Adaptive SP & Machine Intelligence
Multi-way Analysis of Big Data

Danilo Mandic

room 813, ext: 46271

Department of Electrical and Electronic Engineering
Imperial College London, UK

d.mandic@imperial.ac.uk, URL: www.commsp.ee.ic.ac.uk/∼mandic

Outline

- Challenges in Big Data analytics
- Big Data and Machine Intelligence
- Data structures: From a scalar to a tensor
- Some basic operations on tensors
- Tensorisation $\looparrowright$ a key step in tensor decompositions
- Canonical Polyadic Decomposition (CPD) and its applications
- Links between the CPD and Tucker decomposition
- Partial Least Squares (PLS) and Higher-Order PLS (HOPLS)
- Applications

Big data processing $\looparrowright$ current status

- Computers excel at algorithmic tasks (well-posed mathematical problems)
- Biological systems are superior to digital systems for ill-posed problems with noisy data
- Pigeon: $\sim 10^9$ neurons, cycle time $\sim$ 0.1 seconds. Each neuron sends 2 bits to $\sim$ 1,000 other neurons. This is equivalent to 2×10^{13} bit operations per second
- Old PC: $\sim 10^7$ gates, cycle time 10^{-7} seconds, connectivity $= 2$ $\looparrowright 10^{15}$ bit operations per second
- Both have similar raw processing capability, but pigeons are better at recognition tasks
- Is there a way to present large date streams to computers in a more physically meaningful manner?

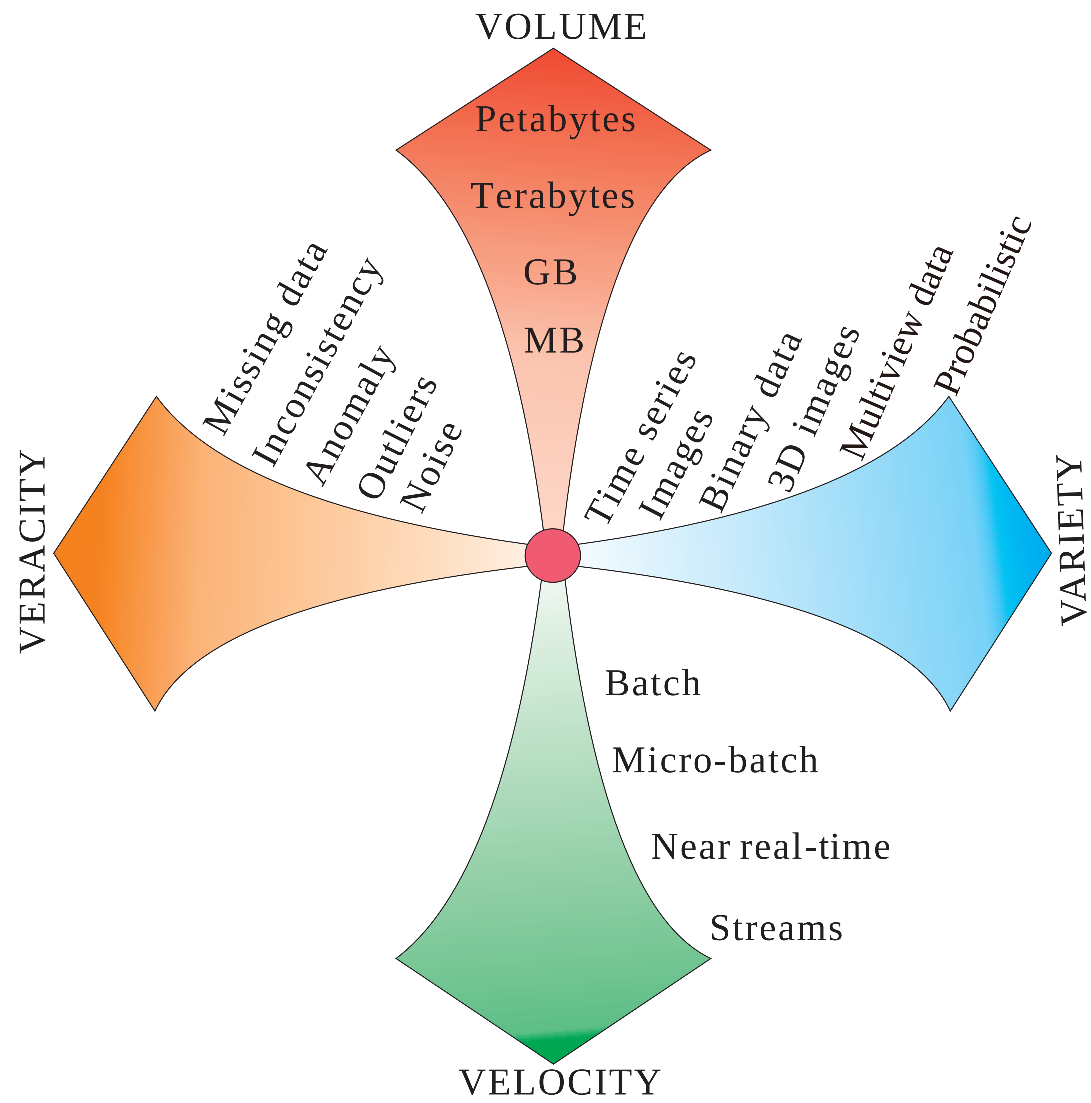
Some facts about Big Data opportunities

According to "Big Data: The next frontier for innovation, competition, and productivity", published by McKinsey Global Institute in May 2011:

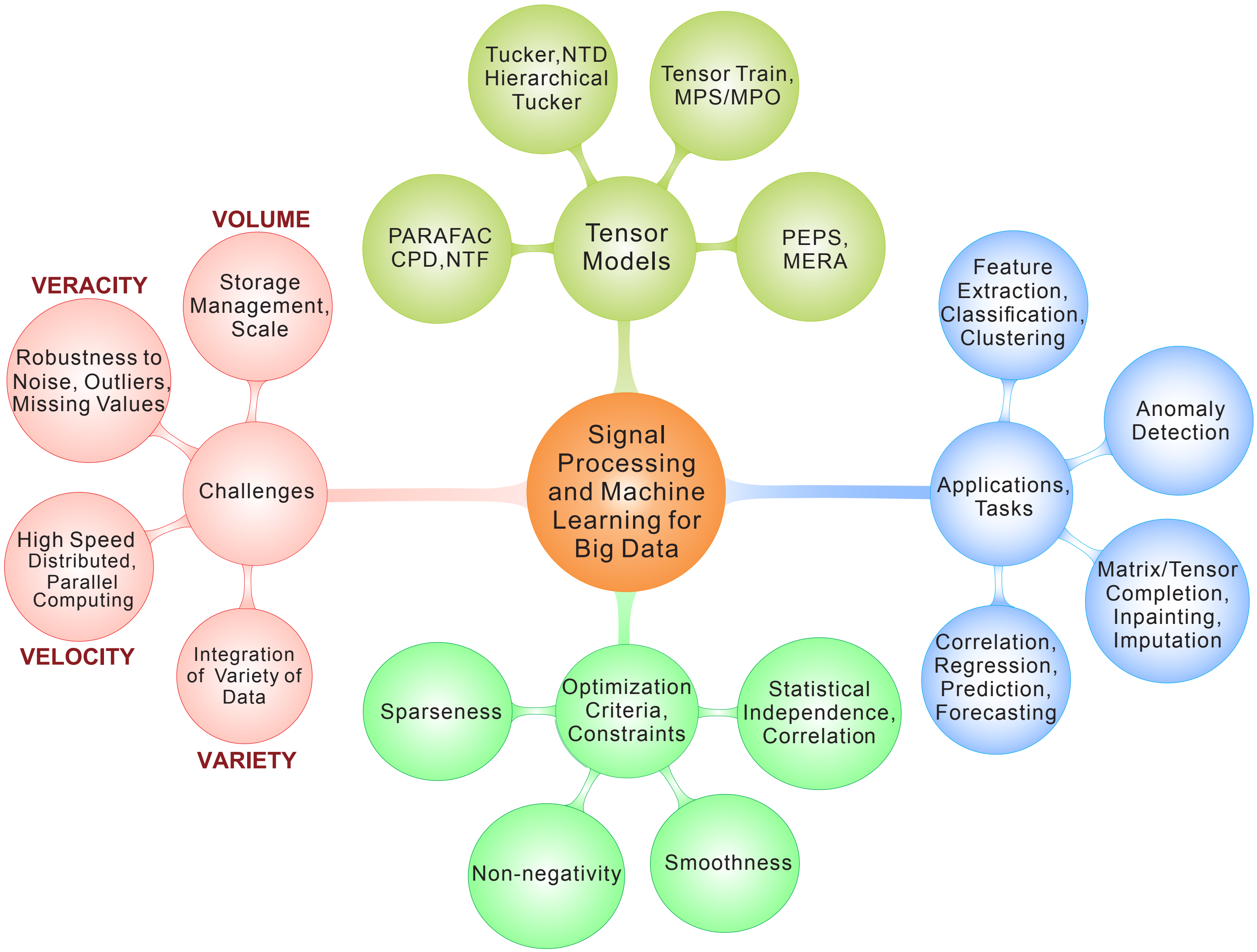
- It would cost USD 600 to by a disk drive which can store all off the music in the world
- In 2010, there were 4 billion mobile phone users in the world
- There is more than 30 billion pieces of content shared on social networks every month
- There is a predicted 40 % growth in global data generated per year versus a 5 % growth in global IT spending
- This all tells us that there are big opportunities for us working in Adaptive Signal Processing and Machine Intelligence

The four V's of big data: Volume, Variety, Velocity, Veracity

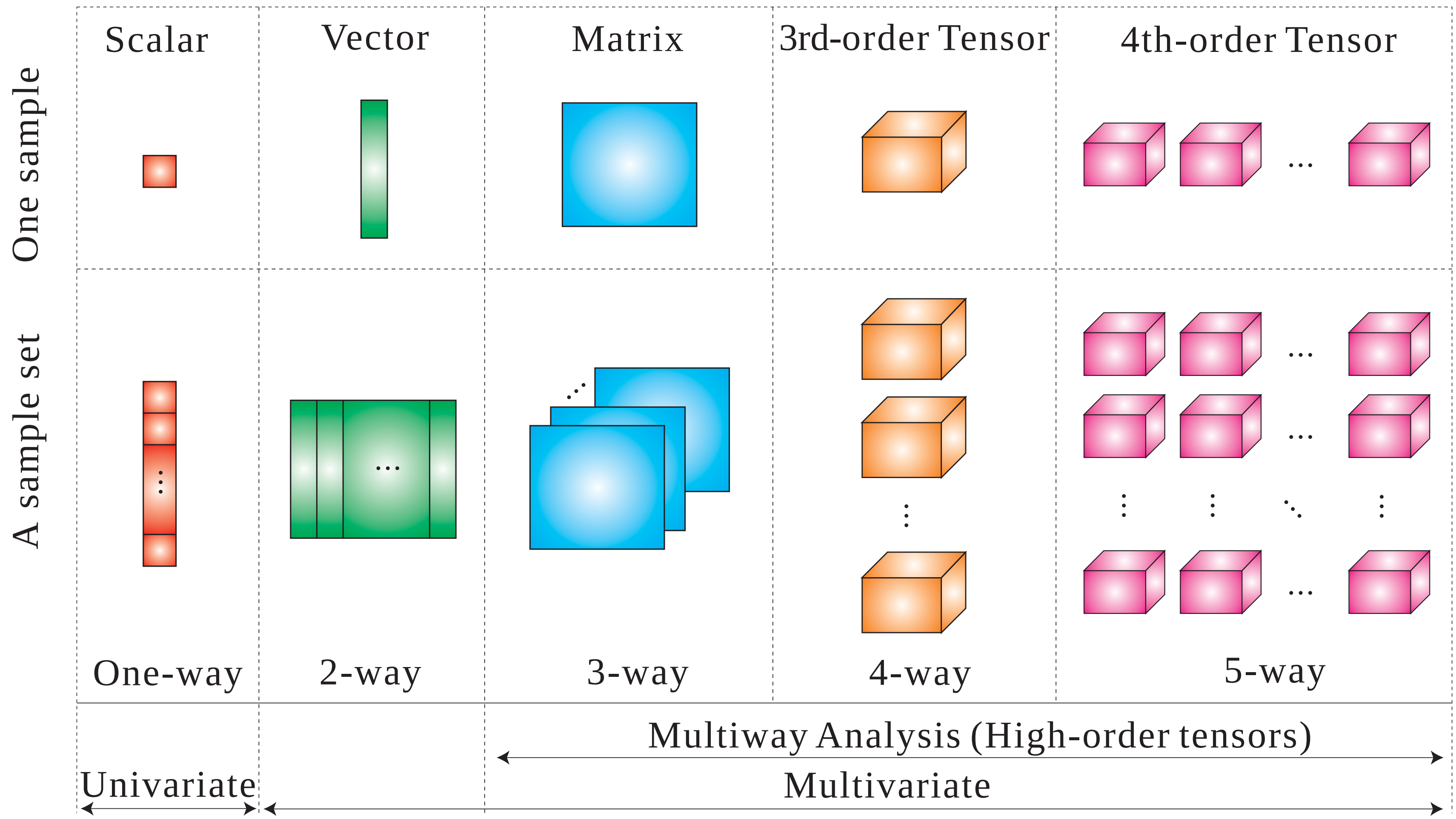
Other V's may include Visualisation, Variability, Value (quality of data), ...



Signal processing an machine learning for big data
Challenges and opportunities



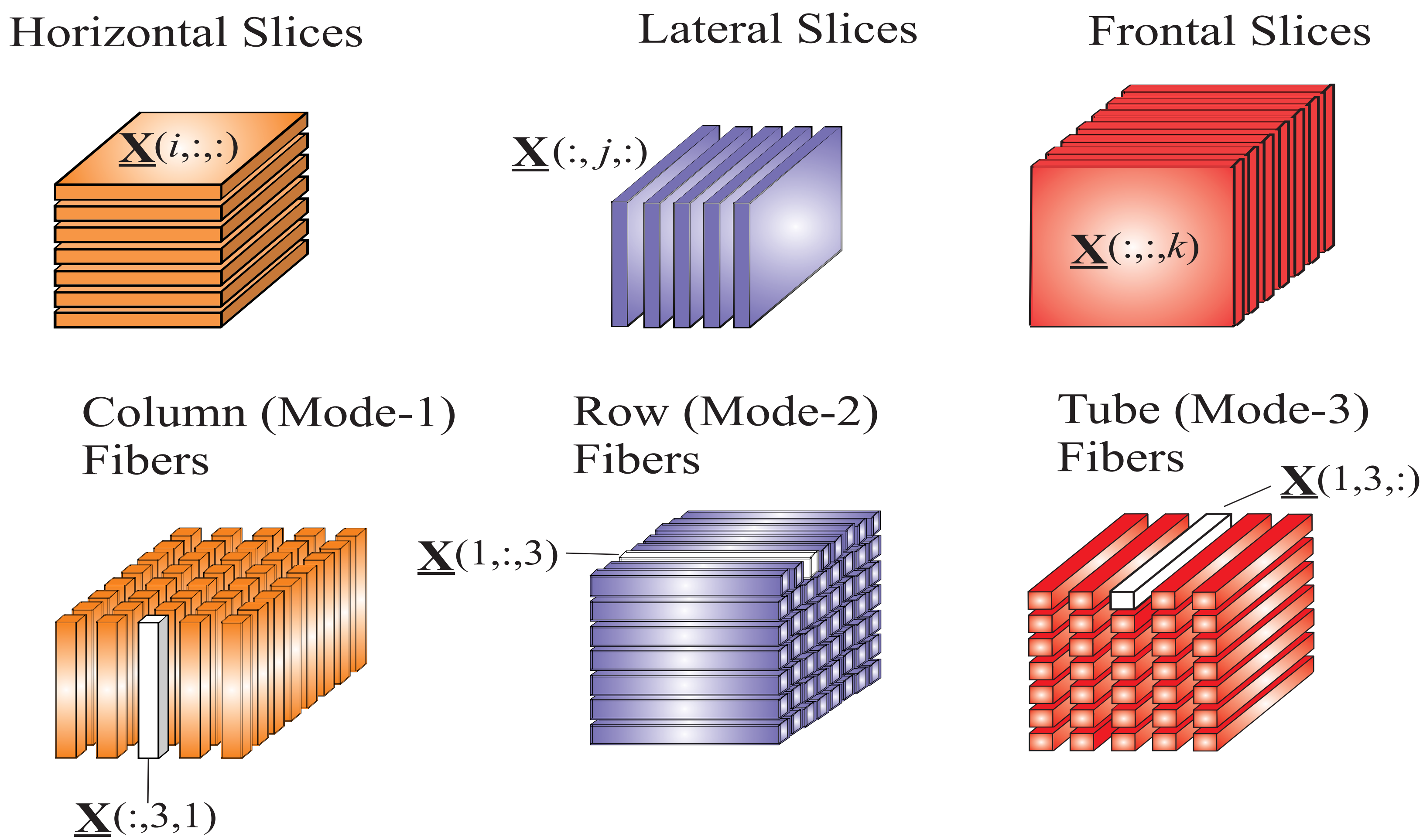
Types of data: From a scalar to a tensor



For example, a 4th-order tensor is a vector of 3rd-order tensors (top right)

Sub-structures within tensors

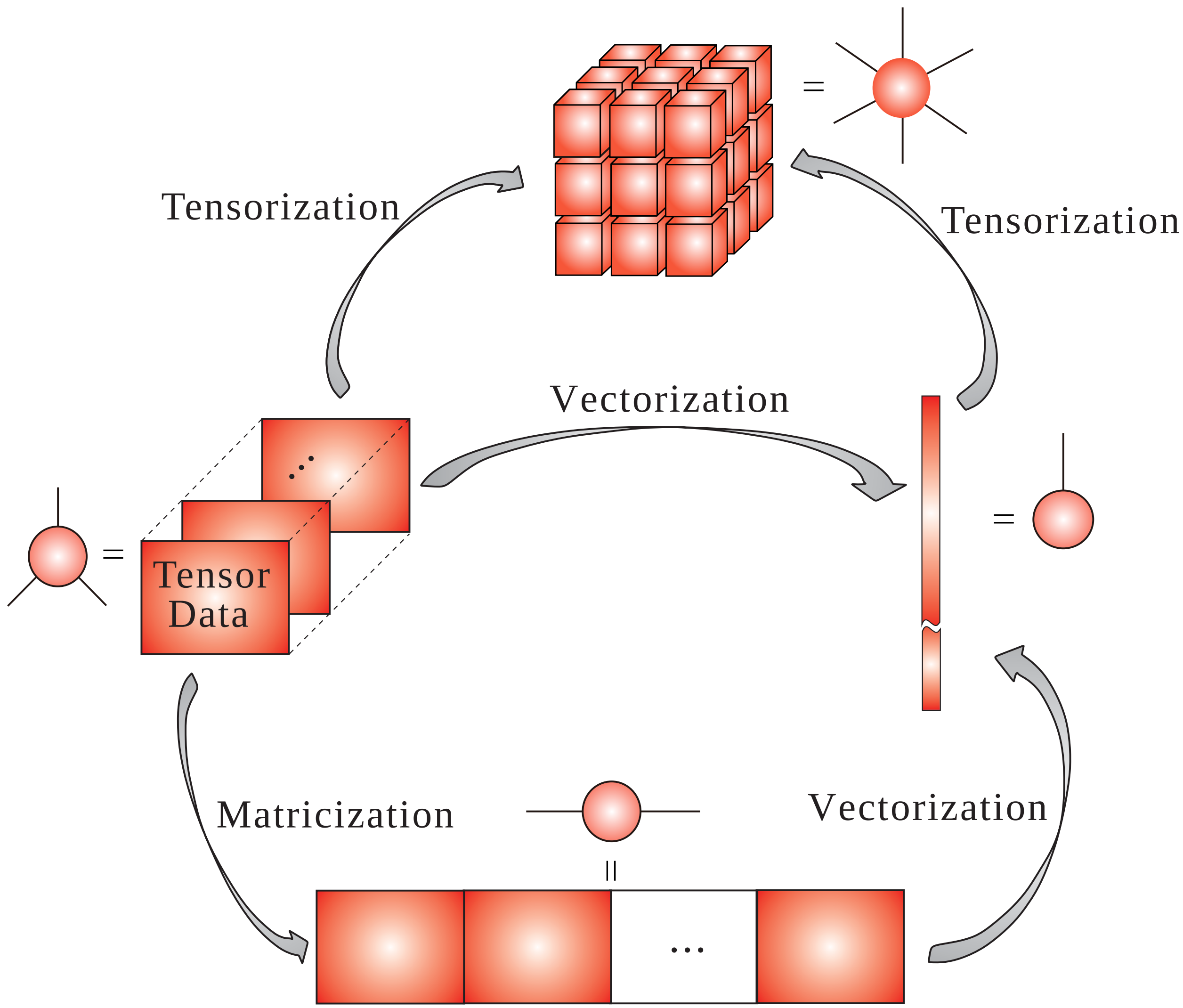
order-1 tensor = a vector   order-2 tensor = a matrix   dimensions = modes



☞ a fiber is produced by fixing two indices and varying one, e.g. $\underline{\mathbf{X}}(1,3,:)$

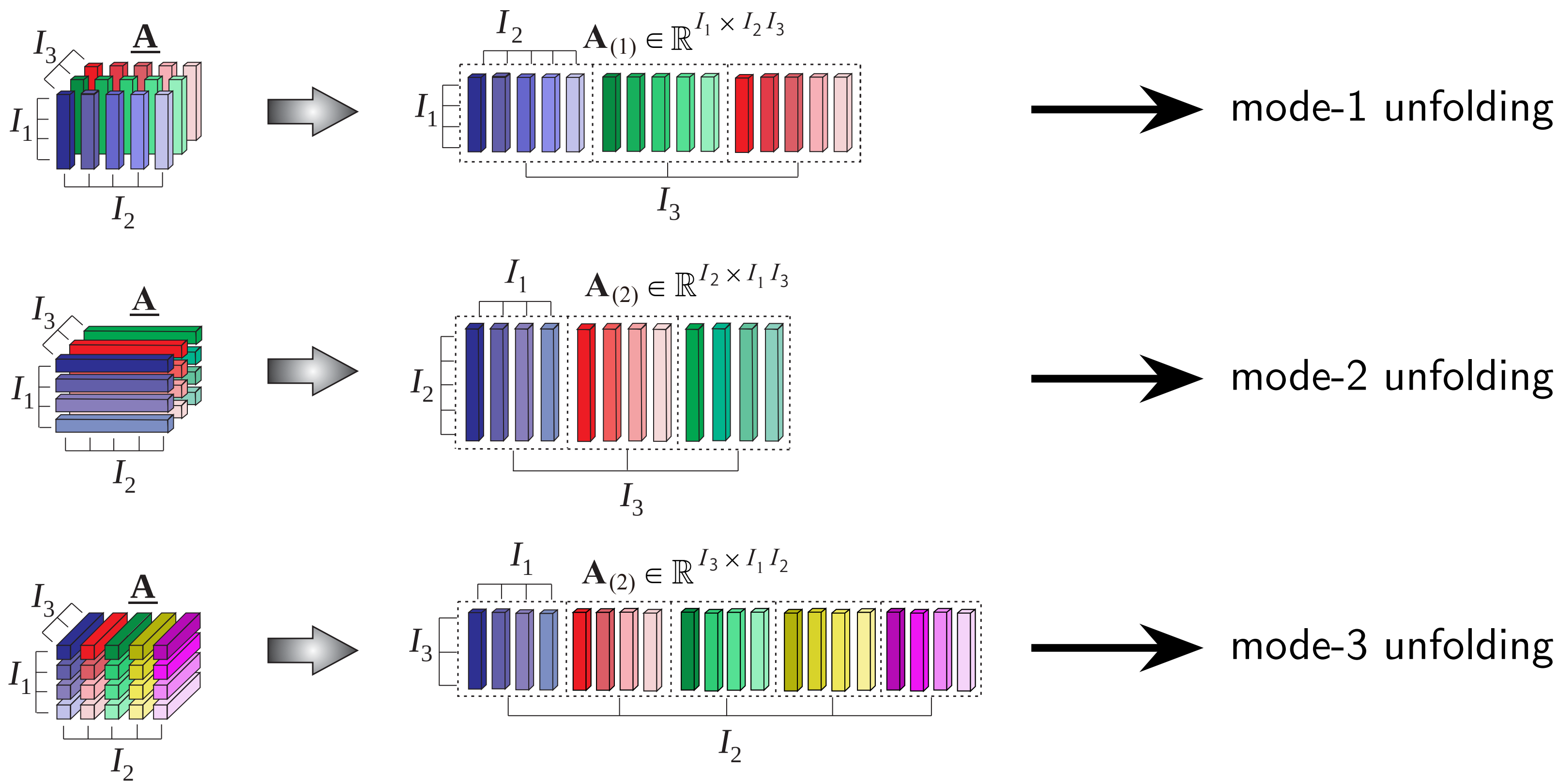
Reshaping of data structures: General concept

Vector, matrix or small-scale tensor $\looparrowright$ higher-order tensor is referred to as **folding**



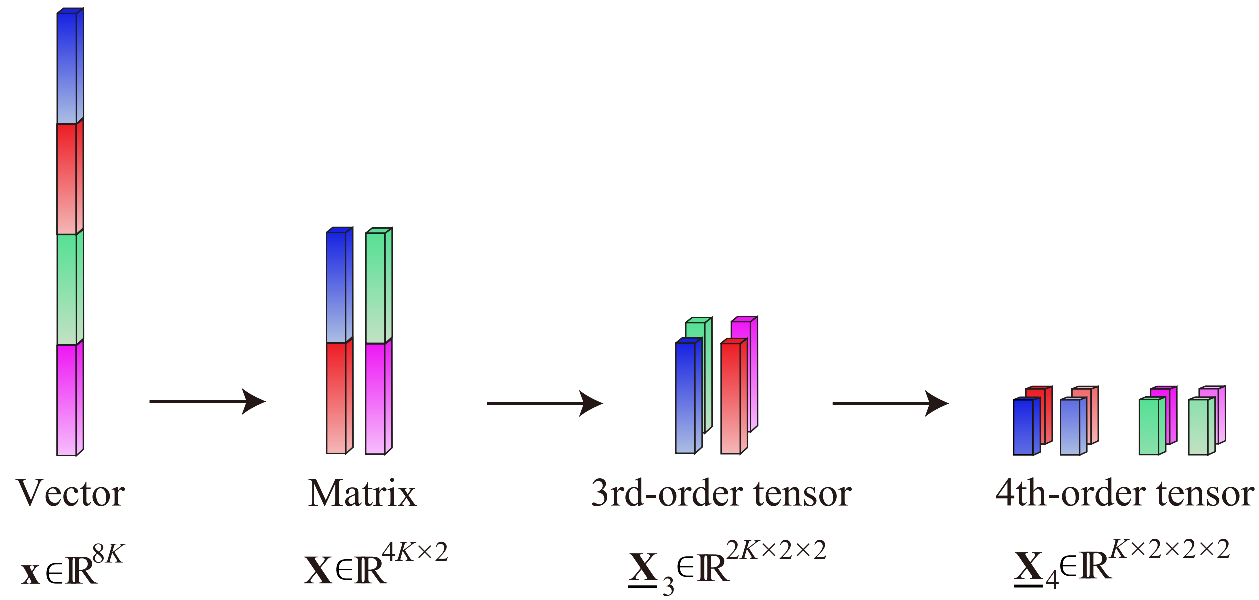
Unfolding of a tensor in different modes

Converts a higher-order tensor into a smaller tensor, matrix, or vector



- This operation maps tensor entries into a matrix, in e.g. a 'slice-by-slice' manner
- Such flattening (unfolding) prior to data analysis breaks the inherent structure in data and obscures latent dependencies between the modes

Tensorisation $\looparrowright$ blessing of dimensionality



Tensorization (creation of a tensor from a vector of a matrix) can be performed through:

- **Re-arrangement of lower-dimensional data.** One-way exponential sig. $x(k) = az^k$ can be folded into a rank-1 Hankel matrix, thus introducing redundancy (Slide 20)
- **Mathematical construction**. Through e.g. time x frequency x channel representation
- **Experimental design.** EEG data over I channels, J subjects, K trials (Slides 16-19)
- **Natural tensor data.** In HDTV, RGB color images are generated as 3rd-order tensors of size $1920 \times 1080 \times 3$. Similar situation exists in hyperspectral imaging (Slide 42)

Example 1: From a matrix to a 3D array
Example of a video clip

Each frame is 1,000 pixels by 1,000 pixels $\mathbf{X}_i \in \mathbb{R}^{1,000\times 1,000}$

20 seconds of recording with 50 FPS rate = 1,000 frames

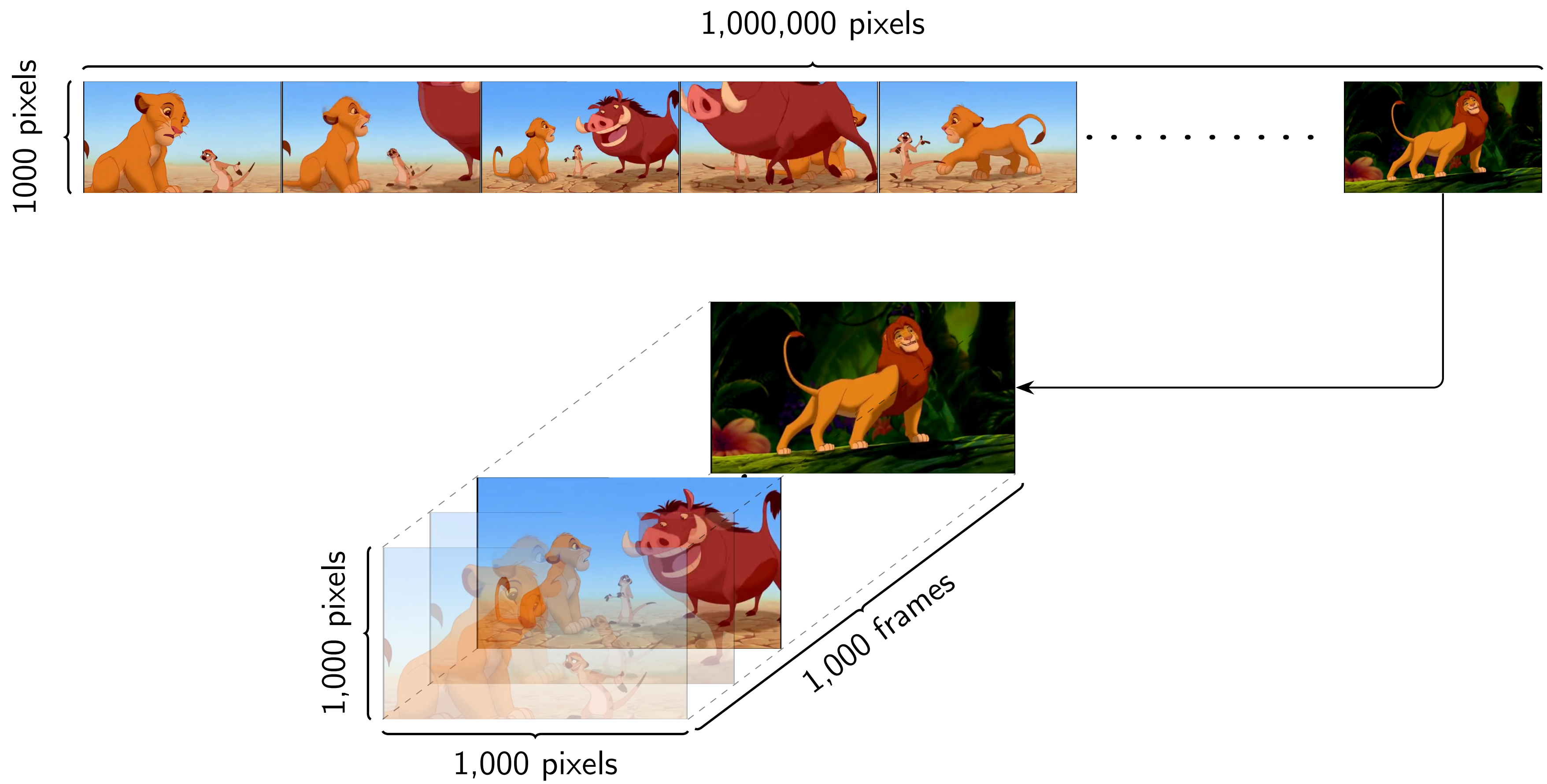
A video clip can be seen as a short & wide matrix $\mathbf{X} \in \mathbb{R}^{1,000\times 1,000,000}$

Analysis of all frames at once in this way is not informative or compact

- Significant difference in dimensions $\looparrowright$ processing is computationally expensive, difficult and not physically intuitive
- Any PCA-type solution would require a matrix of size $10^6 \times 10^6$
- This is a perfect scenario for low-rank tensor approximations and the inherent super-compression capability of tensor representations

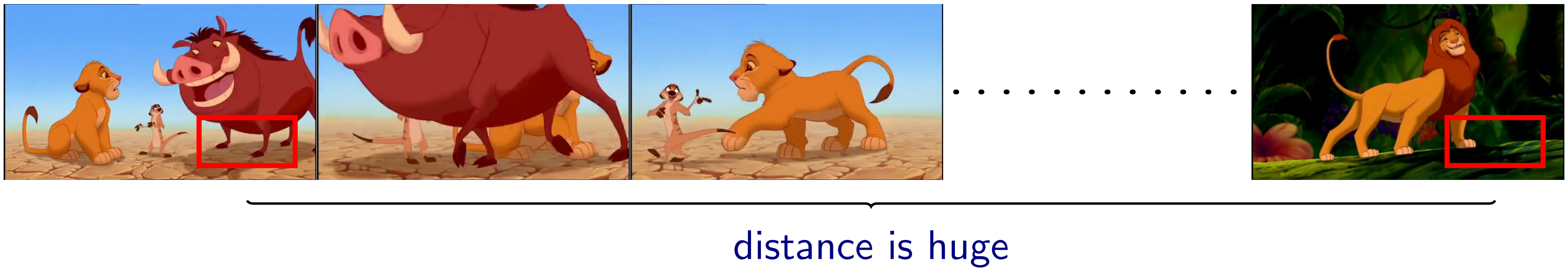
☞ Reshape this awkward-to-analyse data into a compact 3D array

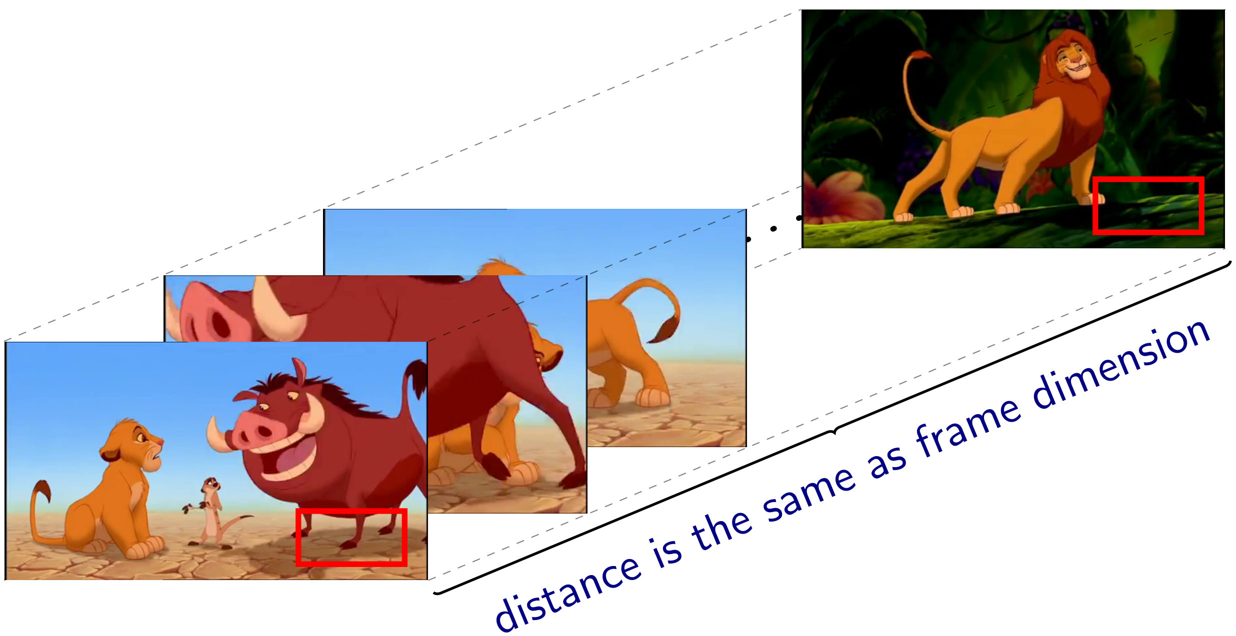
Example 1: Video clip $\hookrightarrow$ tensor construction



- A simple re-arrangement into a cube transforms the 1,000 $\times$ 1,000,000 matrix of frames into a 3-way tensor of size 1,000 $\times$ 1,000 $\times$ 1,000

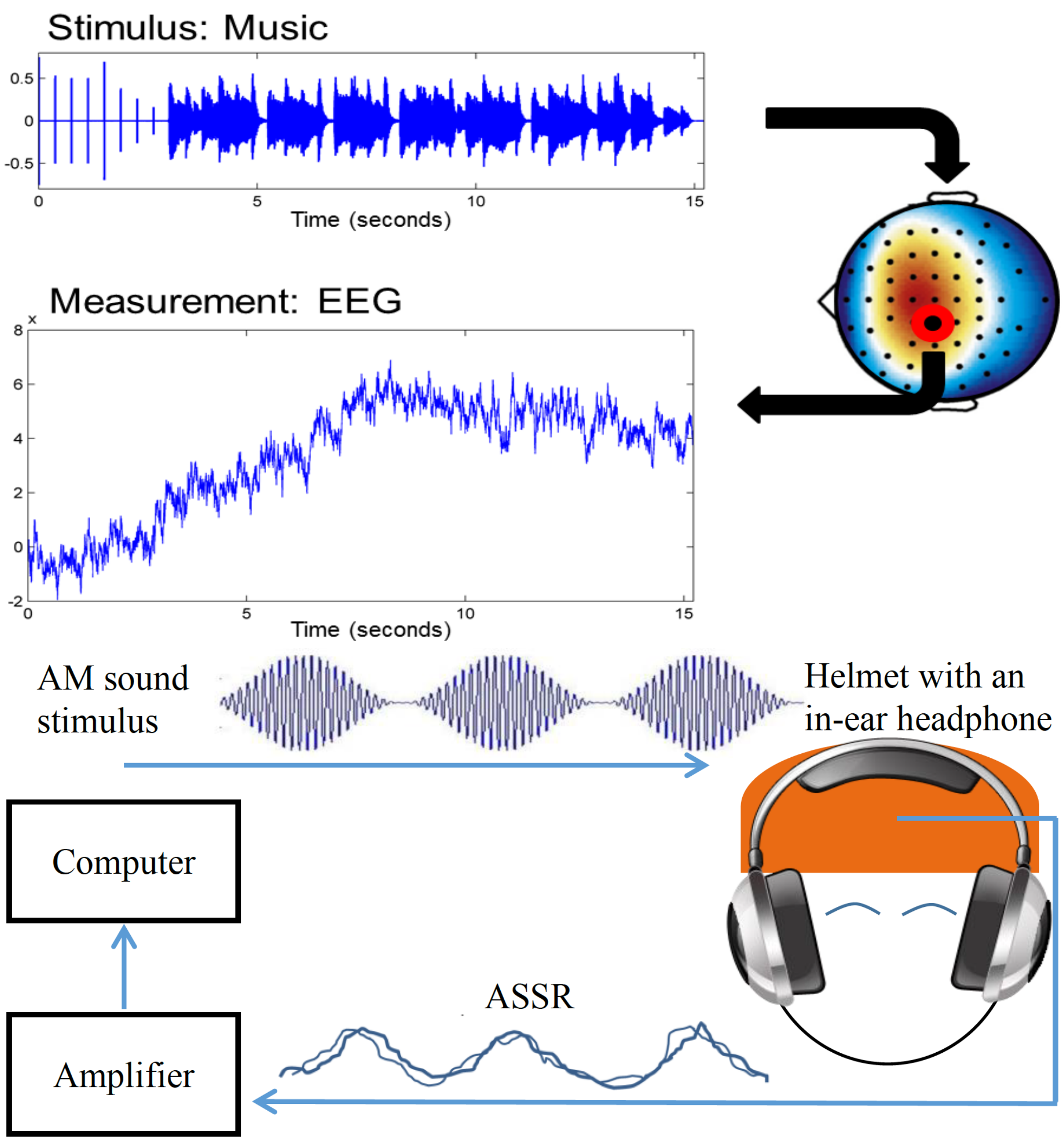
Example 1: Video clip $\looparrowright$ compact tensor representation





As compared to a matrix case (top row), we have the same number of data points $1000 \times 1000 \times 1000$, but arranged in a much more informative way. This provides a more intuitive and compact data representation and better statistical inference.

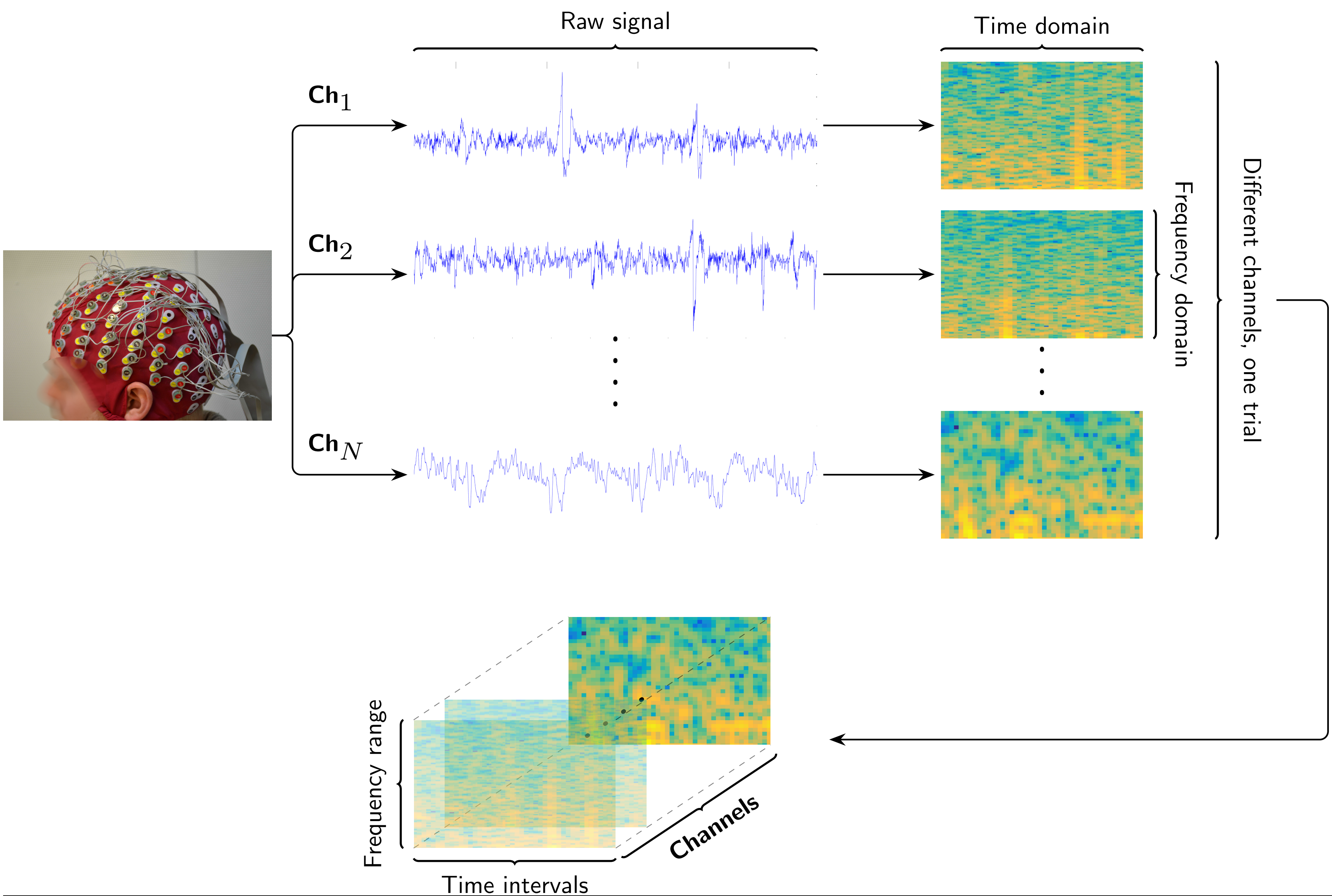
Tensorisation: Multi-way representation of multichannel biomedical data



- The electroencephalogram (EEG) is one of the fundamental tools for functional brain imaging, as it is non-invasive and has high temporal resolution
- Brain signals contain latent features which are much more likely to be found from recordings across a large number of recording channels, multiple trials, multiple subjects, multiple stimuli, ...
- The EEG recordings are therefore inherently multi-dimensional (many channels), and multi-way

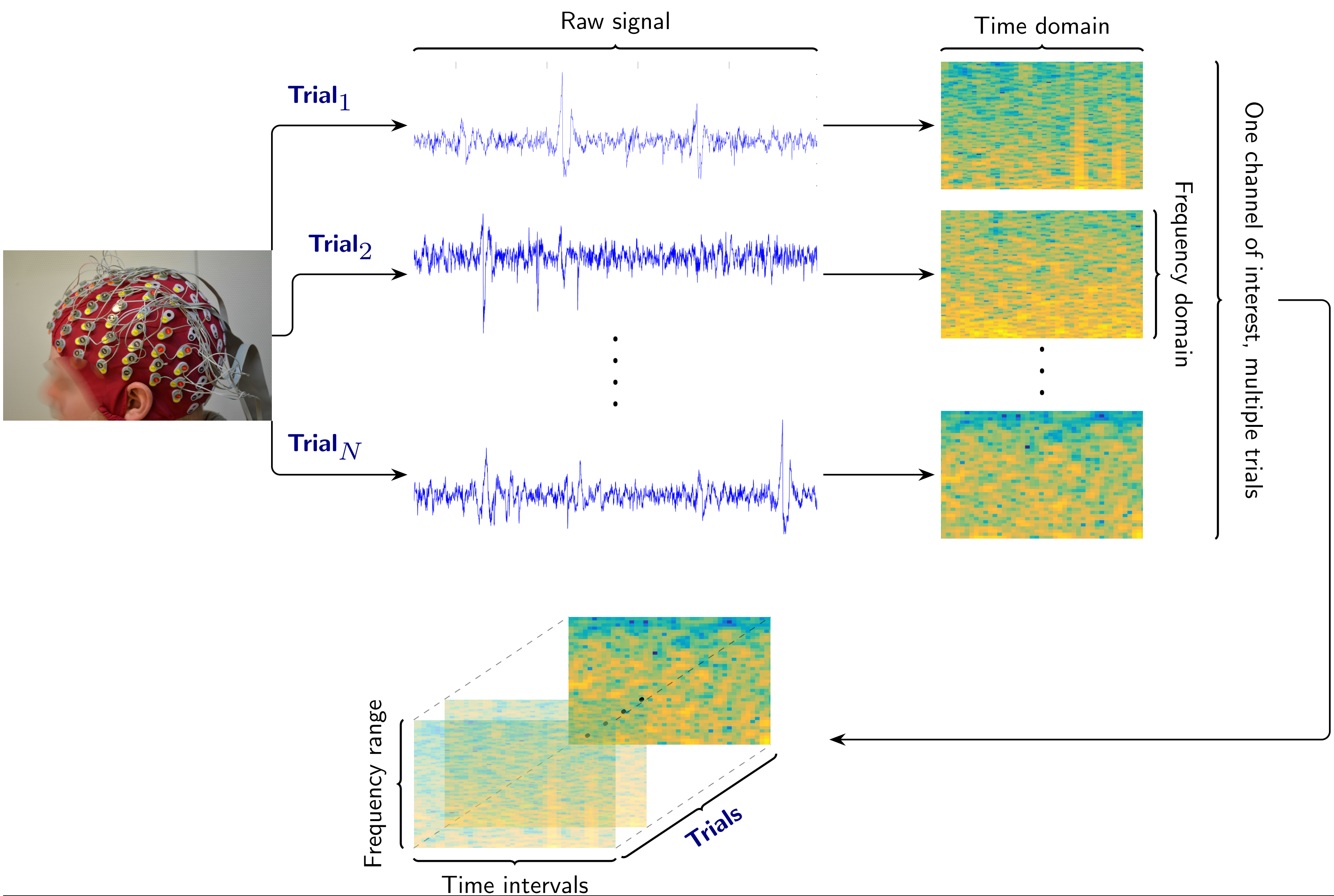
Example 2a: Tensor construction from different channels

↬ channel × frequency × time



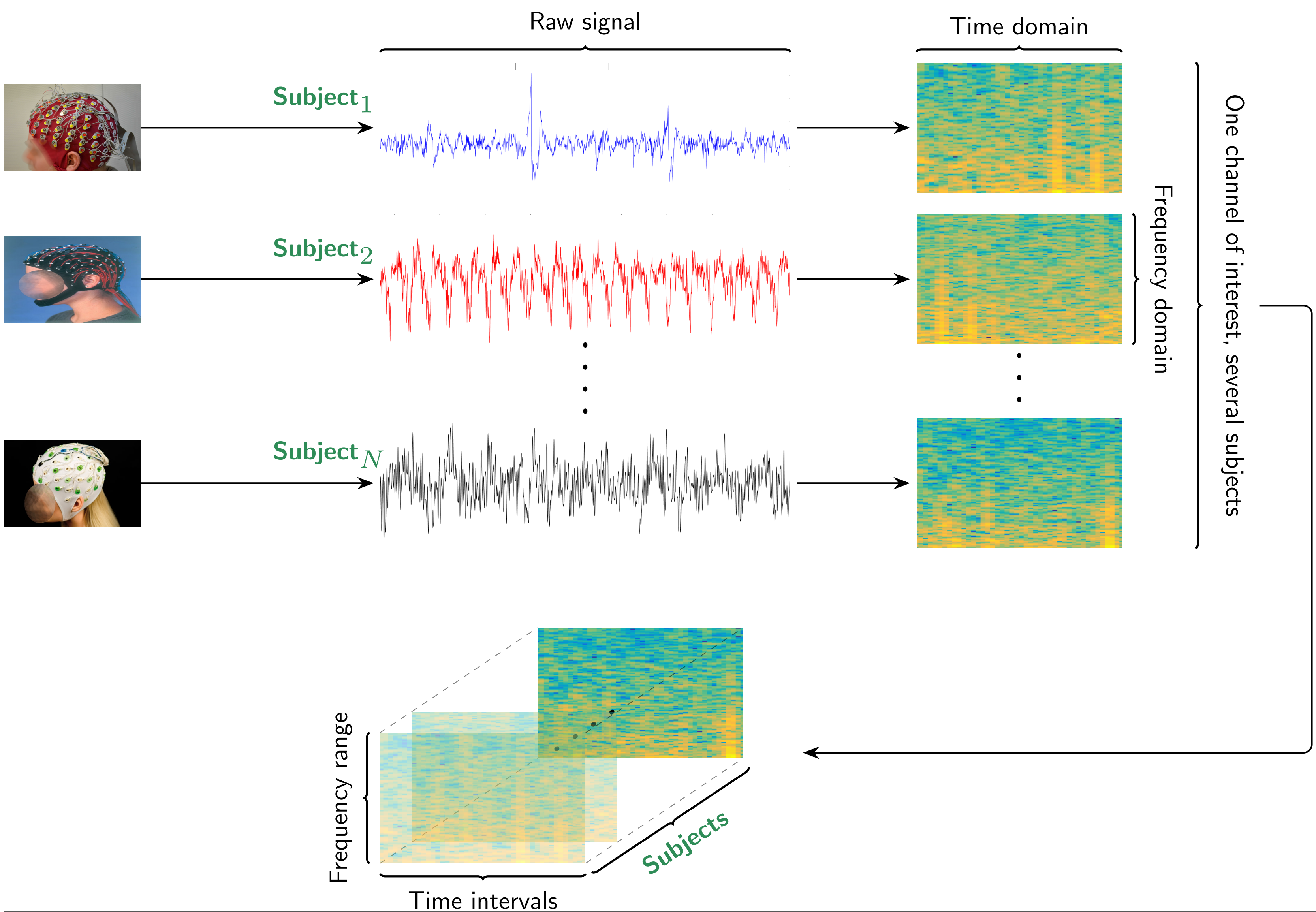
Example 2b: Tensor construction from different trials

↬ trial × frequency × time

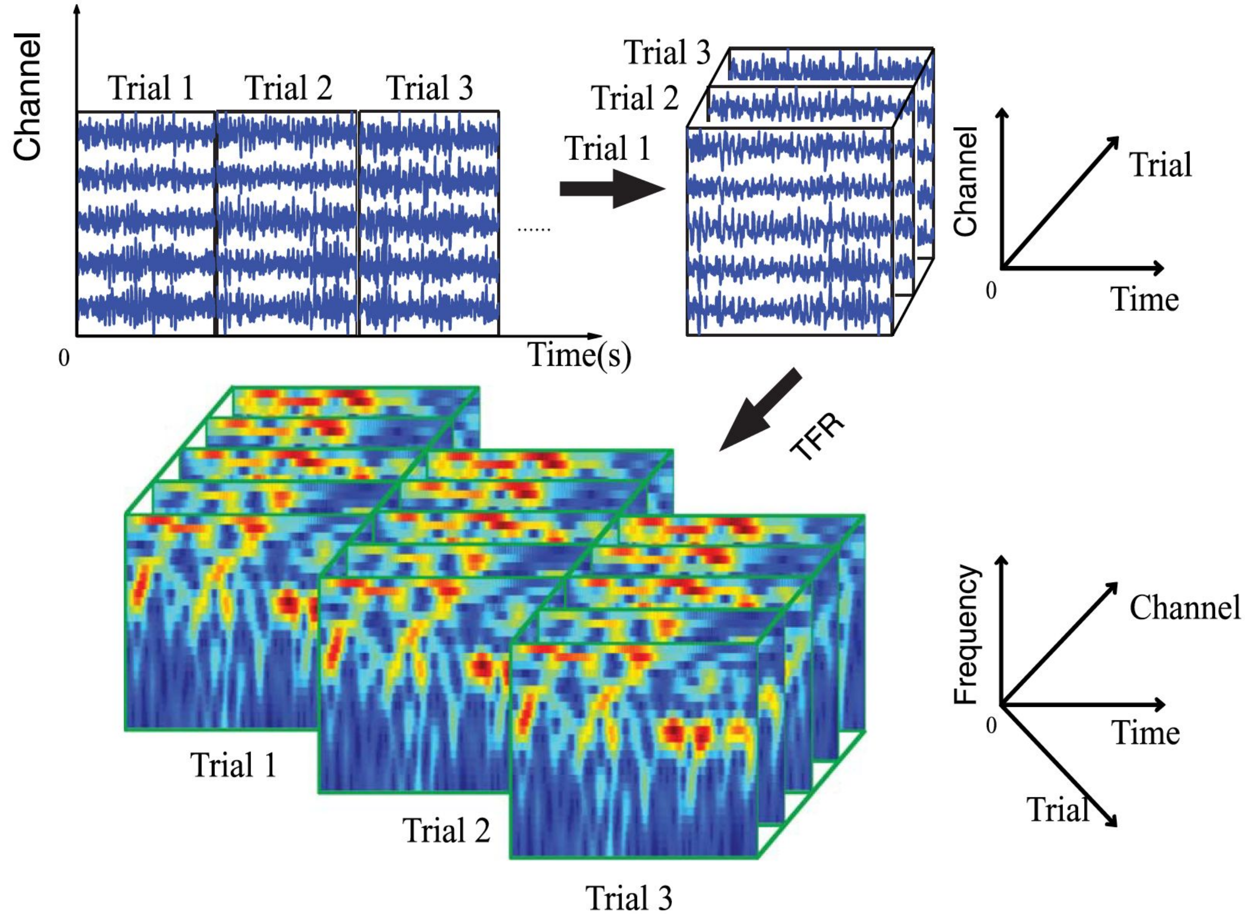


Example 2c: Tensor construction from different subjects

↬ subject × frequency × time

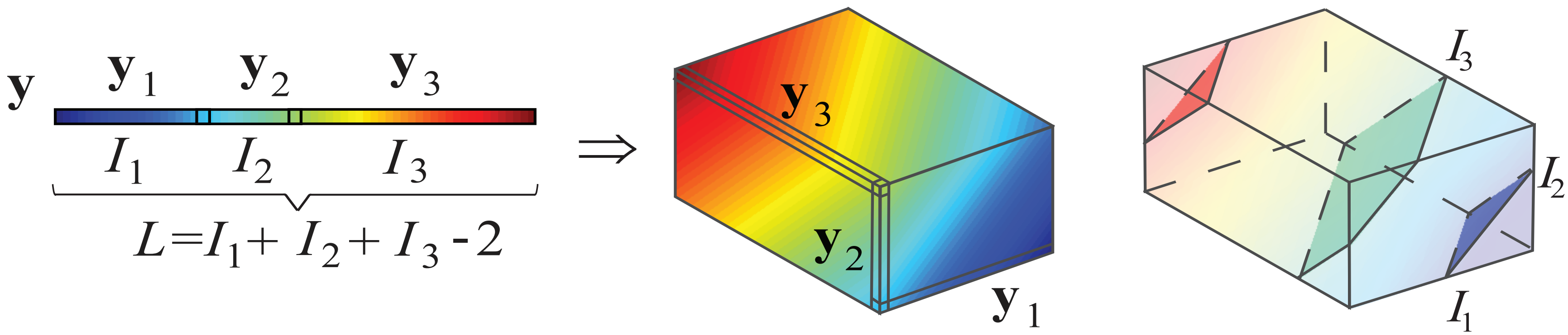


Example 2d: Putting it all together, construction of a 4D tensor with modes channel $\times$ trial $\times$ frequency $\times$ time



- Each data channel is a matrix of channels $\times$ time. Multiple trials form a 3D array
- Time frequency representation (TFR) yields a 4D multi-way array of data. If we include the # Subject, then we have a 5th-order tensor, and so on

Deterministic folding techniques for structured data: The Hankel folding operator



- Consider a sampled exponential signal $\mathbf{z}[k] = az^k$, which produces a data stream

$$\begin{bmatrix} a & az & az^2 & az^3 & \cdots \end{bmatrix} \tag{1}$$

- It can be re-arranged into a Hankel matrix, $\mathbf{H}$, of rank-1 as follows:

$$\mathbf{H} = \begin{bmatrix} a & az & az^2 & \cdots \\ az & az^2 & az^3 & \cdots \\ az^2 & az^3 & az^4 & \cdots \\ \vdots & \vdots & \vdots & \end{bmatrix} = a \begin{bmatrix} 1 \\ z \\ z^2 \\ \vdots \end{bmatrix} \begin{bmatrix} 1 & z & z^2 & \cdots \end{bmatrix} = a\, \mathbf{z} \circ \mathbf{z} \tag{2}$$

- For multivariate data, each data channel, i, can be mapped into a Hankel matrix, $\mathbf{H}_i$
- These channel-wise Hankel matrices can then be stacked together into a tensor $\underline{\mathbf{H}}$

Deterministic folding techniques for structured data: The Toeplitz folding operator

- Consider the discrete convolution of two vectors, $\mathbf{x}$ and $\mathbf{y}$, of respective lengths I and $L > I$, given by

$$\mathbf{z} = \mathbf{x} * \mathbf{y} \tag{3}$$

- The entries $\mathbf{z}_{I:L}$ can be represented in a linear algebraic form as

$$\mathbf{z}_{I:L} = \mathbf{Y}^T\mathbf{x} = \begin{bmatrix} y(I) & y(I-1) & y(I-2) & \cdots & y(1) \\ y(I+1) & y(I) & y(I-1) & \cdots & y(2) \\ y(I+2) & y(I+1) & y(I) & \cdots & y(3) \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ y(L) & y(L-1) & y(L-2) & \cdots & y(J) \end{bmatrix} \begin{bmatrix} x(1) \\ x(2) \\ x(3) \\ \vdots \\ x(I) \end{bmatrix} \tag{4}$$

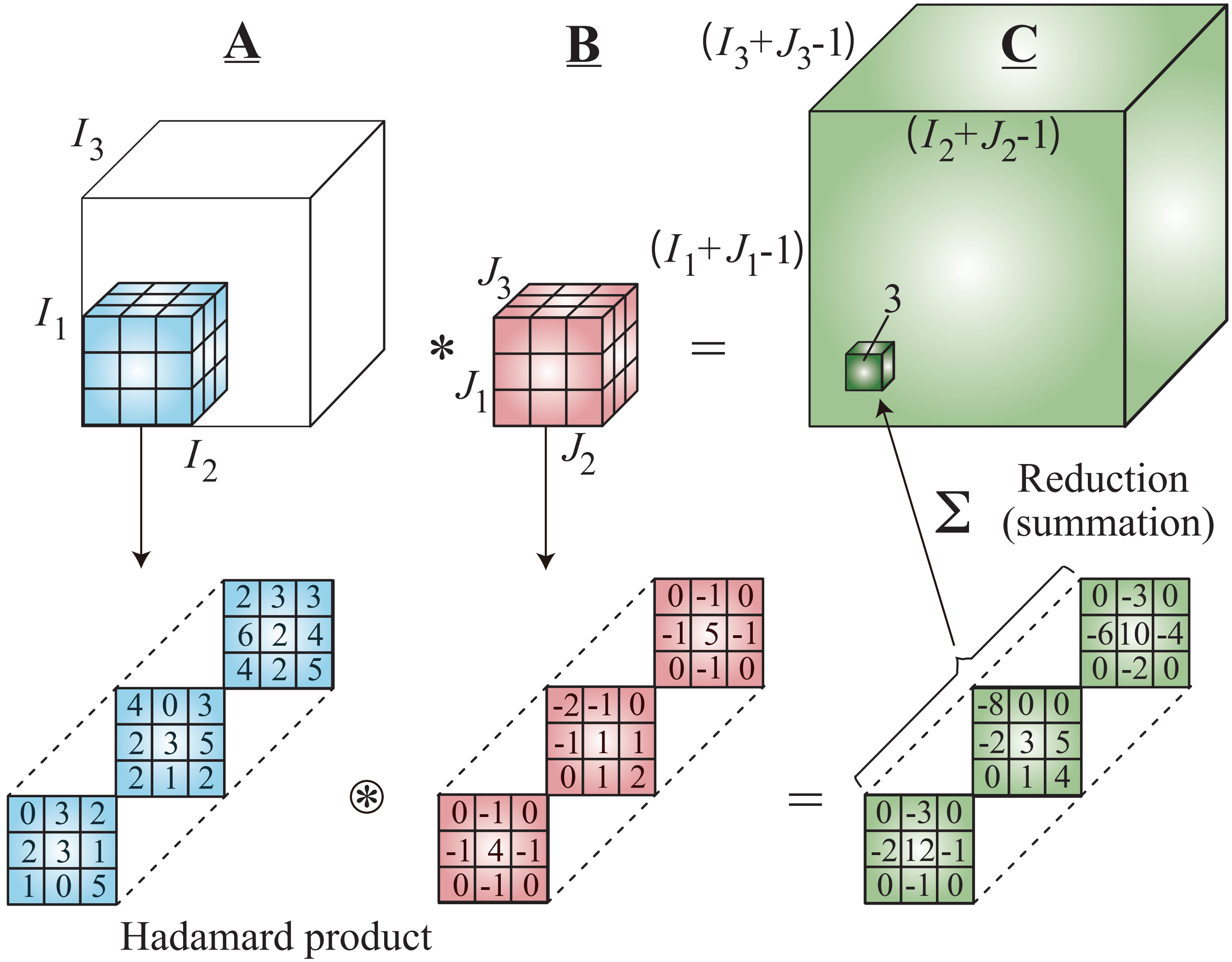
- A linear matrix operator, $\mathbf{Y}$, is called the Toeplitz matrix of the generating vector $\mathbf{y}$
- The convolution of three or more vectors allows us to construct a higher-order tensor

$$\mathbf{z} = \mathbf{x}_1 * \mathbf{x}_2 * \mathbf{y} \tag{5}$$

- First, a Toeplitz matrix $\mathbf{Y}$ is obtained from $\mathbf{x}_1 * \mathbf{x}_2$ as shown in Eq. (4)
- Each row of $\mathbf{Y}(k,:)$, when convolved with a generating vector $\mathbf{y}$, produces its own Toeplitz matrix $\mathbf{Y}_k$, $k = 1, \ldots, J$
- Finally, stacking all $\mathbf{Y}_k$ along e.g. the third mode, gives the tensor $\underline{\mathbf{Y}} = [\mathbf{Y_1}, \ldots, \mathbf{Y_J}]$

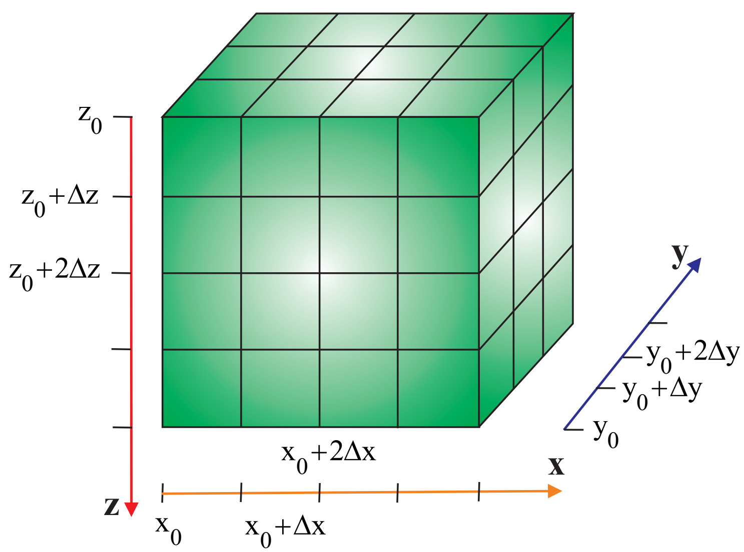
Example 3: Connection with 'flat' DSP, 3D convolution

Many standard operators are readily generalisable to tensors, e.g. the convolution



Curse of dimensionality

- The term **curse of dimensionality** was coined by Bellman (1961) to indicate that the number of samples needed to estimate an arbitrary function with a given level of accuracy grows exponentially with the number of variables, that is, with the dimensionality of the function
- In other words, curse of dimensionality refers to an exponentially increasing number of parameters required to describe an extremely large number of degrees of freedom
- In the context of tensors, the number of elements, I^N, of an Nth-order tensor of size $I \times I \times \cdots \times I$ grows exponentially with the tensor order, N



Example 4: Scientific computing

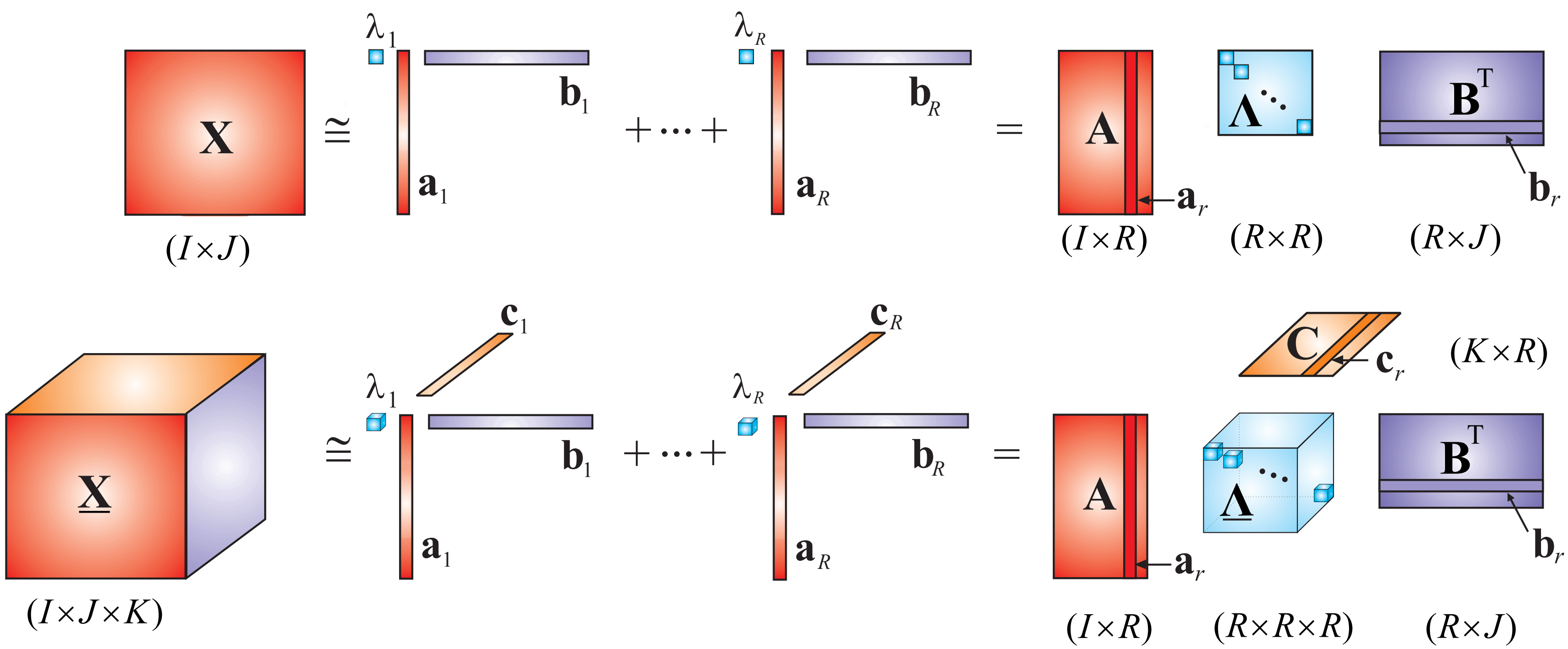
For computational purposes we often need to sample a multidimensional function on a grid (e.g. brain scans)

- For a tri-variate function (N=3, left) sampled at I=1000 points, this will give $I^N = 1000^3 = 10^9$ samples
- For N=4 and I=10,000 this gives $I^4 = 10^{16}$ samples

Remedy: Canonical Polyadic Decomposition (CPD)

Top: Singular Value Decomposition (SVD) for matrices

Bottom: Canonical Polyadic Decomposition (CPD) for tensors ↬ tensor rank = R



- Top: A 'flat-view' matrix $\mathbf{X}$ can be decomposed into a sum of rank-1 matrices $\mathbf{X}_i$
- An 3rd-order tensor $\underline{\mathbf{X}}$ captures 3 dimensions (modes) and can be factorised in the same way ↬ as sum of rank-1 tensors $\underline{\mathbf{X}}_i = \mathbf{a}_i \circ \mathbf{b}_i \circ \mathbf{c}_i, i = 1, 2, \ldots, R$
- This is procedure is referred to as the **Canonical Polyadic Decomposition**
  Canonical ☞ the minimal (rank-1) structure (minimum number of factors)
  Polyadic ☞ the structure is formed by N elements (outer product of N vectors)

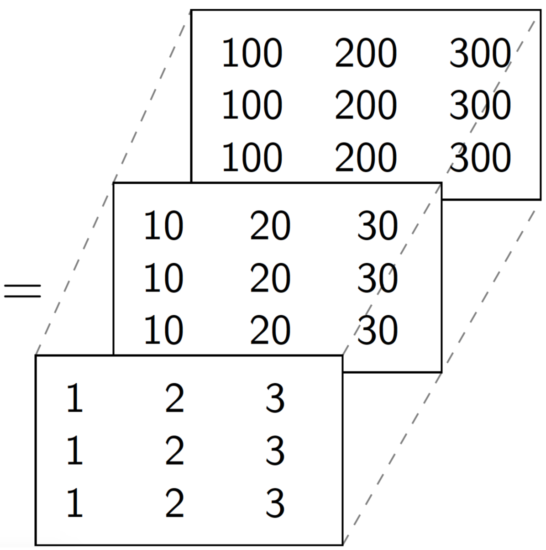
Example 5: The outer product in three dimensions

Consider the vectors $\mathbf{a} = \begin{bmatrix} 1 & 1 & 1 \end{bmatrix}^T, \mathbf{b} = \begin{bmatrix} 1 & 2 & 3 \end{bmatrix}^T, \mathbf{c} = \begin{bmatrix} 1 & 10 & 100 \end{bmatrix}^T$.

$$\mathbf{a} \circ \mathbf{b} \circ \mathbf{c} = \textbf{?} \tag{6}$$

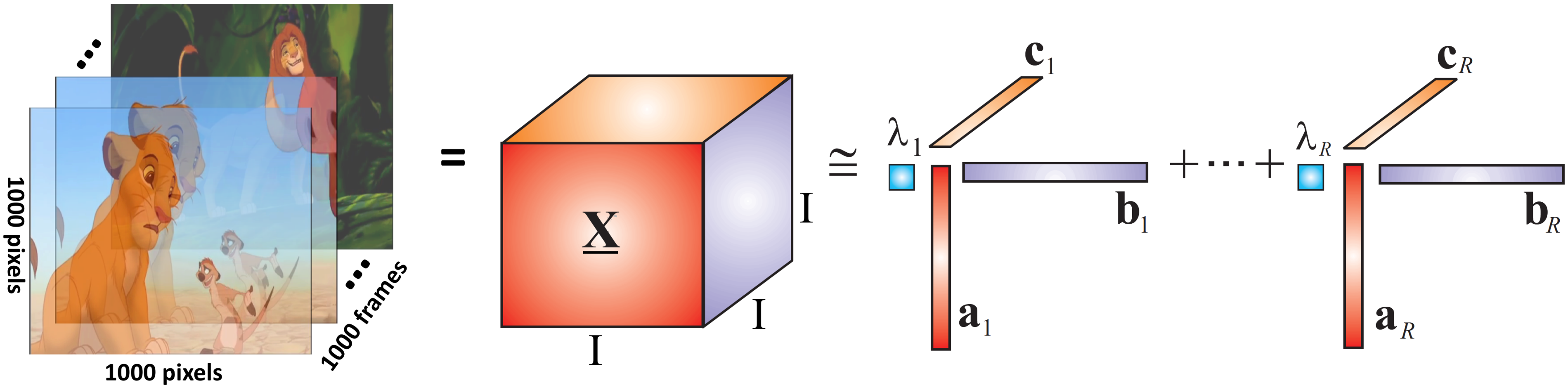
$$\mathbf{a} \circ \mathbf{b} \circ \mathbf{c} = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} \circ \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix} \circ \begin{bmatrix} 1 \\ 10 \\ 100 \end{bmatrix}$$

$$= \begin{bmatrix} 1 & 2 & 3 \\ 1 & 2 & 3 \\ 1 & 2 & 3 \end{bmatrix} \circ \begin{bmatrix} 1 \\ 10 \\ 100 \end{bmatrix} =$$



Example 6: CPD applied to our video-clip example

Inherent compression within the CPD $\looparrowright$ storage and computational advantages



After tensorizing the video clip, tensor order $N = 3$, the dimension in every mode $I = 1000$, and the tensor rank is R. Typically $R \lll I$.

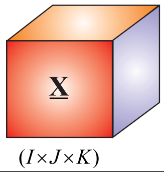
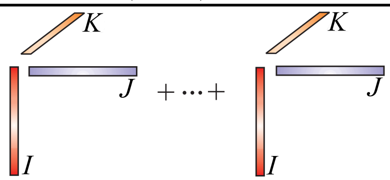
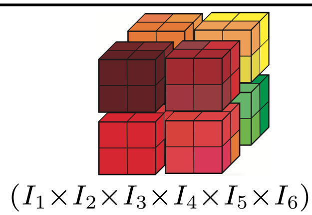
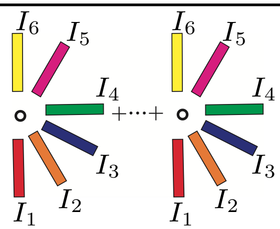
with length($\mathbf{a}_i$)=1000, length($\mathbf{b}_i$)=1000, length($\mathbf{c}_i$)=1000, $\quad i = 1, 2, \ldots, R$

- **Raw data format** $\looparrowright$ $I^N = 1000 \times 1000 \times 1000 = 10^9$ pixels = 1 Giga-pixel
- **In the CPD format**, this becomes $N \times I \times R = 3 \times 1000 \times 10 = 30,000$ pixels (for R=10), that is, compression of almost 5 orders of magnitude
- In scientific computing, if we sample a cube at $I = 10,000$ points, then $I^N = 10^{12}$ raw samples become $N \times I \times R = 3 \times 10^5$ samples in CPD

  For N=4, I=10^4, R=10, the $I^N = 10^{16}$ raw samples $\rightsquigarrow$ 4×10^5 samples in CPD

Super-compression inherent to tensor decomp. (TDs)

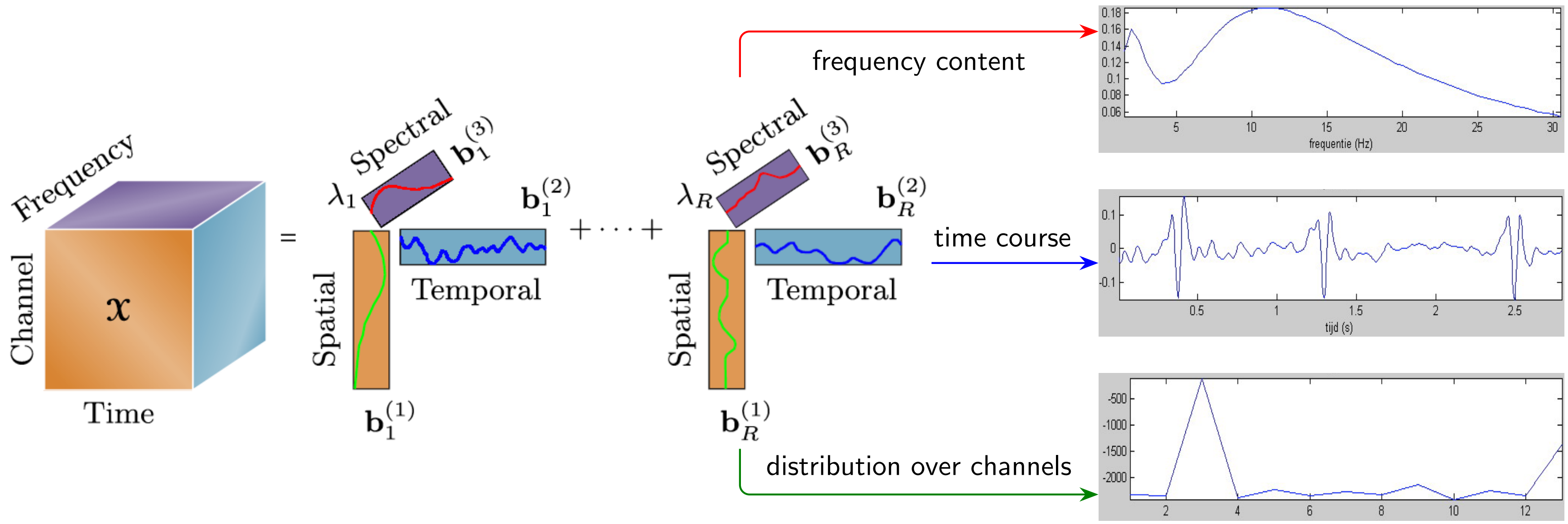
Exponential complexity for the raw data format $\rightsquigarrow$ linear complexity for TDs

Data format	length(mode$_n$)=10	length(mode$_n$)=10^m	General case
	10^3	10^{3m}	IJK
	$R \cdot 3 \cdot 10$	$R \cdot 3 \cdot 10^m$	$R(I+J+K)$
	10^6	10^{6m}	$\prod_{n=1}^{6} I_n$
	$R \cdot 6 \cdot 10$	$R \cdot 6 \cdot 10^m$	$R\sum_{n=1}^{6} I_n$

Number of elements in a data format

- R is the rank of a tensor $\underline{\mathbf{X}}$ $\looparrowright$ CPD is a sum of R rank-1 terms. In practice $R \ll I_n$
- For an N^{th}-order tensor, all I^N elements are efficiently represented through the CPD as a linear (instead of exponential) function of the number of elements in each mode

Intuition and physical meaning behind the CPD



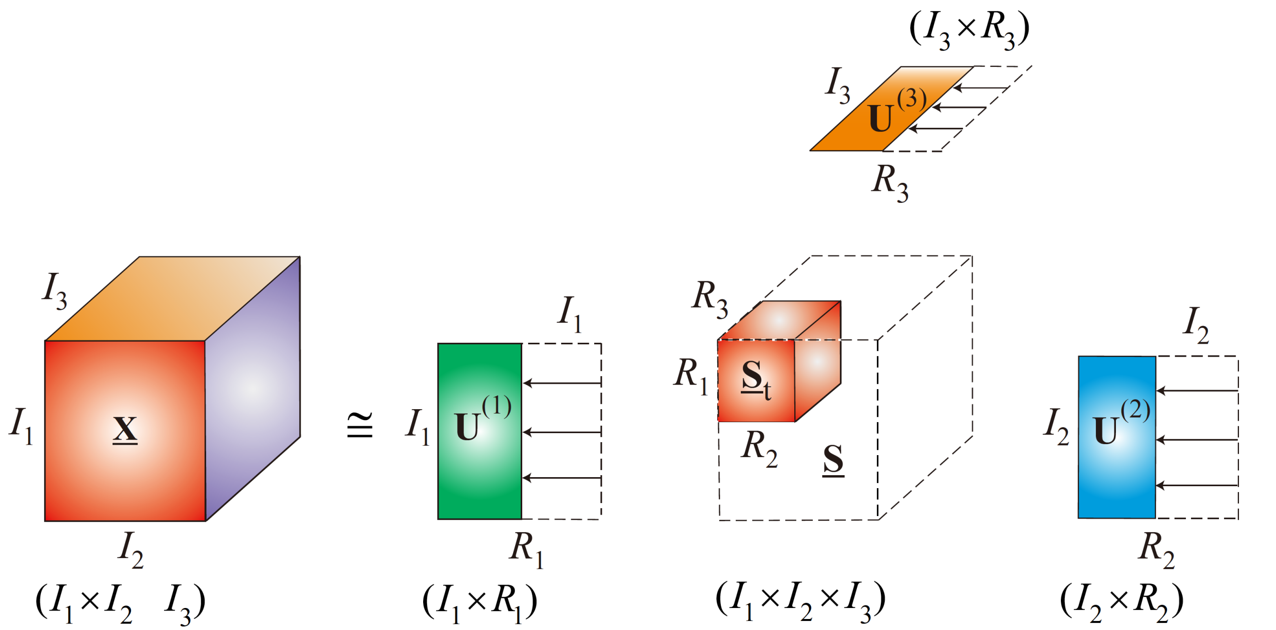
- Components $\mathbf{b}_i^{(1)}, \mathbf{b}_i^{(2)}, \mathbf{b}_i^{(3)}$ (factor 1) are associated with one another (linked)
- However, none of them is associated with any other set of such components (factors) for $i \neq j$, e.g. with $\mathbf{b}_R^{(1)}, \mathbf{b}_R^{(2)}, \mathbf{b}_R^{(3)}$
- Every 'basis' vector has an associated physical meaning, in its respective dimension
- Vectors $\mathbf{b}_1^{(1)}, \mathbf{b}_2^{(1)}, \ldots, \mathbf{b}_R^{(1)}$ can be combined into a factor matrix $\mathbf{B}^{(1)}$ etc., to give

$$\underline{\mathbf{X}} = \sum_{r=1}^{R} \lambda_r \cdot \mathbf{b}_r^{(1)} \circ \mathbf{b}_r^{(2)} \circ \mathbf{b}_r^{(3)} = [\![\underline{\mathbf{D}}; \mathbf{B}^{(1)}, \mathbf{B}^{(2)}, \mathbf{B}^{(3)}]\!] \tag{7}$$

Tucker Decomposition (TKD)

TKD with imposed orthogonality constrains $\looparrowright$ Higher-Order SVD (**HOSVD**)

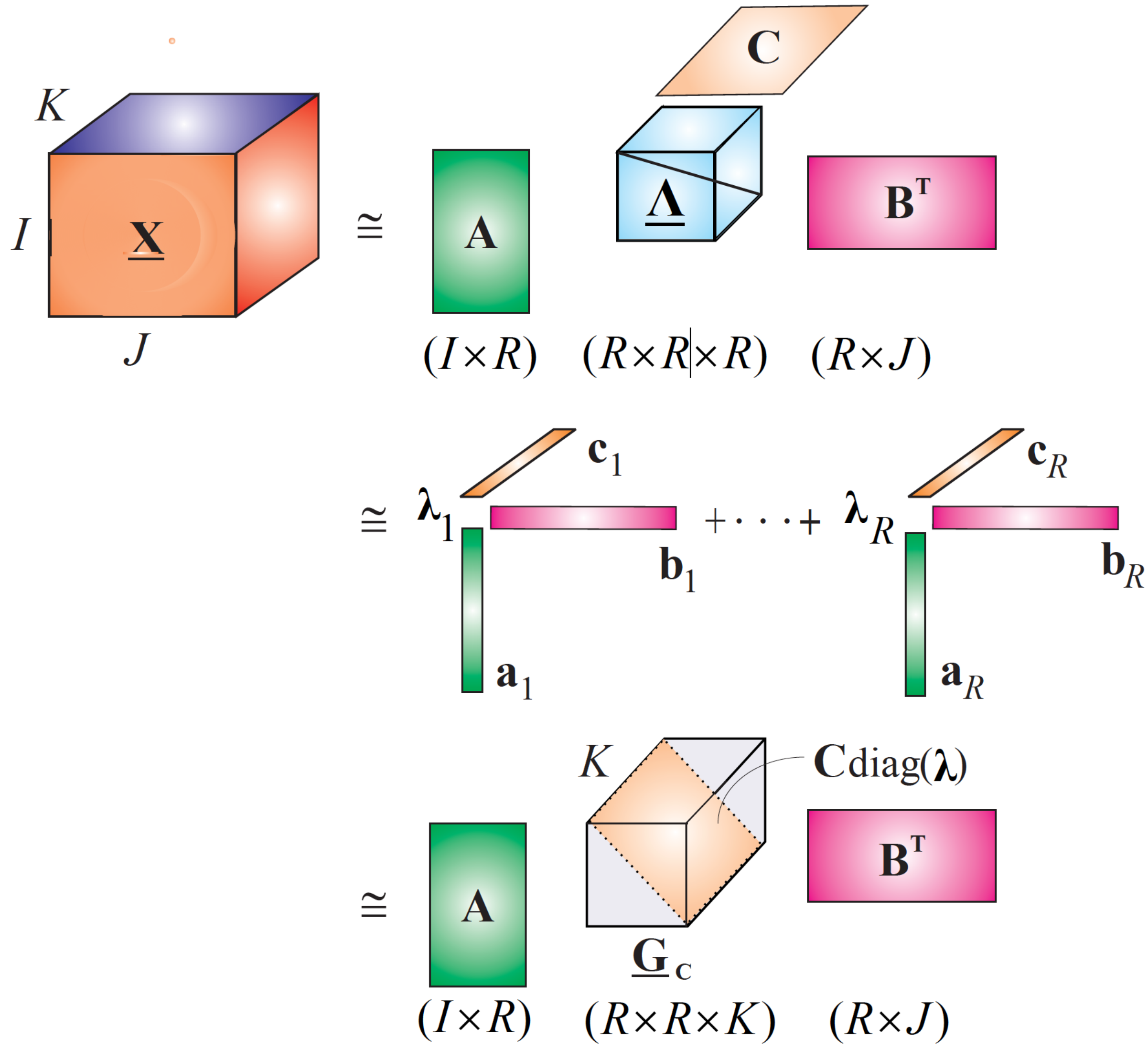
The TKD is not unique, but the **subspaces** defined by $\mathrm{U}^{(1)}, \mathrm{U}^{(2)}, \mathrm{U}^{(3)}$ **are unique**



- Eeach vector of $\mathbf{U}^{(1)}$ is associated with every vector of $\mathbf{U}^{(2)}$ and $\mathbf{U}^{(3)}$ through the **core tensor** $\underline{\mathbf{S}}$ $\looparrowright$ $\underline{\mathbf{X}} \approx \sum_{r_1=1}^{R_1} \sum_{r_2=1}^{R_2} \sum_{r_3=1}^{R_3} \underline{\mathbf{S}}_{r_1 r_2 r_3} \cdot \mathbf{u}_{r_1}^{(3)} \circ \mathbf{u}_{r_2}^{(2)} \circ \mathbf{u}_{r_3}^{(3)}$
- By imposing orthogonality constrains on each factor matrix, we arrive at the natural generalisation of the matrix SVD, the higher-order SVD (HOSVD)
- Low-rank approximation (truncation) is then implemented in analogy with SVD, but separately for each mode, as shown above, where R_1, R_2, R_3 are the truncated ranks

Relation between the CPD and TKD

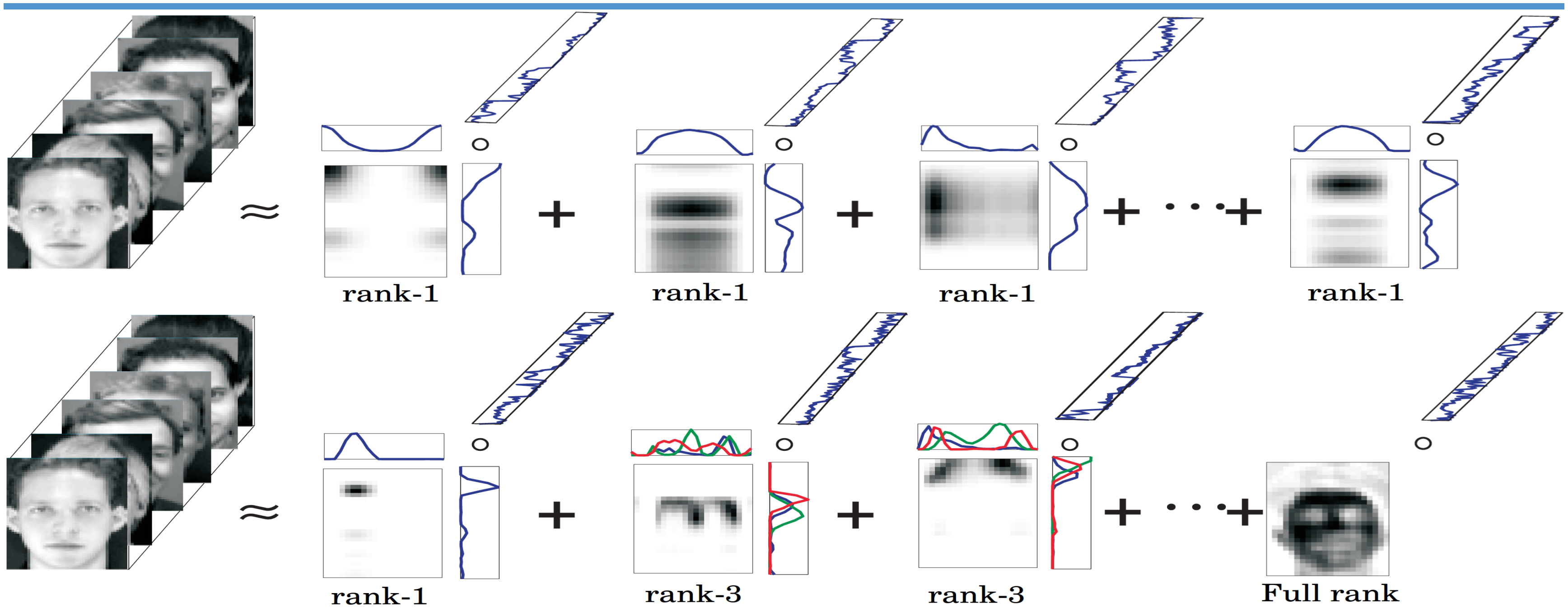
CPD = TKD with a diagonal core



Block Term Decomposition (BTD)

Combination of the CPD and TKD concepts ↬ modeling of complex components

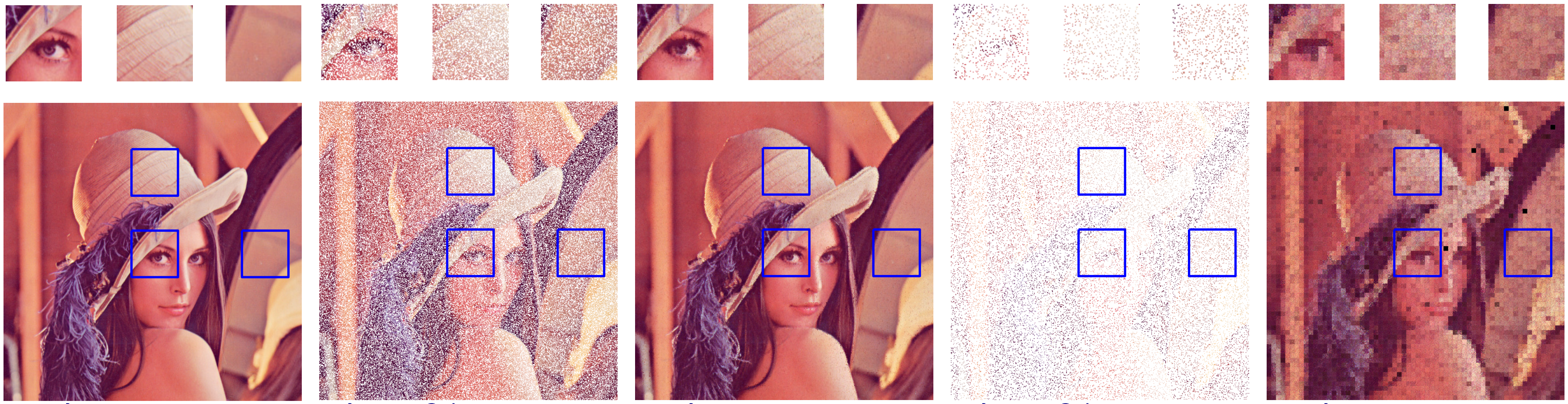
Top: CPD ⇝ sum of rank-1 tensors **Bottom:** BTD ⇝ generalization of CPD



- Complexity of basis images varies according to their ranks. **Rank-1** ↬ local structures. **Full-rank** ↬ more complex structures related to global information
- Combination of basis images with different ranks ↬ structures with a range of complexity levels that represent local and global features at the same time
- The BTD is as a sum of tensors with different ranks ↬ flexible estimation of data
- Each basic sub-tensor in the sum captures a similar structure (regarding dimensions, sparsity profile and constraints) among all examples in a dataset
- With the same number of features, the BTD approximates data better than the CPD

Example 7: Tensor completion (missing data recovery)

A type of BTD (Kronecker BTD) recovers an image with even 90 % missing data



a) original **b) 40% missing** **c) recovered** **d) 90% missing** **e) recovered**

- Missing data may arise due to faulty or unrealiable sensors (Veracity, see Slide 5)
- Missing data recovery is based on the available information (inpainting)
- The RGB image is a natural tensor (see Slide 11), in this case of size $512 \times 512 \times 3$
- For data with structure, like the above image, TDs can peform missing data recovery whereby the missing pixels are recovered through a Kronecker product of available pixels and an "indicator tensor" (binary mask determined by available/mixing pixels)
- Observe good results with even 90% of missing pixels
- The problem of data reconstruction from incomplete information is closely related to the Compressed Sensing paradigm (see Slide 42)

Beyond standard regression $\looparrowright$ latent component analysis
The Partial Least Squares (PLS) method

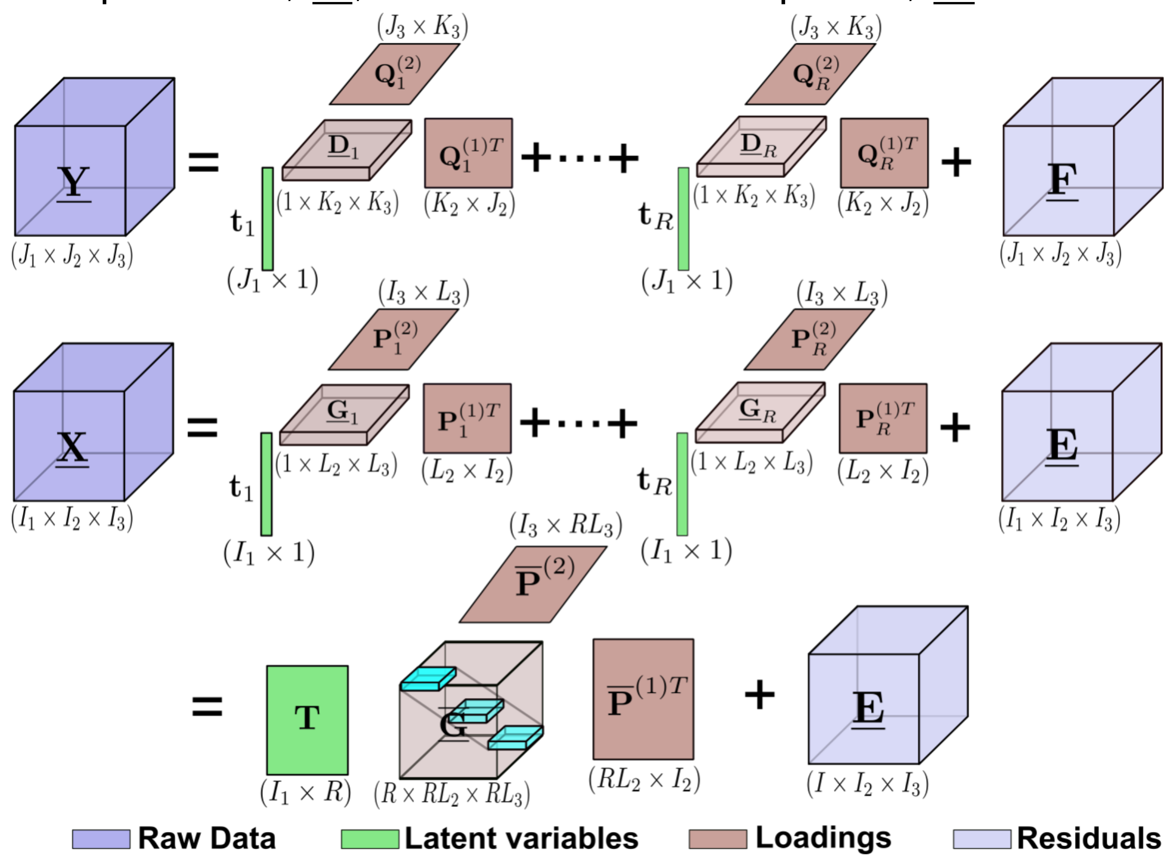
- Regression refers to the modelling of one or more dependent variables (outputs, responses), $\mathbf{Y}$, by a set of independent variables (regressors, predictors), $\mathbf{X}$
- Concept behind PLS ☞ underlying structure is governed by latent variables shared between the $\mathbf{X}$ and $\mathbf{Y}$
- Thus, PLS compromises between fitting $\mathbf{X}$ and predicting $\mathbf{Y}$
- Compare with ARMA(p,q) modelling where the input is white noise (has no structure)

$$\underset{(I\times N)}{\mathbf{X}} = \underset{(I\times R)}{\mathbf{T}}\;\underset{(R\times N)}{\mathbf{P}^{\mathrm{T}}} + \underset{(I\times N)}{\mathbf{E}} = \sum_{r=1}^{R} \boldsymbol{t}_r \boldsymbol{p}_r^{\mathrm{T}} + \underset{(I\times N)}{\mathbf{E}}$$

$$\underset{(I\times M)}{\mathbf{Y}} = \underset{(I\times R)}{\mathbf{U}}\;\underset{(R\times M)}{\mathbf{Q}^{\mathrm{T}}} + \underset{(I\times M)}{\mathbf{F}} = \sum_{r=1}^{R} \boldsymbol{u}_r \boldsymbol{q}_r^{\mathrm{T}} + \underset{(I\times M)}{\mathbf{F}}$$

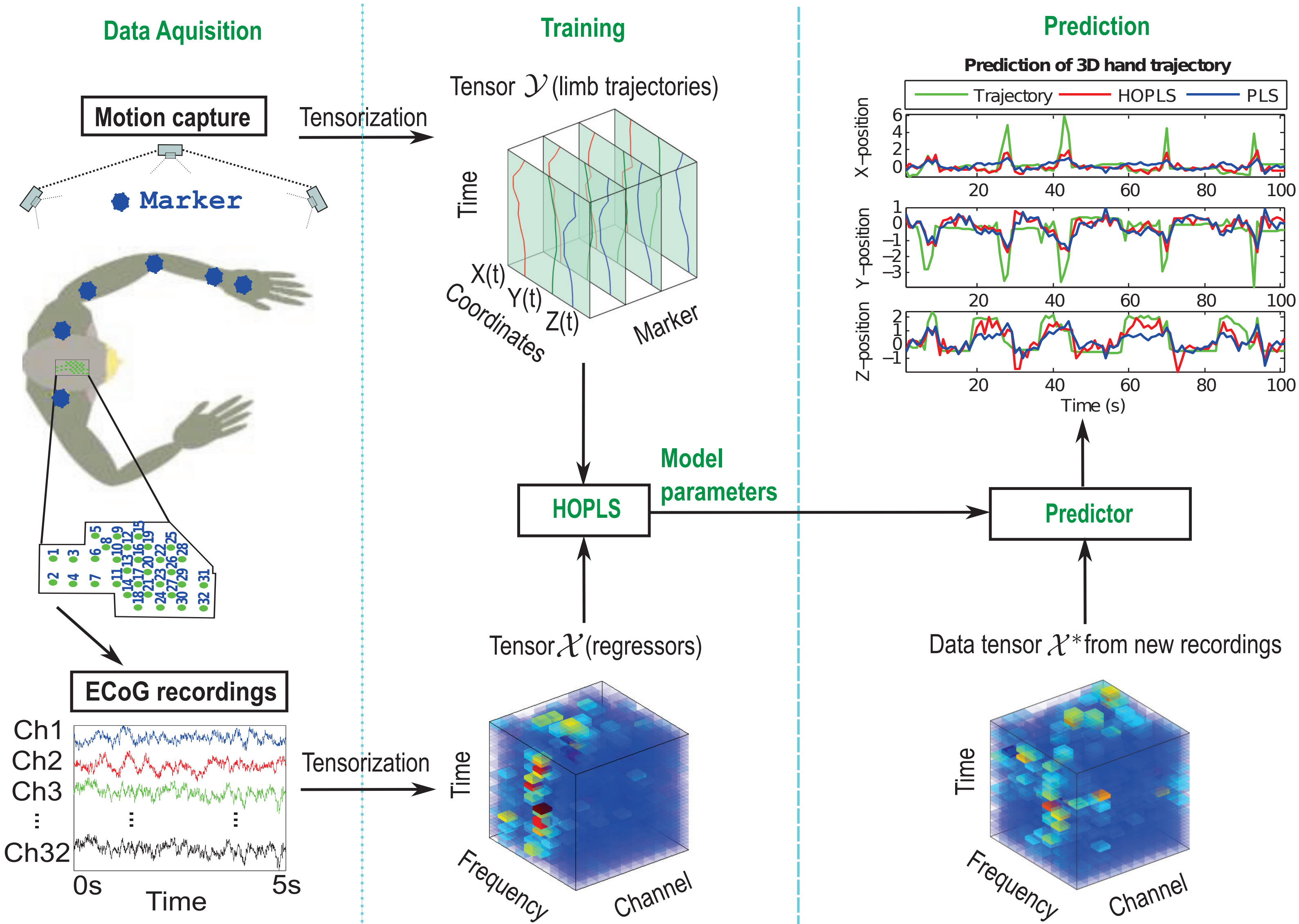
Tensor-valued PLS
The Higher-Order Partial Least Squares (HOPLS)

- **Goal:** to predict a tensor $\underline{\mathbf{Y}}$ from a tensor $\underline{\mathbf{X}}$
- **Approach:** to extract the common latent variables between $\underline{\mathbf{Y}}$ and $\underline{\mathbf{X}}$
- **Advantages:** ability to model interactions between complex latent components of both the tensor of predictors, $\underline{\mathbf{X}}$, and the tensor of responses, $\underline{\mathbf{Y}}$



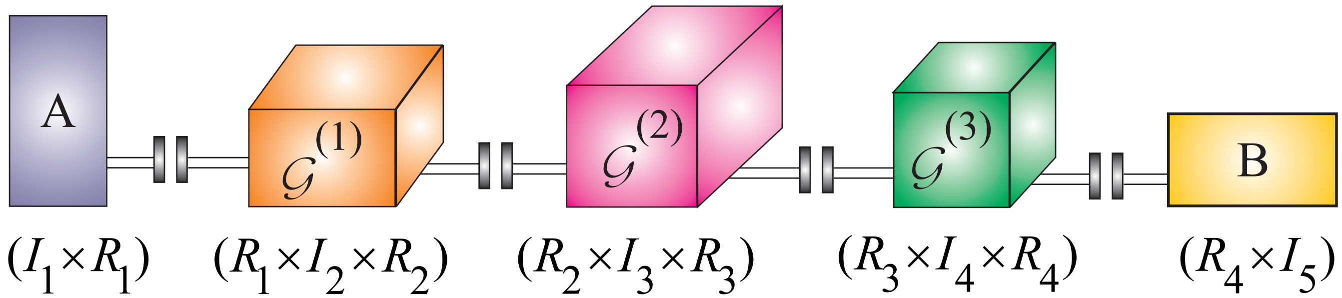
Example 8: Prediction of arm movement from brain activity

Predictors: Brain activity (EEG). **Responses:** 3-D arm movement trajectory (X,Y,Z)

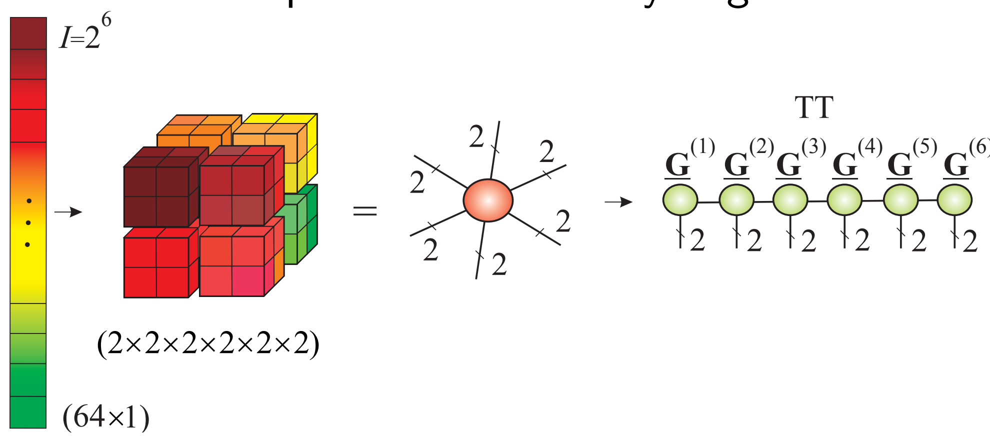


Advanced concepts: Tensor train (TT) decomposition

Curse of dimensionality can be eliminated through tensor network representations

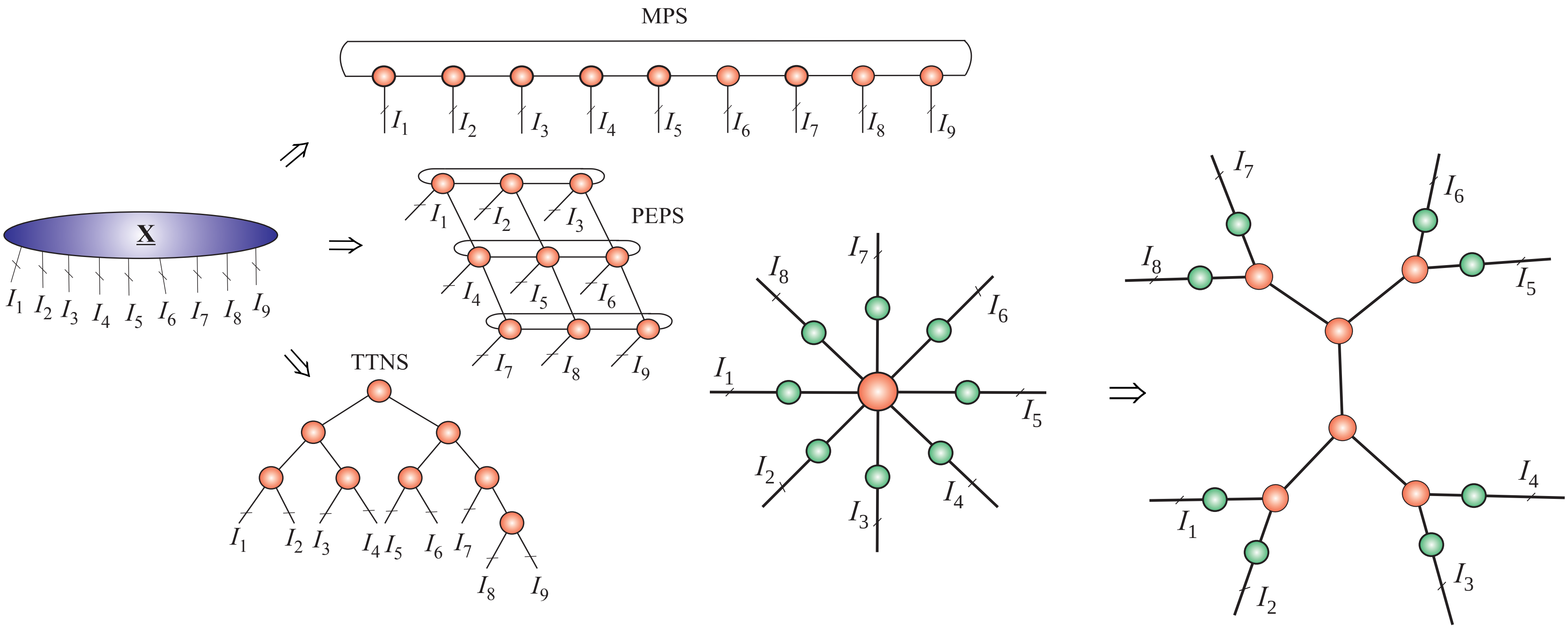


- More degrees of freedom $\rightsquigarrow$ more latent dependencies need to be preserved
- This inevitably leads to *curse of dimensionality (CoD)* (see Slide 23) $\looparrowright$ the number of elements grows exponentially with the the tensor order (number of dimensions)
- TT decomposition represents an Nth-order tensor via two factor matrices, $\mathbf{A}$ and $\mathbf{B}$, and $(N-2)$ small core tensors, $\underline{\mathbf{G}}^i$. These are connected through tensor contractions
- This allows for a distributed representation of very large data on multiple computers



Other types of tensor networks (TNs)

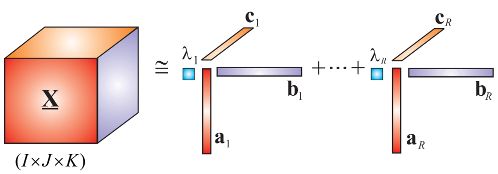
The number of free edges determines the order of a core tensor (usually 3 or 4)



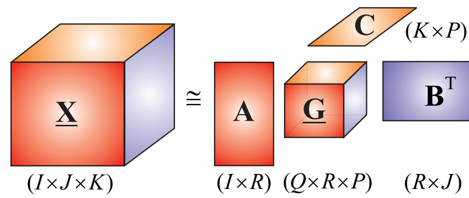
- Tensor network architectures can be with or without loops $\looparrowright$ the Matrix Product State (MPS), Tree Tensor Network State (TTNS), Projected Entangled-Pair States (PEPS), Hierarchical Tucker (HT)
- TNs decompose a very high-order tensor into sparsely (weakly) connected low-order and small-size core tensors (red circles) $\looparrowright$ computational and storage benefits

Comparison of multidimensional decompositions

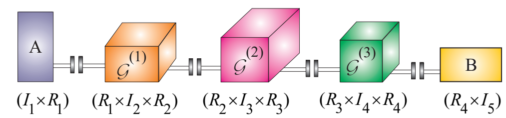
CPD



Tucker



Tensor Train



	CPD	Tucker	Tensor Train
storage complexity	$\mathcal{O}(NIR)$	$\mathcal{O}(NIR + R^N)$	Depends on a choosen type
inherent structure	Represented through rank-1 terms	Represented through core tensors and factor matrices	Represented through tensor contractions
uniqueness conditions	Very soft and depend on the CPD structure	Constrains should be imposed on factor matrices	N/A

Applications across data science

- Civil engineering $\looparrowright$ condition monitoring in structures
- Social networks $\looparrowright$ analysis of information content and information spread
- Multiscale volume visualization $\looparrowright$ integration of tensor decompositions into interactive large-scale volume rendering
- Transportation systems $\looparrowright$ traffic planning and management in intelligent transportation
- Environmental monitoring $\looparrowright$ distributed analysis of ecological parameter spreading at different locations and times
- Internet of things $\looparrowright$ analysis of massive amounts of data captured by embedded devices in large-scale autonomous systems
- Video surveillance $\looparrowright$ crowd density estimation and motion recognition for detection of abnormal activities
- Data fusion $\looparrowright$ combining multiple and diverse data sources to make informed decisions $\looparrowright$ '$1 + 1 > 2$'
- User/topic clustering in text $\looparrowright$ a general tensor model may involve the dimensions e.g. User $\times$ Keyword $\times$ Time
- Network security $\looparrowright$ anomaly via a model Source IP $\times$ Target IP $\times$ Port $\times$ Time

Conclusions

- Multiway data representation and the associated multilinear algebra are a natural way to approach the Big Data paradigm
- Representation of data through higher-order tensors is both physically meaningful and yields storage and computational advantages
- A particular emphasis has been on tensor decompositions (Canonical Polyadic, Tucker) and their applications
- The associated low-rank tensor approximations enable super-compression in tensor formats, thus aleviating or completely eliminating the **curse of dimensionality** associated with Big Data
- With tensors, the complexity of storage becomes linear, $\mathcal{O}(NIR)$, instead of the exponential, $\mathcal{O}(I^N)$, complexity in the raw data format, where N is the number of dimensions in data, R the rank of a tensor, and I the size of the dimensions (modes)
- Tensor networks $\looparrowright$ distributed storage and computing of otherwise unmanageable volumes of data
- Applications $\rightsquigarrow$ video analytics, biomedical eng., social networks, ...

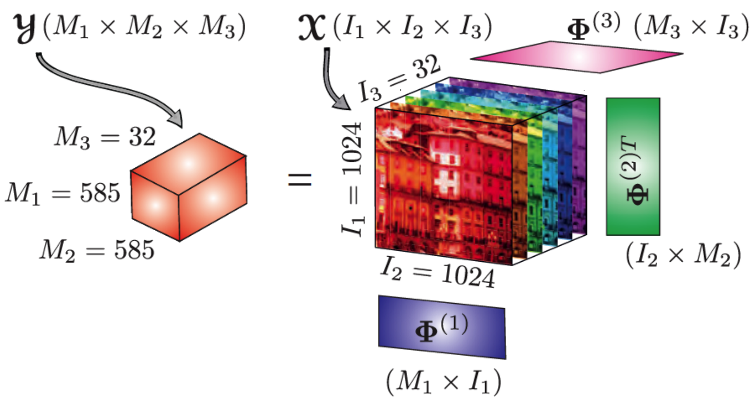
Literature

1. T. G. Kolda and B. W. Bader. "Tensor decompositions and applications". SIAM Review, 51(3):455-500, 2009.
2. A. Cichocki, D. P. Mandic, *et al.*, "Tensor decompositions for signal processing applications: From two-way to multiway component analysis", IEEE Signal Processing Magazine, 32(2):145-163, 2015.
3. Q. Zhao, D. P. Mandic, A. Cichocki *et al.* "Higher order partial least squares (HOPLS): A generalized multilinear regression method". IEEE Transactions on Pattern Analysis and Machine Intelligence, 35(7):1660-1673, 2013.
4. A. Cichocki, D. P. Mandic, *et al.*, "Tensor networks for dimensionality reduction and large scale optimization. Part 1: Low-rank tensor decomposition", Frontiers and Trends in Machine Learning, 9(45):249-429, 2016.
5. A. Cichocki, D. P. Mandic, *et al.*, "Tensor networks for dimensionality reduction and large scale optimization. Part 2: Applications and Future Perspectives", Frontiers and Trends in Machine Learning, 2017.
6. L. De Lathauwer, *et al.*, "A multilinear singular value decomposition", SIAM Journal on Matrix Analysis and Applications 21(4):1253-1278, 2000.
7. J. B. Kruskal, "Three-way arrays: rank and uniqueness of trilinear decompositions, with application to arithmetic complexity and statistics", Linear Algebra and its Applications 18(2):1253-1278, 1977.

Example 9: Higher-Order Compressed Sensing (HO-CS)

An extension to tensors overcomes the loss of spatial and contextual relationships in data, owing to high degrees of structure inherent in tensors

Kronecker-CS of a 32-channel hyperspectral image $\mathcal{X}$



CS $\rightsquigarrow$ signal reconstruction when the set of measurements is much smaller than the original data

Top: Measurement scenario

Top right: Original huge hyperspectral image

Bottom: The hyperspectral image of affordable size, reconstructed using HO-CS

Original hyperspectral image - RGB display

(1024 x 1024 x 32) (256 x 256 x 32)

Reconstruction (SP=33%, PSNR = 35.51dB) - RGB display

(1024 x 1024 x 32) (256 x 256 x 32)

Notes

○

Notes

- ○