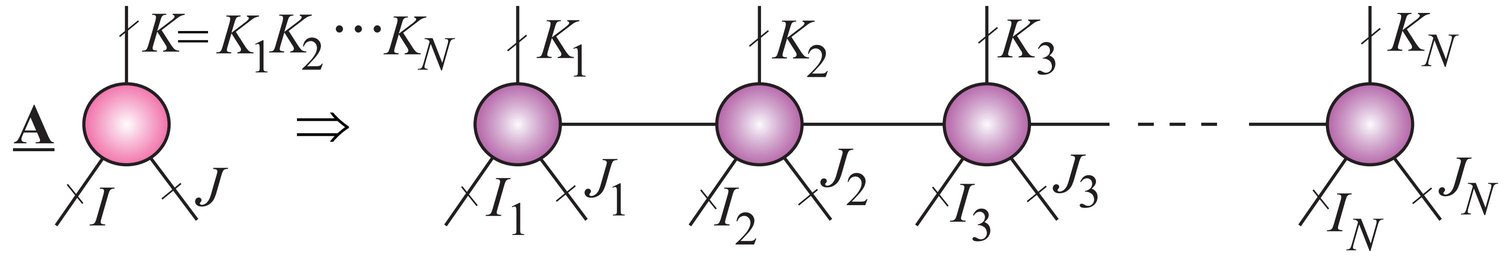
- TT format of tensorized vector $\mathbf{a} \in \mathbb{R}^{I}$



- TT format of tensorized matrix $\mathbf{A} \in \mathbb{R}^{I \times J}$



- TT format of tensorized large-scale low-order tensor $\underline{\mathbf{A}} \in \mathbb{R}^{I \times J \times K}$



Advantages of TT

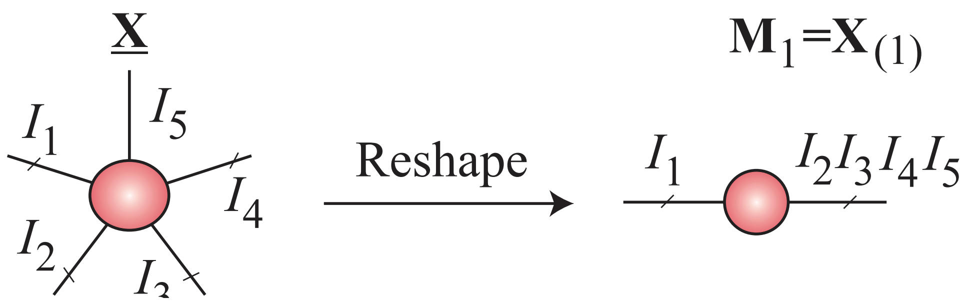
Main benefits of TT format:

- No need to specify the binary dimension tree as HT format
- Simplicity in performing basic math operations on tensors using TT format, employing only core tensors
  - ✓ e.g., matrix-by-matrix multiplication, tensor addition, tensor entry-wise product
- Only TT-cores needs to be stored, making the number of parameters to scale linearly in tensor order
  - ✓ $\sum_{n=1}^{N} R_{n-1} R_n I_n \sim \mathcal{O}(NR^2 I), \quad R := \max_n \{R_n\}, \quad I := \max_n \{I_n\}$

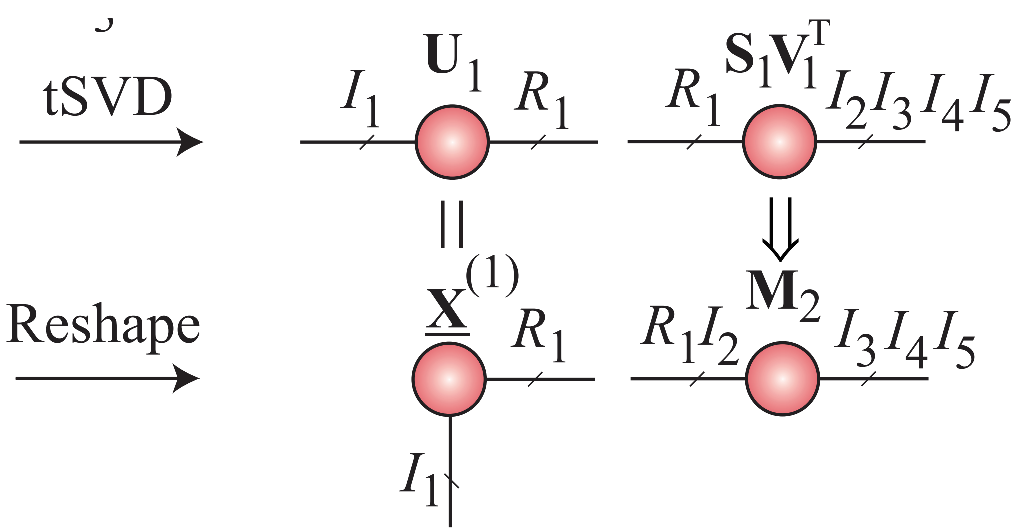
Algorithm for TT Decomposition

- TT-SVD algorithm for TT decomposition applies truncated SVD (tSVD) sequentially to the unfolding matrices

  i) High-order tensor $\underline{\mathbf{X}}$ is first reshaped into a long matrix $\mathbf{M}_1$



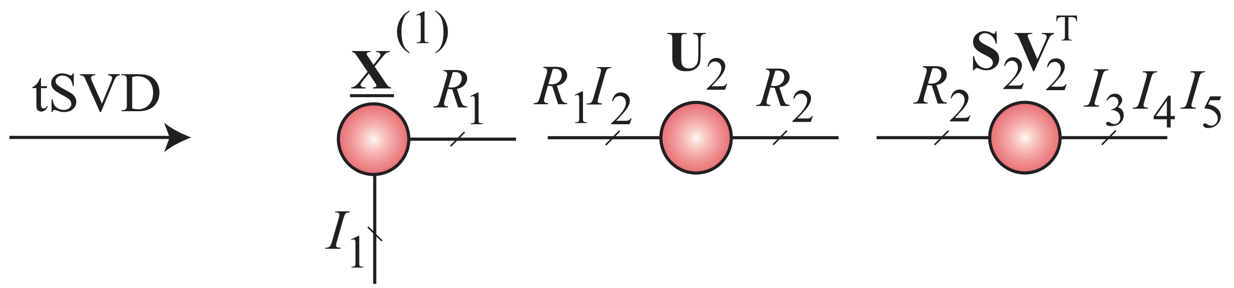
  ii) tSVD is performed to produce low-rank factorization $\mathbf{M}_1 \cong \mathbf{U}_1\mathbf{S}_1\mathbf{V}_1^{\mathrm{T}}$



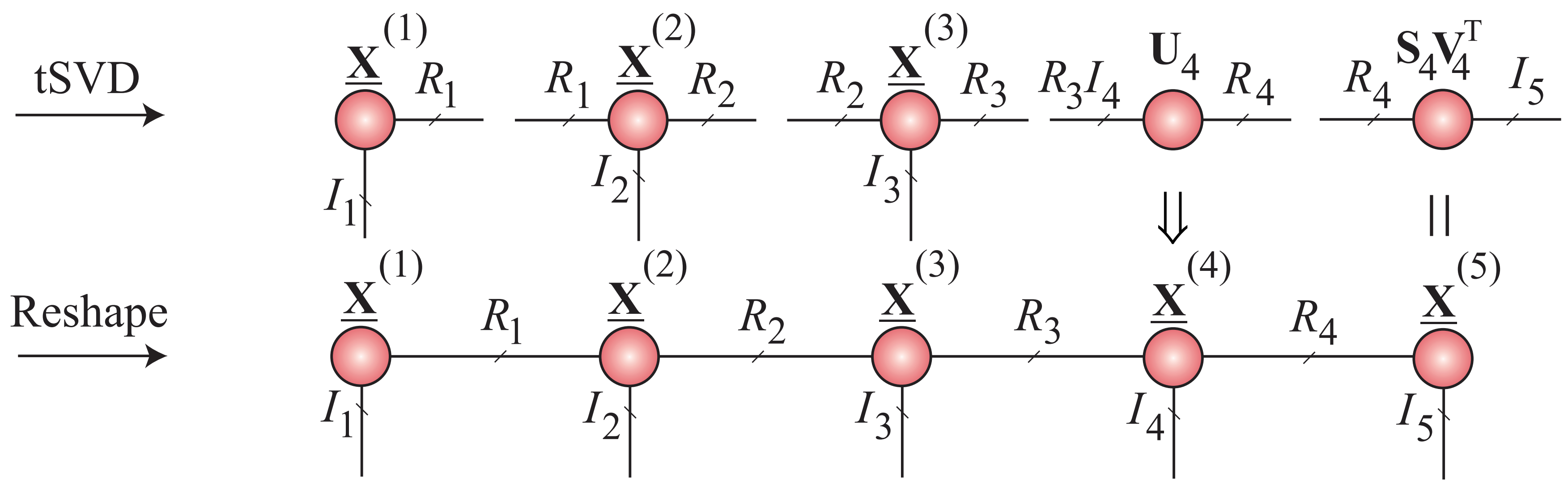
  iii) Matrix $\mathbf{U}_1$ becomes the first core $\underline{\mathbf{X}}^{(1)}$, while $\mathbf{S}_1\mathbf{V}_1^{\mathrm{T}}$ is reshaped into $\mathbf{M}_2$

Algorithm for TT Decomposition Cont

iv) Perform tSVD to yield $\mathbf{M}_2 \cong \mathbf{U}_2\mathbf{S}_2\mathbf{V}_2^{\mathrm{T}}$, and reshape $\mathbf{U}_2$ into an core $\underline{\mathbf{X}}^{(2)}$



v) Repeat the procedure until all the cores are extracted



- TT-SVD algorithm using truncated SVD (tSVD)

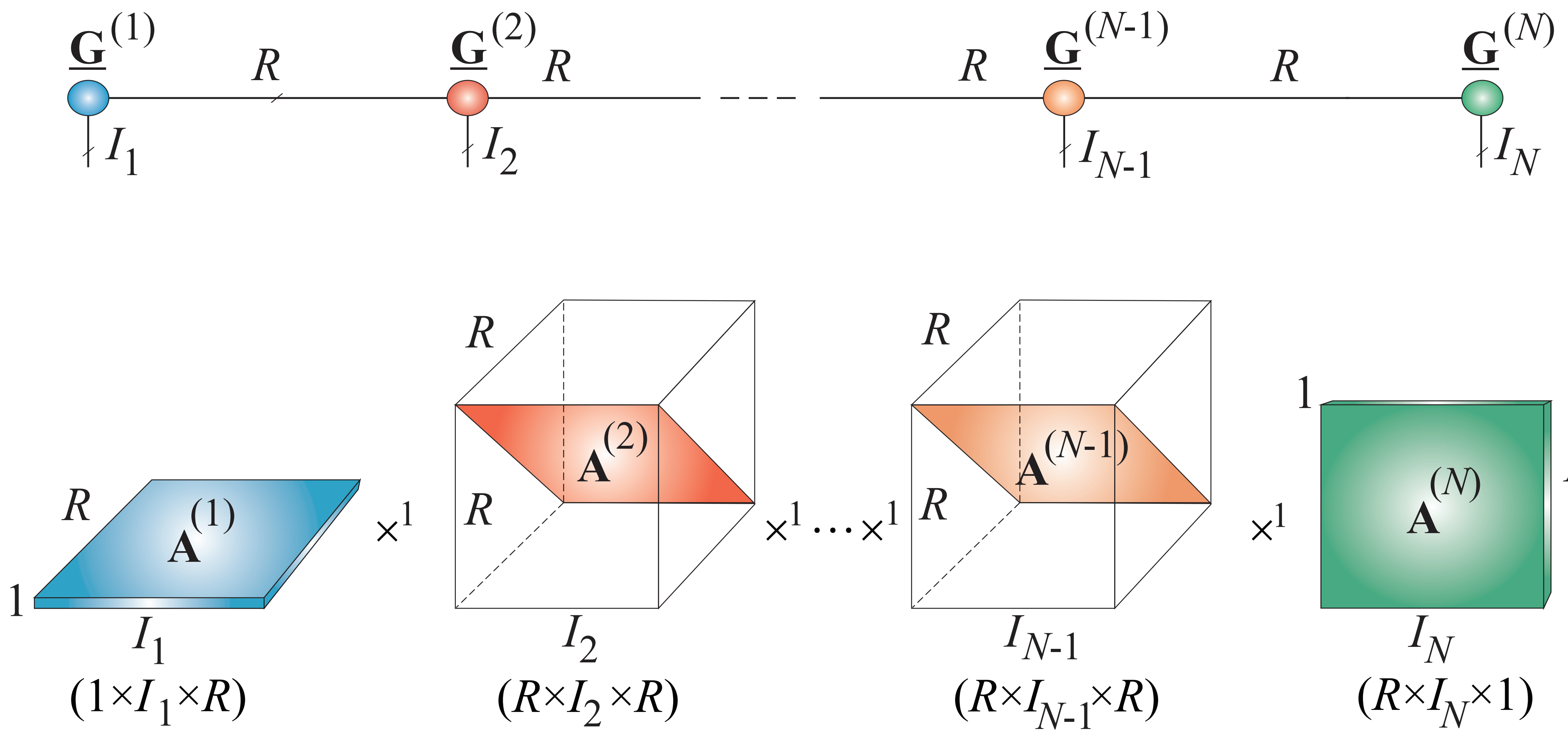
Input: Nth-order tensor $\underline{\mathbf{X}} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ and approximation accuracy ε

Output: Approximative representation of a tensor in the TT format
$\hat{\underline{\mathbf{X}}} = \langle\langle \hat{\underline{\mathbf{X}}}^{(1)}, \hat{\underline{\mathbf{X}}}^{(2)}, \ldots, \hat{\underline{\mathbf{X}}}^{(N)} \rangle\rangle$, such that $\|\underline{\mathbf{X}} - \hat{\underline{\mathbf{X}}}\|_F \leqslant \varepsilon$

1: Unfolding of tensor $\underline{\mathbf{X}}$ in mode-1 $\mathbf{M}_1 = \mathbf{X}_{(1)}$
2: Initialization $R_0 = 1$
3: **for** $n = 1$ to $N-1$ **do**
4: Perform tSVD $[\mathbf{U}_n, \mathbf{S}_n, \mathbf{V}_n] = \text{tSVD}(\mathbf{M}_n, \varepsilon/\sqrt{N-1})$
5: Estimate nth TT rank $R_n = \text{size}(\mathbf{U}_n, 2)$
6: Reshape orthogonal matrix $\mathbf{U}_n$ into a 3rd-order core
$\hat{\underline{\mathbf{X}}}^{(n)} = \text{reshape}(\mathbf{U}_n, [R_{n-1}, I_n, R_n])$
7: Reshape the matrix $\mathbf{V}_n$ into a matrix
$\mathbf{M}_{n+1} = \text{reshape}\left(\mathbf{S}_n \mathbf{V}_n^{\mathrm{T}}, [R_n I_{n+1}, \prod_{p=n+2}^{N} I_p]\right)$
8: **end for**
9: Construct the last core as $\hat{\underline{\mathbf{X}}}^{(N)} = \text{reshape}(\mathbf{M}_N, [R_{N-1}, I_N, 1])$
10: **return** $\langle\langle \hat{\underline{\mathbf{X}}}^{(1)}, \hat{\underline{\mathbf{X}}}^{(2)}, \ldots, \hat{\underline{\mathbf{X}}}^{(N)} \rangle\rangle$.

Links between CP and TT

Any specific TN format, especially CP, can be converted to TT format

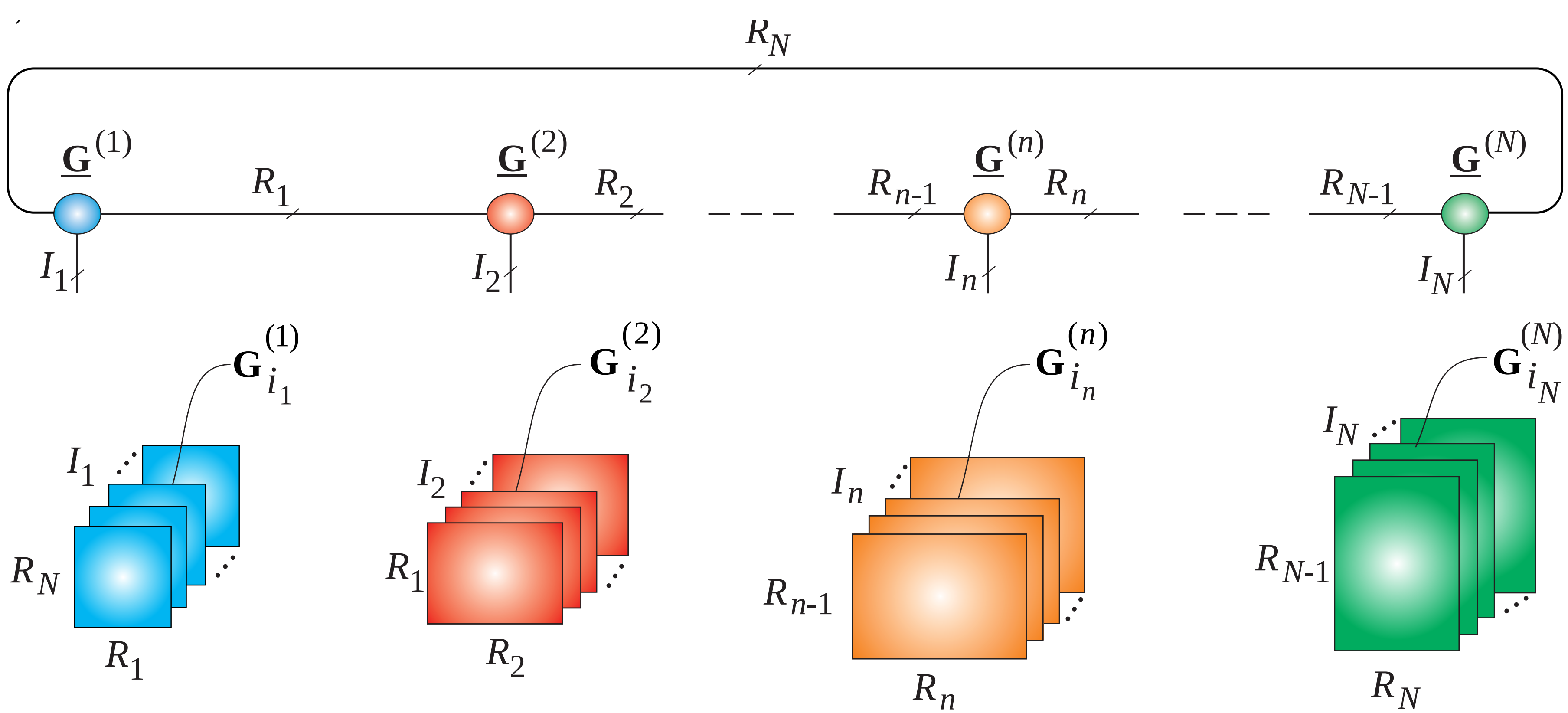


Tensor Ring Decomposition

- Tensor train decomposition (TR) generalizes TT with a single loop connecting the first and last core
- All the nodes (TR-cores) are of 3rd-order tensors

$$x_{i_1,i_2,\ldots,i_N} = \mathrm{tr}\,(\mathbf{G}^{(1)}_{i_1} \;\; \mathbf{G}^{(2)}_{i_2} \;\; \cdots \;\; \mathbf{G}^{(N)}_{i_N}) =$$

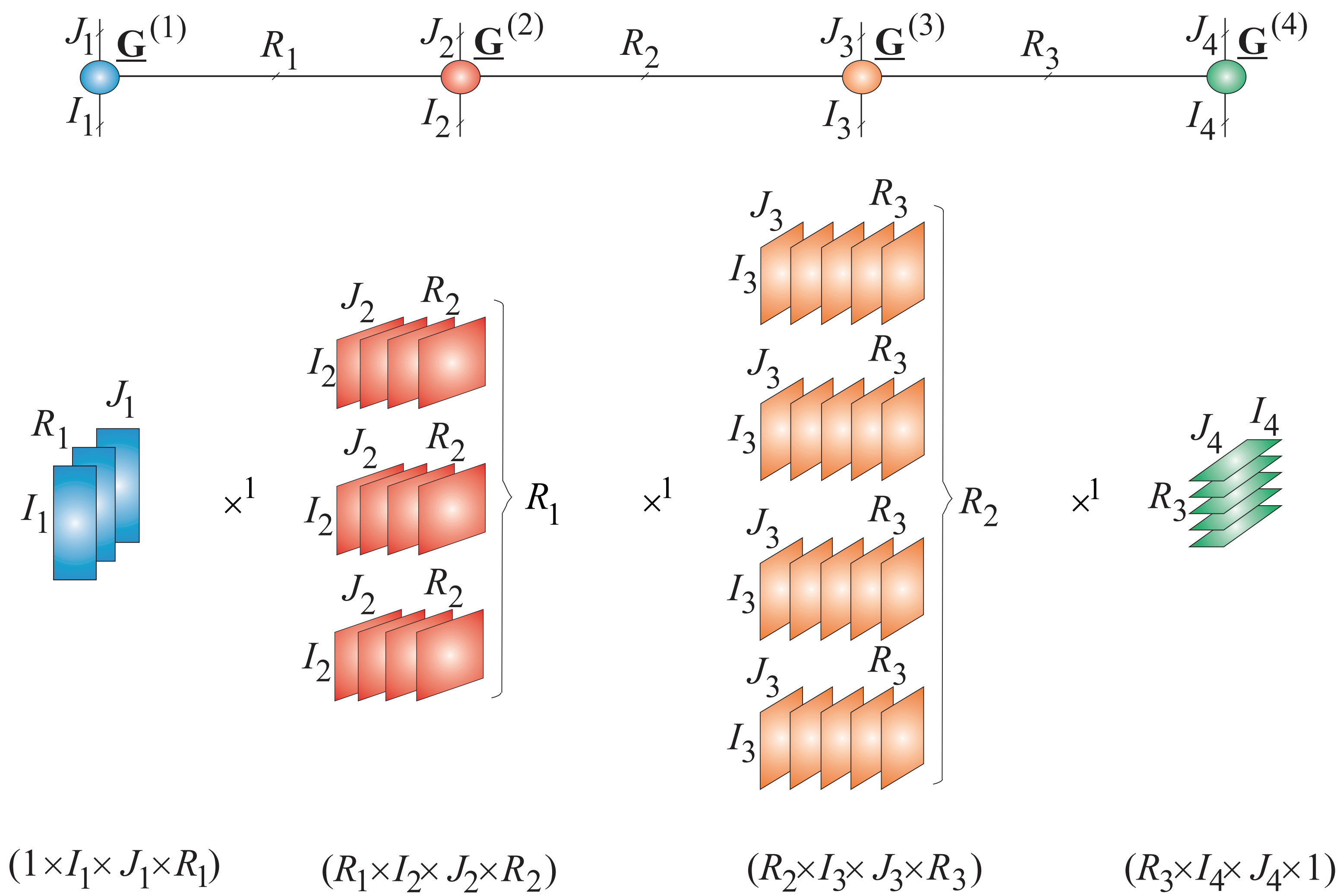
$$\sum_{r_1=1}^{R_1} \sum_{r_2=1}^{R_2} \cdots \sum_{r_N=1}^{R_N} g^{(1)}_{r_N,\,i_1,\,r_1} \; g^{(2)}_{r_1,\,i_2,\,r_2} \cdots g^{(N)}_{r_{N-1},\,i_N,\,r_N}$$



Matrix Tensor Train Decomposition

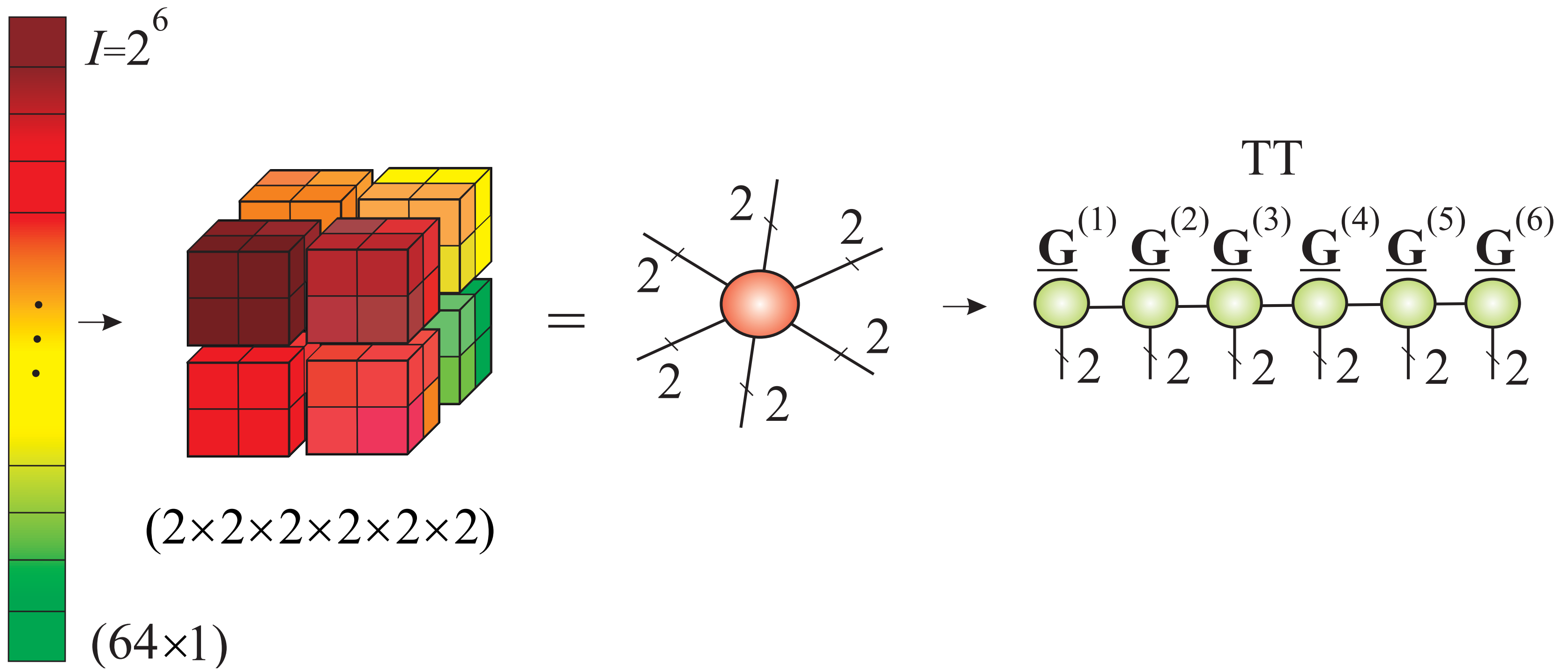
- The matrix tensor train (matrix TT) or matrix product operator (MPO) is a variant of TT that can represent huge-scale structured matrices by
  - ✓ first converting $\mathbf{X} \in \mathbb{R}^{I \times J}$ into a 2Nth-order tensor $\underline{\mathbf{X}} \in \mathbb{R}^{I_1 \times J_1 \times I_2 \times J_2 \times \cdots I_N \times J_N}$
  - ✓ then decomposing tensor into a train of 4th-order cores similar to TT-cores

$$\underline{\mathbf{X}} = \underline{\mathbf{G}}^{(1)} \times^1 \underline{\mathbf{G}}^{(2)} \times^1 \cdots \times^1 \underline{\mathbf{G}}^{(N-1)} \times^1 \underline{\mathbf{G}}^{(N)}$$



Quantized Tensor Train Decomposition

- Recall tensorization creates a high-order tensor from a low-order original data
- Quantization is a special case of tensorization with each mode has a very small size, typically 2,3 or 4
- Low-rank TN approximation with high compression ratios can be achieved by quantization
- Quantization tensor networks (QTN) adopts small-size 3rd-order tensor cores that are sparsely interconnected via tensor contraction
  - ✓ e.g. an implementation of QTN using quantized tensor train (QTT)



Operations in TT Format

In TT format, basic math operations can be efficiently performed using slice matrices of individual core tensors

✓ e.g. consider matrix-by-vector multiplication $\mathbf{A}\mathbf{x} = \mathbf{y}$

- ➡ matrix $\mathbf{A} \in \mathbb{R}^{I \times J}$ and vectors $\mathbf{x} \in \mathbb{R}^{J}$, $\mathbf{y} \in \mathbb{R}^{I}$ are represented in TT format with size $I = I_1 I_2 \cdots I_N$ and $J = J_1 J_2 \cdots J_N$
- ➡ cores are $\underline{\mathbf{A}}^{(n)} \in \mathbb{R}^{P_{n-1} \times I_n \times J_n \times P_n}$, $\underline{\mathbf{X}}^{(n)} \in \mathbb{R}^{R_{n-1} \times J_n \times R_n}$ and $\underline{\mathbf{Y}}^{(n)} \in \mathbb{R}^{Q_{n-1} \times I_n \times Q_n}$

$$\underline{\mathbf{A}} = \sum_{p_1, p_2, \ldots, p_{N-1}=1}^{P_1, P_2, \ldots, P_{N-1}} \mathbf{A}^{(1)}_{1, p_1} \circ \mathbf{A}^{(2)}_{p_1, p_2} \circ \cdots \circ \mathbf{A}^{(N)}_{p_{N-1}, 1}$$

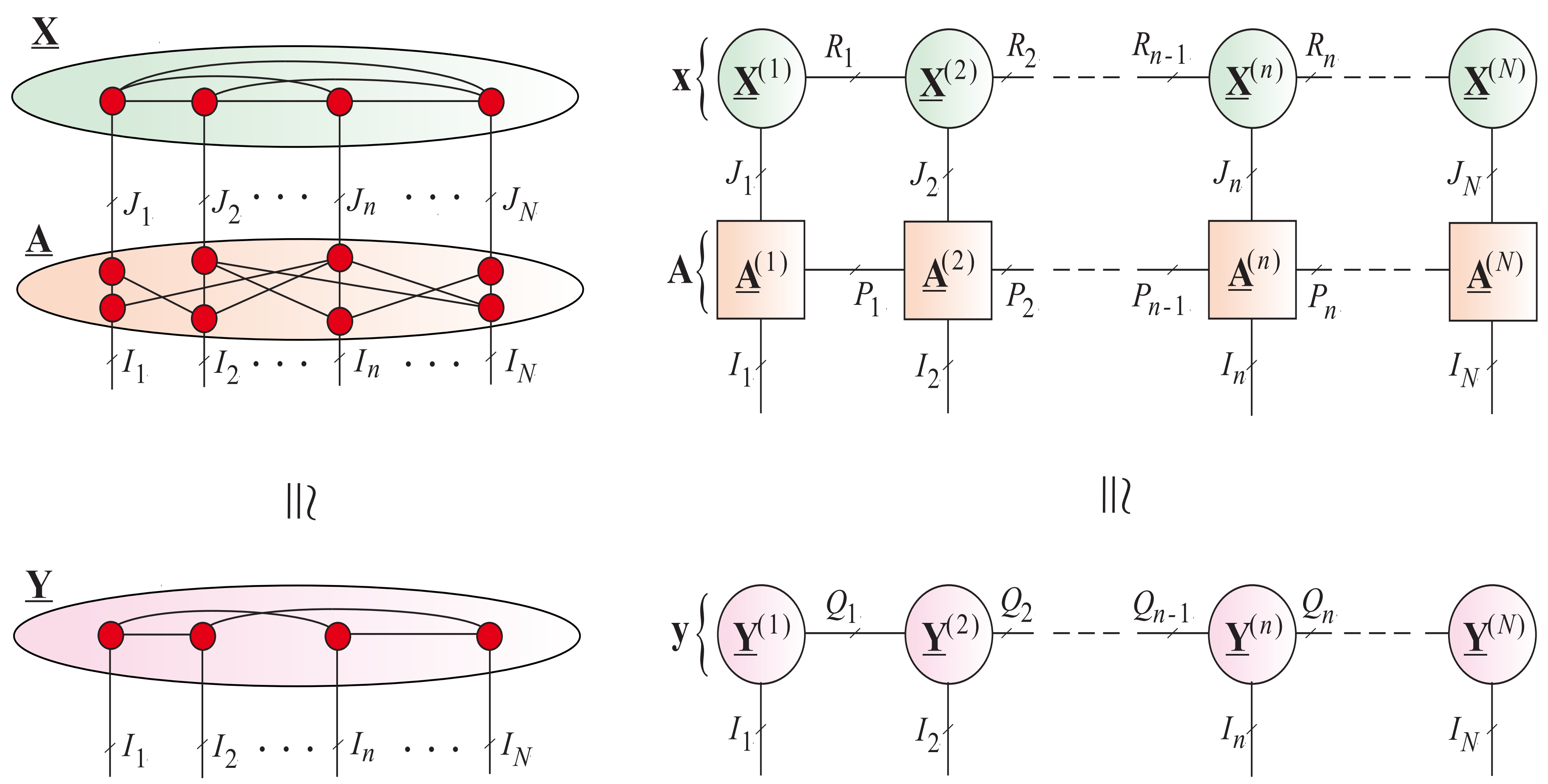
$$\underline{\mathbf{X}} = \sum_{r_1, r_2, \ldots, r_{N-1}=1}^{R_1, R_2, \ldots, R_{N-1}} \mathbf{x}^{(1)}_{r_1} \circ \mathbf{x}^{(2)}_{r_1, r_2} \circ \cdots \circ \mathbf{x}^{(N)}_{r_{N-1}}$$

$$\underline{\mathbf{Y}} = \sum_{q_1, q_2, \ldots, q_{N-1}=1}^{Q_1, Q_2, \ldots, Q_{N-1}} \mathbf{y}^{(1)}_{q_1} \circ \mathbf{y}^{(2)}_{q_1, q_2} \circ \cdots \circ \mathbf{y}^{(N)}_{q_{N-1}},$$

where $\mathbf{y}^{(n)}_{q_{n-1}, q_n} = \mathbf{y}^{(n)}_{\overline{r_{n-1}\, p_{n-1}},\, \overline{r_n\, p_n}} = \mathbf{A}^{(n)}_{p_{n-1},\, p_n} \mathbf{x}^{(n)}_{r_{n-1},\, r_n} \in \mathbb{R}^{I_n}$ with $Q_n = P_n\, R_n$

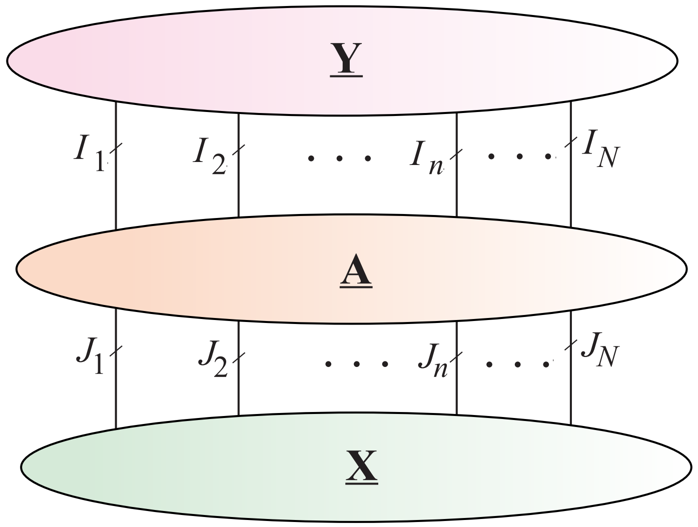
Operation in TT Format Cont

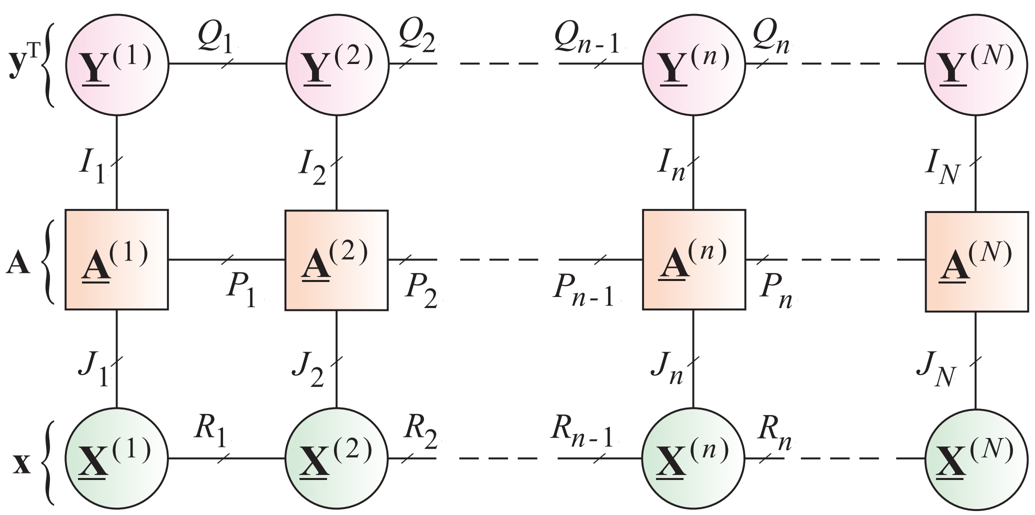
- Matrix-by-vector multiplication $\mathbf{Ax} = \mathbf{y}$ is represented by arbitrary TN and TT



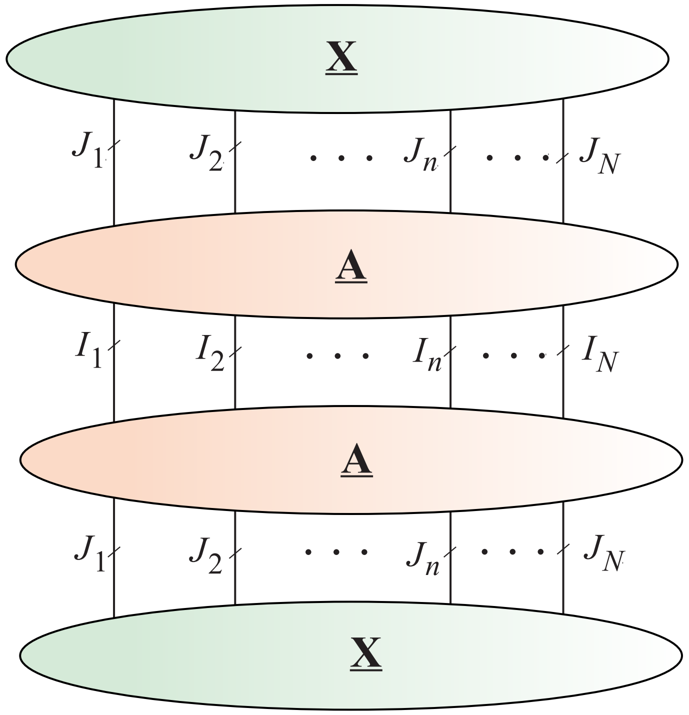
Operation in TT Format Cont

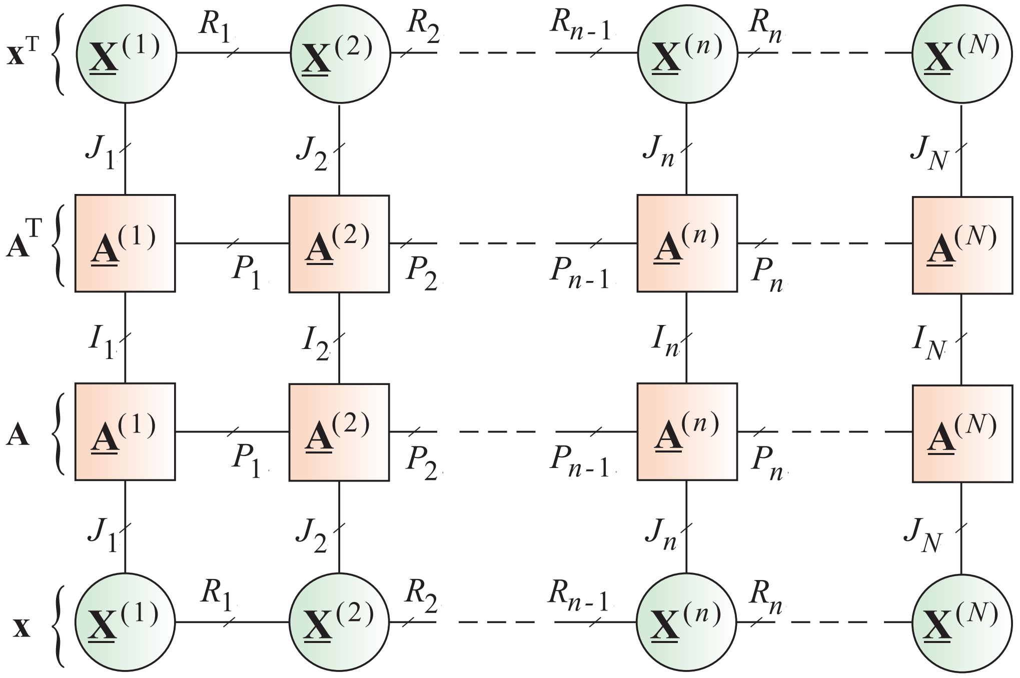
- Represent typical cost function $J_1(\mathbf{x}) = \mathbf{y}^{\mathrm{T}}\mathbf{A}\mathbf{x}$ by arbitrary TN and TT





- Represent another cost function $J_2(\mathbf{x}) = \mathbf{x}^{\mathrm{T}}\mathbf{A}^{\mathrm{T}}\mathbf{A}\mathbf{x}$ by arbitrary TN and TT





Computation of EVD in TT Format

- ML applications often require computation of extreme eigenvalues/eigenvectors of a large-scale symmetric matrix
- Standard eigenvalue decomposition (EVD) can be formulated as

$$\mathbf{A}\,\mathbf{x}_k = \lambda_k \mathbf{x}_k, \qquad k = 1, 2, \ldots, K$$

- Typical iterative solution for extreme EVD problem involves optimizing the Rayleigh quotient (RQ) cost function

$$J(\mathbf{x}) = R(\mathbf{x}, \mathbf{A}) = \frac{\mathbf{x}^{\mathrm{T}}\mathbf{A}\mathbf{x}}{\mathbf{x}^{\mathrm{T}}\mathbf{x}} = \frac{\langle \mathbf{A}\mathbf{x}, \mathbf{x} \rangle}{\langle \mathbf{x}, \mathbf{x} \rangle}$$

$$\lambda_{max} = \max_{\mathbf{x}} R(\mathbf{x}, \mathbf{A}), \quad \lambda_{min} = \min_{\mathbf{x}} R(\mathbf{x}, \mathbf{A})$$

- Traditional methods are prohibitive for very large-scale matrix $\mathbf{A} \in \mathbb{R}^{I \times I}$ say $I = 10^{15}$

Computation of EVD in TT Format Cont

- TN solution is to represent RQ cost function via low-rank TT format
- Thus a large EVD problem can be converted into a set of small EVD sub-problems by following steps:

i) Tensorize the matrix $\mathbf{A} \in \mathbb{R}^{I \times I}$ and eigenvector $\mathbf{x} \in \mathbb{R}^{I}$ and then represent them in matrix TT format and TT format, respectively

$$\underline{\mathbf{A}} \cong \langle\langle \underline{\mathbf{A}}^{(1)}, \ldots, \underline{\mathbf{A}}^{(N)} \rangle\rangle \in \mathbb{R}^{I_1 \times I_1 \times \cdots \times I_N \times I_N}$$

$$\underline{\mathbf{X}} \cong \langle\langle \underline{\mathbf{X}}^{(1)}, \ldots, \underline{\mathbf{X}}^{(N)} \rangle\rangle \in \mathbb{R}^{I_1 \times \cdots \times I_N}$$

ii) Reparametrize $\mathbf{x}$ by separating the mode-n TT core from rest TT cores using tensor contraction and frame equations

$$\mathbf{x} = \mathbf{X}_{\neq n} \, \mathbf{x}^{(n)}$$

with frame matrices $\mathbf{X}_{\neq n} = \mathbf{X}^{<n} \otimes_L \mathbf{I}_{I_n} \otimes_L (\mathbf{X}^{>n})^{\mathrm{T}} \in \mathbb{R}^{I_1 I_2 \cdots I_N \times R_{n-1} I_n R_n}$

Computation of EVD in TT Format Cont

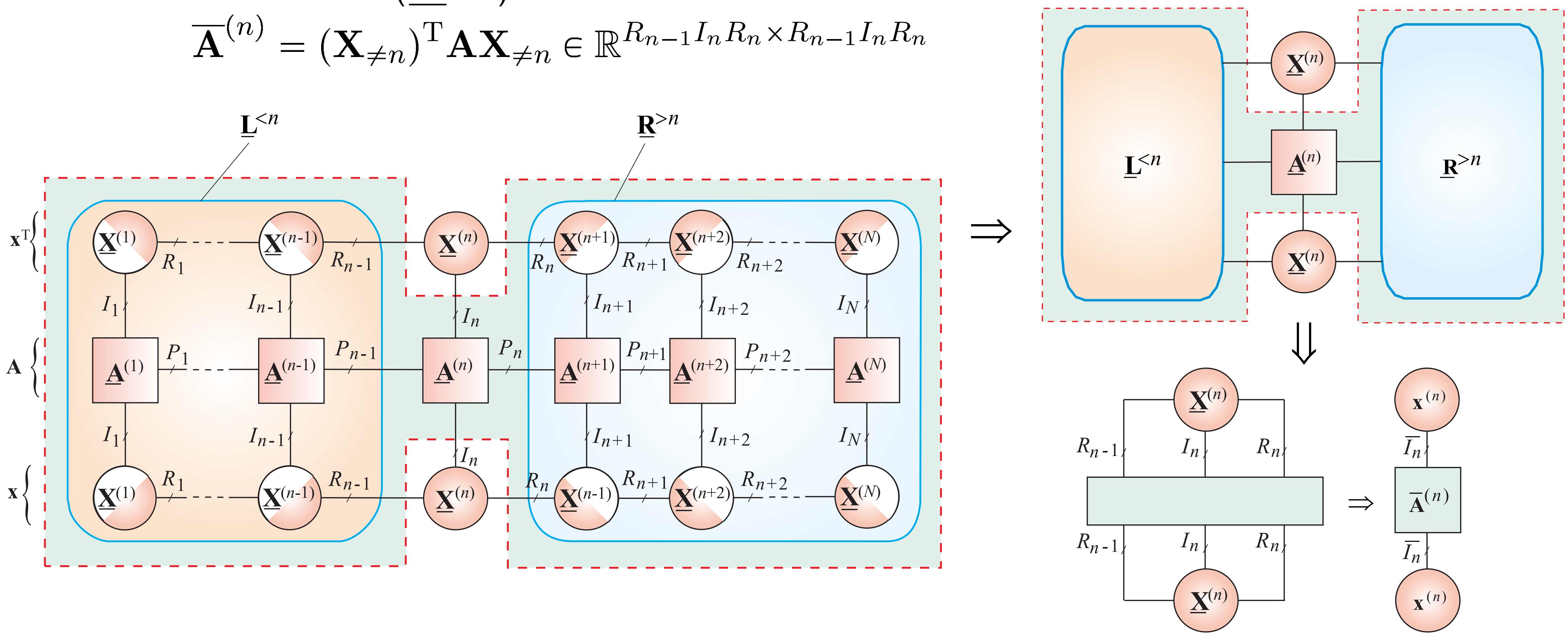
iii) Optimize a set of RQ functions of small matrices $\overline{\mathbf{A}}^{(n)}$ instead of optimizing the original RQ function of a large matrix $\mathbf{A}$

$$\min_{\mathbf{x}} J(\mathbf{x}) = \min_{\mathbf{x}^{(n)}} J(\mathbf{X}_{\neq n}\mathbf{x}^{(n)})$$

$$= \min_{\mathbf{x}^{(n)}} \frac{\mathbf{x}^{(n)\,\mathrm{T}}\,\overline{\mathbf{A}}^{(n)}\,\mathbf{x}^{(n)}}{\langle \mathbf{x}^{(n)}, \mathbf{x}^{(n)} \rangle}, \quad n = 1, 2, \ldots, N$$

where $\mathbf{x}^{(n)} = \mathrm{vec}(\underline{\mathbf{X}}^{(n)}) \in \mathbb{R}^{R_{n-1} I_n R_n}$

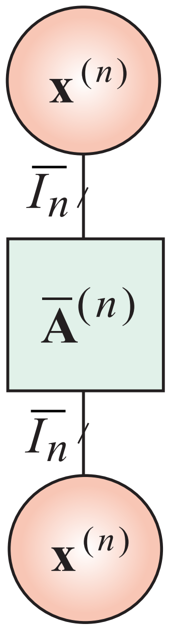
$\overline{\mathbf{A}}^{(n)} = (\mathbf{X}_{\neq n})^{\mathrm{T}} \mathbf{A} \mathbf{X}_{\neq n} \in \mathbb{R}^{R_{n-1} I_n R_n \times R_{n-1} I_n R_n}$



Computation of EVD in TT Format Cont

- In this way, matrices $\overline{\mathbf{A}}^{(n)}$ are usually much smaller than the original matrix $\mathbf{A}$, thus a large-scale EVD problem are converted into a set of much smaller EVD sub-problems

$$\overline{\mathbf{A}}^{(n)}\mathbf{x}^{(n)} = \lambda \mathbf{x}^{(n)}, \quad n = 1, 2, \ldots, N$$



Computation of SVD in TT Format

- Similar to EVD, TT formats can be applied to compute K largest singular values/vectors of a a large matrix $\mathbf{A} \in \mathbb{R}^{I \times J}$
- SVD can be solved by maximizing the following cost function as

$$J(\mathbf{U}, \mathbf{V}) = \mathrm{tr}(\mathbf{U}^{\mathrm{T}} \mathbf{A} \mathbf{V}), \quad \text{s.t.} \quad \mathbf{U}^{\mathrm{T}} \mathbf{U} = \mathbf{I}_K, \;\; \mathbf{V}^{\mathrm{T}} \mathbf{V} = \mathbf{I}_K$$

where $\mathbf{U} \in \mathbb{R}^{I \times K}$ and $\mathbf{V} \in \mathbb{R}^{J \times K}$

- Similarly, the key idea is to perform TT core contractions to reduce the unfeasible huge-scale optimization problem to small scale sub-problems as

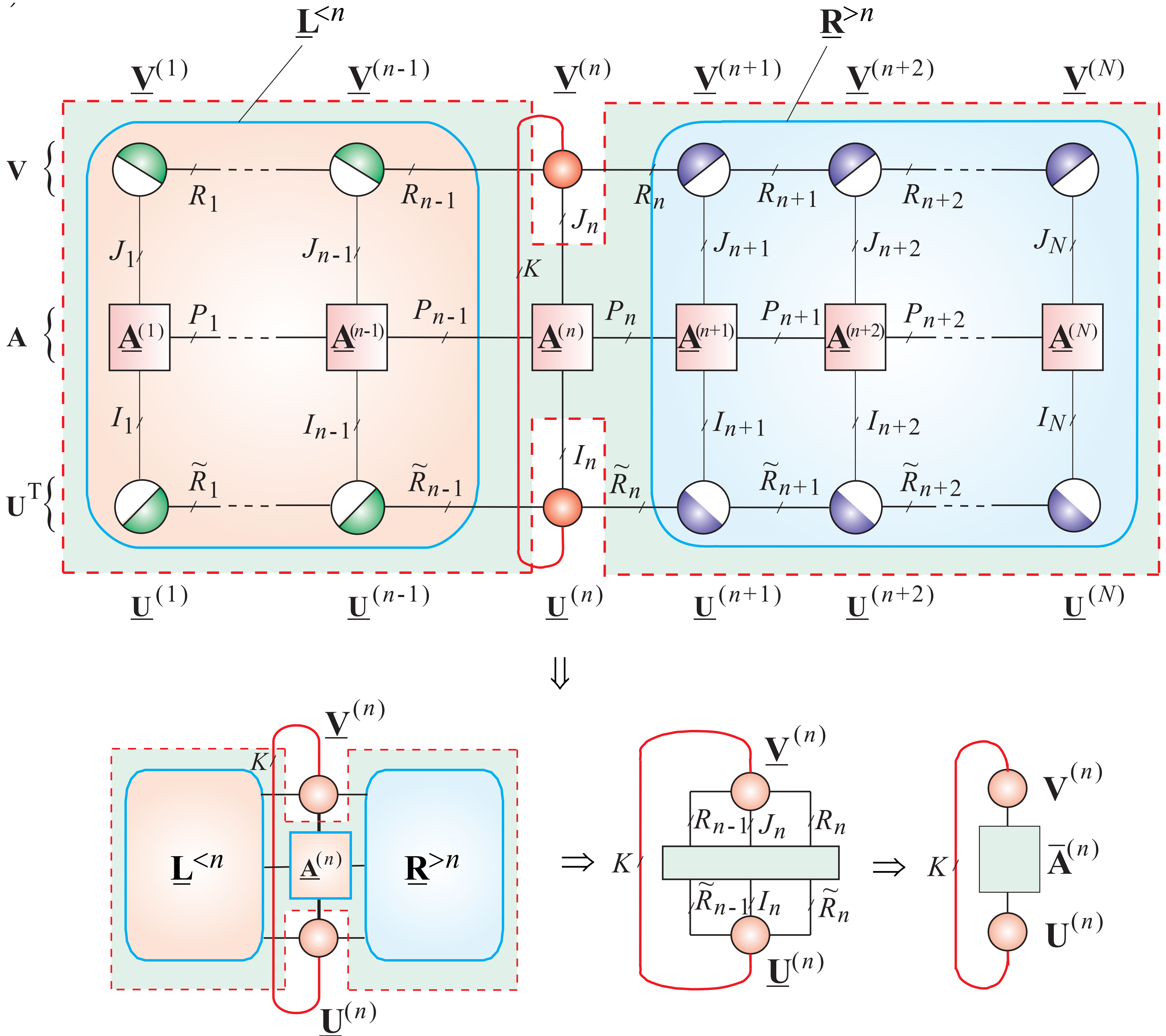
$$\max_{\mathbf{U}^{(n)}, \mathbf{V}^{(n)}} \mathrm{tr}((\mathbf{U}^{(n)})^{\mathrm{T}} \, \overline{\mathbf{A}}^{(n)} \, \mathbf{V}^{(n)}) \quad \text{s.t.} \;\; (\mathbf{U}^{(n)})^{\mathrm{T}} \mathbf{U}^{(n)} = \mathbf{I}_K, \quad (\mathbf{V}^{(n)})^{\mathrm{T}} \mathbf{V}^{(n)} = \mathbf{I}_K$$

where $\mathbf{U}^{(n)} \in \mathbb{R}^{\tilde{R}_{n-1} I_n \tilde{R}_n \times K}$ and $\mathbf{V}^{(n)} \in \mathbb{R}^{R_{n-1} J_n R_n \times K}$

$$\overline{\mathbf{A}}^{(n)} = \mathbf{U}_{\neq n}^{\mathrm{T}} \mathbf{A} \mathbf{V}_{\neq n} \in \mathbb{R}^{\tilde{R}_{n-1} I_n \tilde{R}_n \, \times \, R_{n-1} J_n R_n}$$

Computation of SVD in TT Format Cont

- In this way, the contracted matrices $\overline{\mathbf{A}}^{(n)}$ are much smaller than original matrix $\mathbf{A}$, thus any efficient SVD algorithms can be applied to $\overline{\mathbf{A}}^{(n)}$



Conclusion

- We provide an example-rich guide to the basic properties of TNs
- TN is demonstrated as a promising tool for analyzing extremely-large multidimensional data
- TN can be naturally employed for dimensionality reduction due to their intrinsic compression ability stemming from sparsely distributed representation
- TN is advantageous over matrix-based analysis methods with ability to model strong and weak coupling among multiple models
- TN can serve as a useful fundamental tool to solve a variety of machine learning problems where data has prohibitively large volume, variety and veracity

Question?