University of Hertfordshire UH

Tensor Modelling and Computing: A Survey of methods, applications and performance evaluations

Manal Helal

School of Engineering and Computer Science
Department of Computer Science

Presented to: Computer Science Colloquium
24/02/2021

TEF Gold

Outline

- Abstract
- Definition
- Tensor Decomposition Techniques
- Applications
- State of the Art Performance Evaluations
- Conclusion & Future Directions

ABSTRACT

- Tensors are not just higher dimensional arrays, they are higher order extensions to vectors and matrices. They capture multi-linear structures more efficiently than matricising higher dimensions datasets. Dimensionality curse performance degradation is addressed using various approaches. This talk will survey tensor decompositions techniques and their applications in data mining and machine learning along with performance evaluations, challenges and future trends.

High dimensional Matrices

Linear Algebra

Vectors are linear structure that captures dimension and magnitude representative to some reference frame that is mostly the origin.

Matrices capture linear transformation in the vector spaces for the columns, and used in various linear actions such as in computer graphics for rotations, reflections, and other linear transformations such as scaling, additions, and sheering among others.

High dimensional matrices sometimes refers to matrices of higher number of columns that will require computational power that might not be feasible for large problems.

Dimensionality reduction algorithms are generally applied to capture the structure of the dataset in lower dimensions, such as PCA, capture the highest variance in the first few uncorrelated orthogonal principal components using eigenvectors characteristics and ranking by highest eigen values. SVD provides lower rank compressions using singular values to identify the best approximate rank.

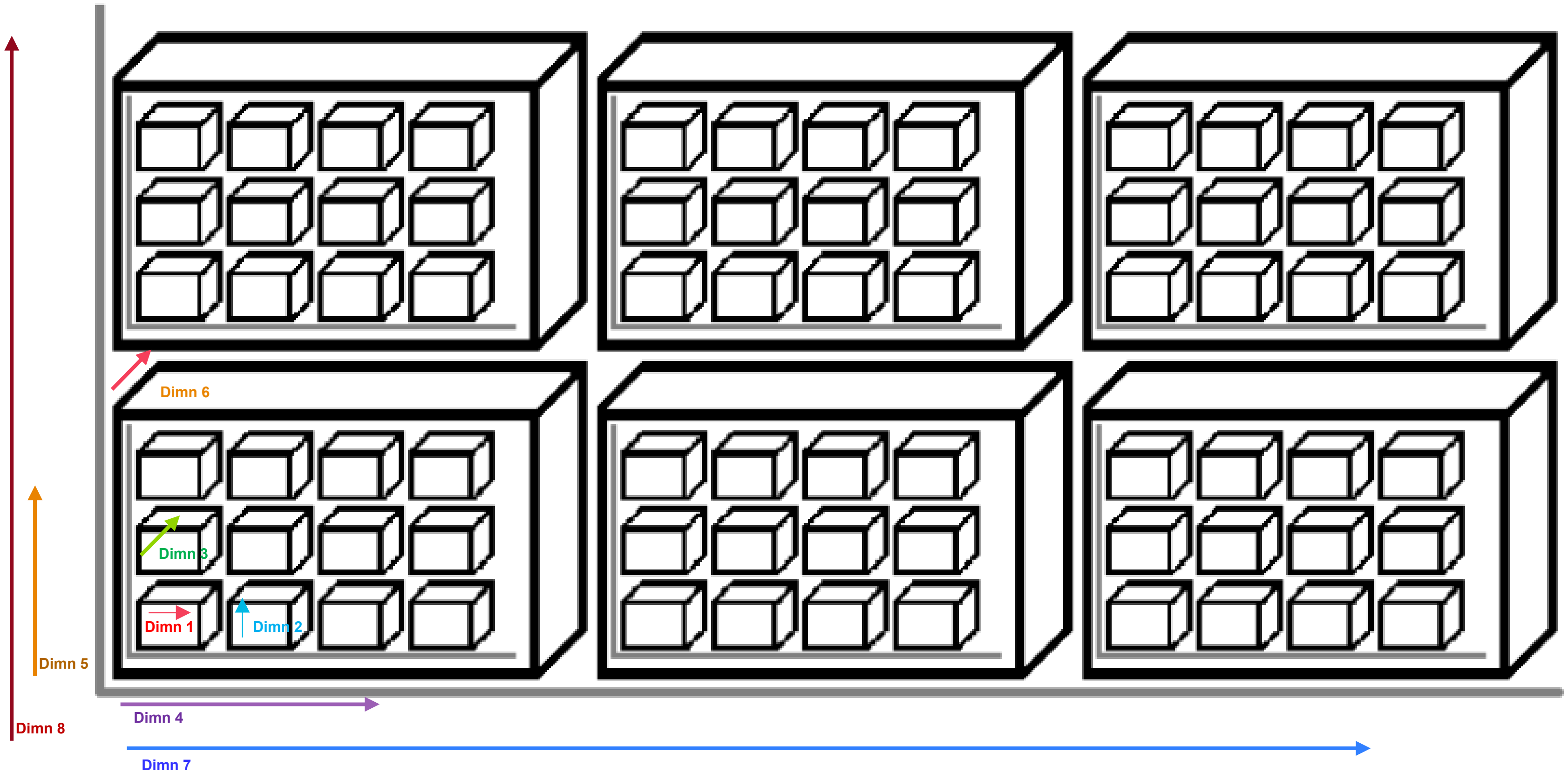
From Matrix to Tensor: A Complex Extrapolation

		Scalar-Level Thinking		
1960's	$\Rightarrow$	$\Downarrow$	$\Leftarrow$	The factorization paradigm: LU, LDL^T, QR, $U\Sigma V^T$, etc.
		Matrix-Level Thinking		
1980's	$\Rightarrow$	$\Downarrow$	$\Leftarrow$	Cache utilization, parallel computing, LAPACK, etc.
		Block Matrix-Level Thinking		
2000's	$\Rightarrow$	$\Downarrow$	$\Leftarrow$	New applications, factorizations, data structures, nonlinear analysis, optimization strategies, etc.
		Tensor-Level Thinking		

Charles Van Loan *et al.*, 'Future Directions in Tensor-Based Computation and Modeling', Arlington, Virginia at the National Science Foundation, Feb. 2009. Available: http://www.cs.cornell.edu/cv/TenWork/Home.htm.

What is a Tensor

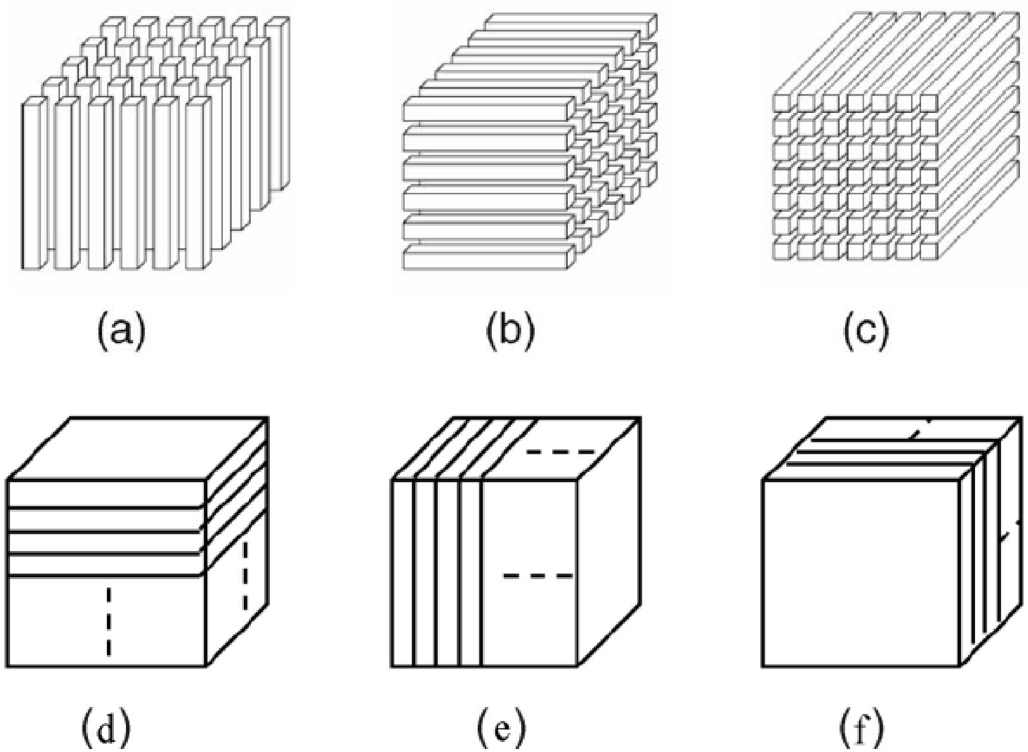
- From relativity theory, tensors were used to interpret movement in space and time, from particles in the atom to the universe astronomical objects in a hierarchy of reference frames.
- Typical usage is in space time analysis, where space is three dimensional in nature (x, y, z) and time t is measured with intervals with constant c, then an object s world-line is expressed as $ds^2 = dx^2 + dy^2 + dz^2 + c^2dt^2$
- A typical tensor object in pattern recognition or machine vision applications is commonly specified in a high-dimensional tensor space.
- Recognition methods operating directly on this suffer from the curse of dimensionality.



Tensor Representation

Tensors maintain the multiway interactions in the higher spaces.

The n indices for the n dimensions, such as the i^{th} index is a point in the domain of the i^{th} coordinate, describing a function mapping the index values as coefficients to variables mapping to an output value in the cell indexed.



(a) Mode-1 (Columns) fibers $x_{:jk}$ (b) Mode-2 (row) fibers: $x_{i:k}$ (c) Mode-3 (tube) Fibers: $x_{ij:}$(d) Horizontal Slices (e) Vertical Slices (f)Frontal Slices

Tensor Operations

Multilinear Algebra

- Tensor N-Mode Products for $X \in \mathcal{R}^{\mathrm{I}_1 \times \mathrm{I}_2 \times \ldots \times \mathrm{I}_N}$ with matrix U $\in \mathcal{R}^{J x \mathrm{I}_N}$ results in tensor $\in \mathcal{R}^{\mathrm{I}_1 \times \ldots \mathrm{I}_{n-1} \times \mathrm{J} \times I_{N+1} \ldots \times I_N}$

$$(\mathrm{X} \times_n U)_{i_1, i_2, \ldots i_{n-1}, j, i_{n+1}, \ldots , i_n} = \sum_{i_n=1}^{I_N} x(i_1, i_2, \ldots i_n) u_{jin}$$

  - The n-mode product of a tensor with a matrix is related to a change of basis in the case when a tensor defines a multilinear operator.

- The Kronecker product for $X \in \mathcal{R}^{\mathrm{I} x J}$ with matrix U $\in \mathcal{R}^{KxL}$ results in tensor $\in \mathcal{R}^{IKxJL}$

$$\chi \otimes U = \begin{bmatrix} x_{1,1}U & \cdots & x_{1,J}U \\ \vdots & \ddots & \vdots \\ x_{I,1}U & \cdots & x_{I,J}U \end{bmatrix} = [x_1 \otimes u_1 \quad x_1 \otimes u_2 \quad x_1 \otimes u_3 \ldots x_J \otimes u_{L-1} \; x_J \otimes u_L]$$

- The Khatri-Rao product for $X \in \mathcal{R}^{\mathrm{I} x K}$ with matrix U $\in \mathcal{R}^{JxK}$ results in tensor $\in \mathcal{R}^{IJxK}$

$$\chi \odot U = [x_1 \otimes u_1 \quad x_2 \otimes u_2 \;\ldots\; x_K \otimes u_k]$$

T. G. Kolda and B. W. Bader, ‘Tensor Decompositions and Applications’, *SIAM Rev.*, vol. 51, no. 3, pp. 455–500, Aug. 2009, doi: 10.1137/07070111X.

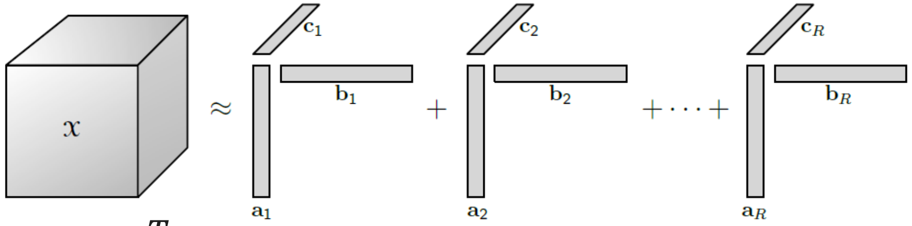
Dimensionality Reduction

Summation Notation - CANDECOMP

- Hitchcock in 1927 proposed the idea of the polyadic form of a tensor, i.e., expressing a tensor as the sum of a finite number of rank-one tensors;
- **Canonical decomposition (CANDECOMP)** factorises a tensor into a sum of $\chi \in \mathcal{R}^{IxJxK}$

$\chi = \sum_{r=1}^{R} a_r \circ b_r \circ c_r \approx \sum_{r=1}^{R} a_{ir} b_{jr} c_{kr}$

for all $a_r \in \mathcal{R}^I$, $b_r \in \mathcal{R}^J$, and $c_r \in \mathcal{R}^K$.



Producing $\chi_{(1)} \approx A(C \odot B)^T$, , $\chi_{(2)} \approx B(C \odot A)^T$, $\chi_{(3)} \approx C(B \odot A)^T$

Concisely expressed as $\chi \approx [\![\lambda; A, B, C]\!] = \sum_{r=1}^{R} \lambda_r \, a_r \circ b_r \circ c_r$

This three-way model is expressed as the frontal slices of χ

N dimensions generalisation: $X \in \mathcal{R}^{\mathrm{I}_1 \times \mathrm{I}_2 \, x \, \ldots x \mathrm{I}_N}$ as $\chi \approx [\![\lambda; A^{(1)}, A^{(2)}, \ldots, A^{(N)}]\!] = \sum_{r=1}^{R} \lambda_r \, a_r^{(1)} \circ a_r^{(2)} \circ \ldots \, a_r^{(N)}$

Example CP applications:

- time-varying EEG spectrum arranged as a three-dimensional array with modes corresponding to time, frequency, and channel.
- vowel-sound data where different individuals (mode 1) spoke different vowels (mode 2) and the formant (i.e., the pitch) was measured (mode 3).

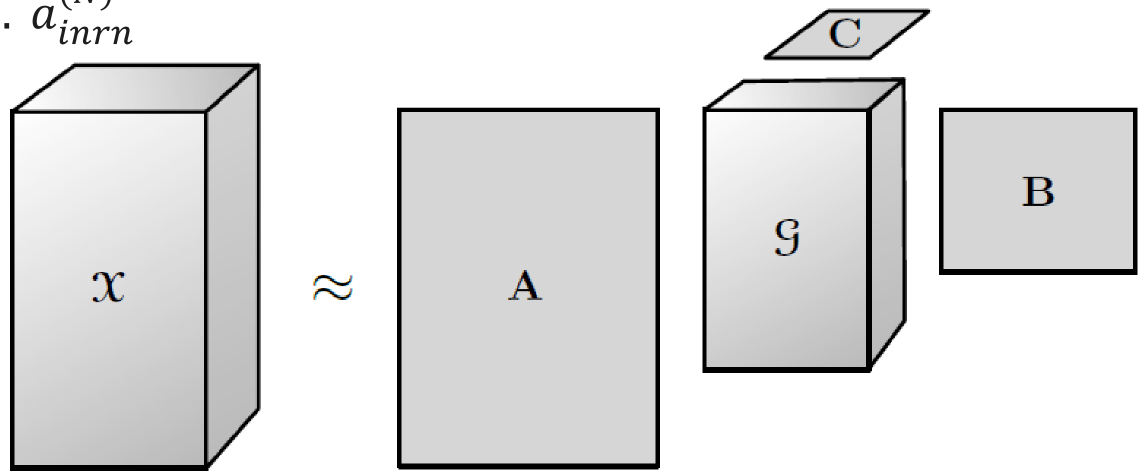
Tucker 3-way and multiway Analysis

- Decomposes a tensor to a core tensor multiplied by a matrix along each mode:

$$\chi \approx G \times_1 A \times_2 B \times_3 C = \sum_{i=1}^{I} \sum_{j=1}^{I} \sum_{k=1}^{K} g_{ijk} a_i \circ b_j \circ c_k = [\![G; A, B, C]\!]$$

N-Dim generalisation: $X \epsilon \mathcal{R}^{I_1 \times I_2 \times \ldots \times I_N}$ as $\chi \approx [\![G; A^{(1)}, A^{(2)}, \ldots., A^{(N)}]\!]$

$$x_{i1i2\ldots iN} = \sum_{r1=1}^{R1} \ldots \sum_{rn=1}^{RN} \ldots g_{r_1 \ldots r_n} \; a_{i1r1}^{(1)} \circ a_{i2r2}^{(2)} \circ \ldots \; a_{inrn}^{(N)}$$



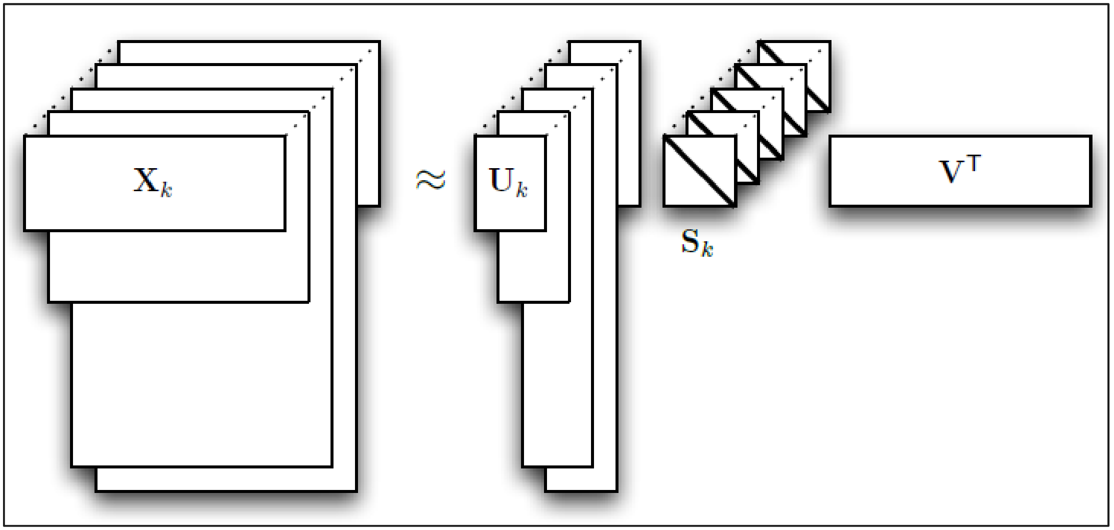
Example Applications:

- TensorFaces takes facial images for different people, each in different angles, lighting, facial expressions, … more modes as required

M. A. O. Vasilescu and D. Terzopoulos, Multilinear analysis of image ensembles: Tensor-Faces, in ECCV 2002: Proceedings of the 7th European Conference on Computer Vision, vol. 2350 of Lecture Notes in Computer Science, Springer, 2002, pp. 447{460.

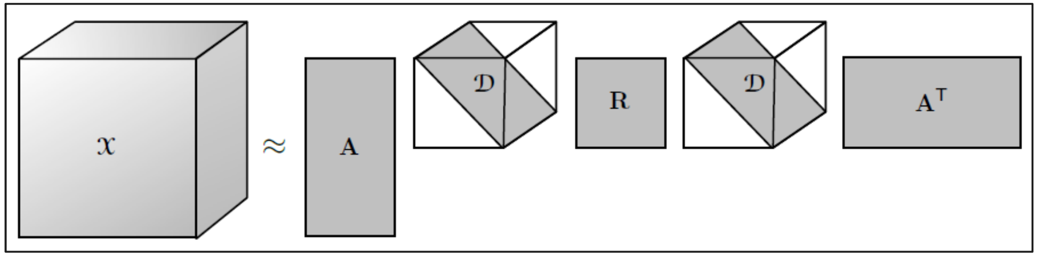
Other Tensor Decomposition Approaches

PARAFAC2



Example Application: PARAFAC2 handles time shifts in resolving chromatographic data with spectral detection. In this application, the first mode corresponds to elution time, the second mode to wavelength, and the third mode to samples.

DEDICOM



$$X \approx ARA^T$$

Example Application: Bader et al. [Temporal analysis of semantic graphs using ASALSAN - ICDM 2007] applied their ASALSAN method for computing DEDICOM on email communication graphs over time. In this case, x_{ijk} corresponded to the (scaled) number of email messages sent from person i to person j in month k.

Dimensionality Curse

- Approximation and separability are of paramount importance. By representing functions of many variables as sums of separable functions, one obtains a method to bypass the curse of dimensionality.
- For example: Tensor networks represent a very high-order tensor by connecting many low-order tensors through contractions and sparse representations. Example datasets are found in solving Hamiltonian eigenvalue problems in quantum chemistry. Vectors of order $n = 2^{100}$ can be successfully approximated with many fewer than n numbers.

Performance Evaluation

Tensor Does not suffer from Dimensionality Curse

Input	Output	VVP	TVP	TTP
$\prod_{n=1}^{N} I_N$	P	$P\prod_{n=1}^{N} I_N$	$P\sum_{n=1}^{N} I_N$	$P\sum_{n=1}^{N} P_N \times I_N$
10×10	4	400	80	40 (Pn = 2)
100×100	4	40,000	800	400 (Pn = 2)
$100 \times 100 \times 100$	8	8,000,000	2400	600 (Pn = 2)
$\prod_{n=1}^{4} 100$	16	1,600,000,000	6400	800 (Pn = 2)

H. Lu, K. N. Plataniotis, and A. N. Venetsanopoulos, ‘A survey of multilinear subspace learning for tensor data’, *Pattern Recognit.*, vol. 44, no. 7, pp. 1540–1551, Jul. 2011, doi: 10.1016/j.patcog.2011.01.004.

Comparison	Linear subspace learning	Multilinear subspace learning
Representation	Reshape into vectors	Natural tensorial representation
Structure	Break natural structure	Preserve natural structure
Parameter	Estimate a large number of parameters	Estimate fewer parameters
SSS problem	More severe SSS problem	Less SSS problem
Massive data	Hardly applicable to massive data	Able to handle massive data

University of Hertfordshire UH

Tensorising Neural Networks

- The dense weight matrices of the fully-connected layers in DNN can be represented by the Tensor Train (TT) format such that the number of parameters is reduced by a huge factor while preserving the expressive power of the layer. TT can compute all the derivatives required by the back-propagation algorithm.

- TT-Format: $\chi \in \mathcal{R}^{J_1 \times J_2 x \ldots x L_D}$ $(j_1;\ldots ; j_d) = G_1[j1]G_2[j2] \ldots G_d[j_d]$

- **Example Application:** Very Deep VGG networks we report the compression factor of the dense weight matrix of a fully-connected layer up to 200000 times leading to the compression factor of the whole network up to 7 times.

A. Novikov, D. Podoprikhin, A. Osokin, and D. Vetrov, ‘Tensorizing Neural Networks’, *ArXiv150906569 Cs*, vol. 28, Dec. 2015, Accessed: Feb. 24, 2021. [Online]. Available: http://arxiv.org/abs/1509.06569.

Vector and Matrix Parallel Processing Examples

Algorithm: $[B] := \textsc{Transpose}(A,B)$

Partition $A \rightarrow \left(A_L \,\big|\, A_R \right)$, $B \rightarrow \left(\begin{array}{c} B_T \\ \hline B_B \end{array} \right)$

where A_L has 0 columns, B_T has 0 rows

while $n(A_L) < n(A)$ **do**

Repartition

$$\left(A_L \,\big|\, A_R \right) \rightarrow \left(A_0 \,\big|\, a_1 \,|\, A_2 \right), \left(\begin{array}{c} B_T \\ \hline B_B \end{array} \right) \rightarrow \left(\begin{array}{c} B_0 \\ \hline b_1^T \\ \hline B_2 \end{array} \right)$$

where a_1 has 1 column, b_1 has 1 row

$b_1^T := a_1^T$ (Set the current row of B to the current column of A)

Continue with

$$\left(A_L \,\big|\, A_R \right) \leftarrow \left(A_0 \,|\, a_1 \,\big|\, A_2 \right), \left(\begin{array}{c} B_T \\ \hline B_B \end{array} \right) \leftarrow \left(\begin{array}{c} B_0 \\ \hline b_1^T \\ \hline B_2 \end{array} \right)$$

endwhile

Algorithm: $[B] := \textsc{Transpose\_alternative}(A,B)$

Partition $A \rightarrow \left(\begin{array}{c} A_T \\ \hline A_B \end{array} \right)$, $B \rightarrow \left(B_L \,\big|\, B_R \right)$

where A_T has 0 rows, B_L has 0 columns

while $m(A_T) < m(A)$ **do**

Repartition

$$\left(\begin{array}{c} A_T \\ \hline A_B \end{array} \right) \rightarrow \left(\begin{array}{c} A_0 \\ \hline a_1^T \\ \hline A_2 \end{array} \right), \left(B_L \,\big|\, B_R \right) \rightarrow \left(B_0 \,\big|\, b_1 \,|\, B_2 \right)$$

where a_1 has 1 row, b_1 has 1 column

```
b₁ = a₁ᵀ;        (Set the current rows of A into the current
                 columns of B.)
```

Continue with

$$\left(\begin{array}{c} A_T \\ \hline A_B \end{array} \right) \leftarrow \left(\begin{array}{c} A_0 \\ \hline a_1^T \\ \hline A_2 \end{array} \right), \left(B_L \,\big|\, B_R \right) \leftarrow \left(B_0 \,|\, b_1 \,\big|\, B_2 \right)$$

endwhile

R. A. van de Geijn and E. S. Quintana-Ort´, *The Science of Programming Matrix Computations*. www.lulu.com, 2008.

Tensor Partitioning Example for Multiple Sequence Alignment

2 Dimension Example

- Figure a shows two sequences partitioning space (Full matrix is (n1 x n2), where n1 is length of first sequence on the rows, and n2 is length of second sequence on the columns) visualising each dot as a partition of a matrix of size pxp (stride size) over three waves. First partition in the first wave, starts from index (0, 0), to index (p,p). The last column and the last row in the partition is sent for communication for following wave starting (p, p) to (2p, p) on one processor, and (p, 2p) on another processor, and so forth.

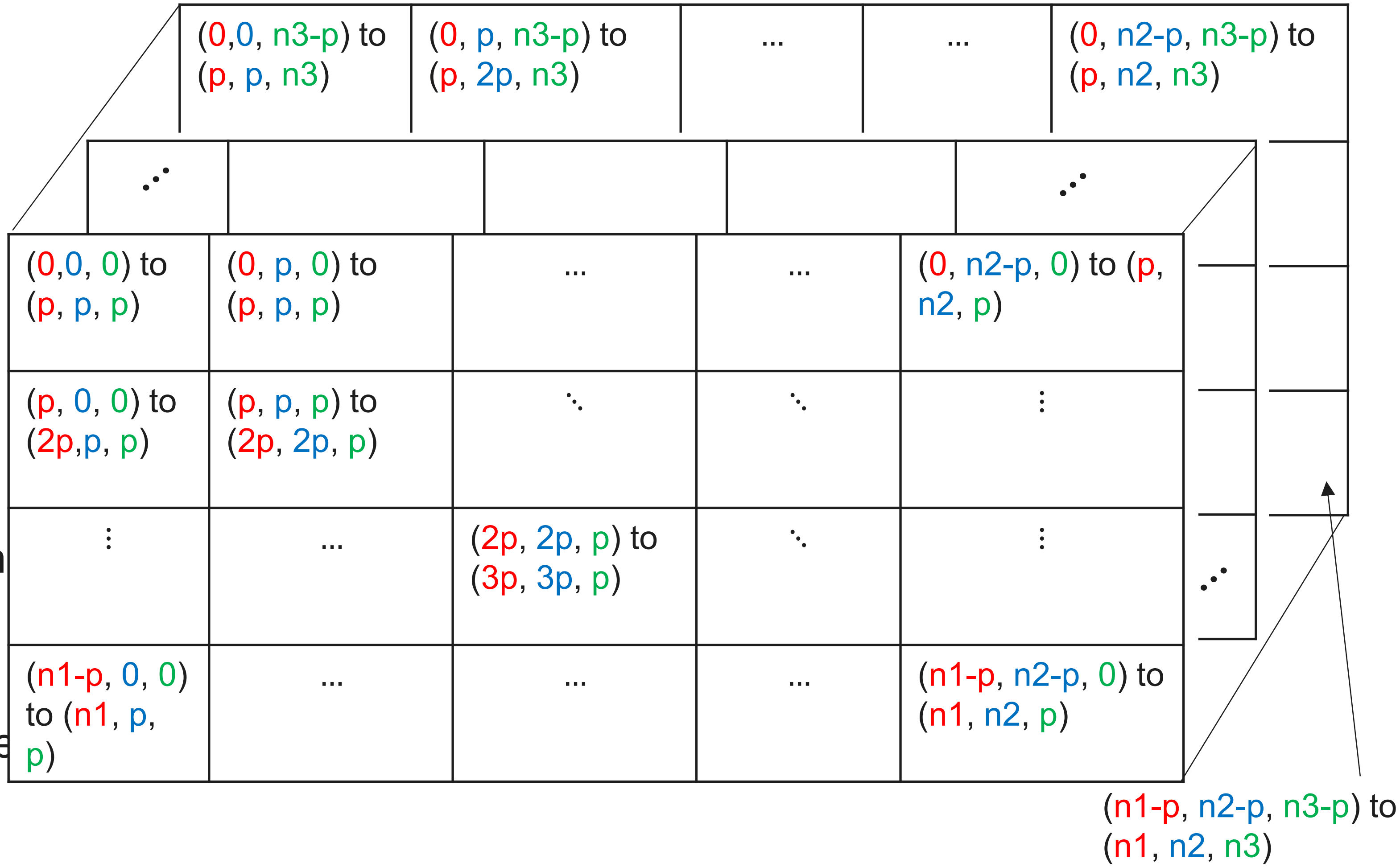
(0,0) to (p,p)	(0,p) to (0,2p)	…	…	(0, n2-p) to (0,n2)
(p,0) to (2p,0)	(p,p) to (2p,2p)	⋱	⋱	⋮
⋮	…	(2p,2p) to (3p,3p)	⋱	⋮
(n1-p, 0) to (n1,0)	…	…	…	(n1-p, n2-p) to (n1,n2)

University of Hertfordshire UH

Tensor Partitioning Example for Multiple Sequence Alignment

3 dimension Example

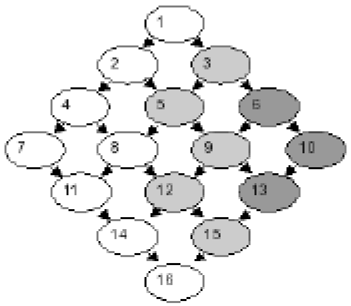
- The Figure shows three sequences partitioning space (Full matrix is (n1 x n2 x n3), where n1 is length of first sequence , … etc) visualising each dot as a partition of a matrix of size pxpxp over n1xn2xn3/p waves. First partition in the first wave, starts from index (0, 0, 0), to index (p,p, p). The last column and the last row in the partition is sent for communication for following wave starting (p, p, p) to (2p, 2p, 2p) on one processor, and (p, 2p) on another processor.



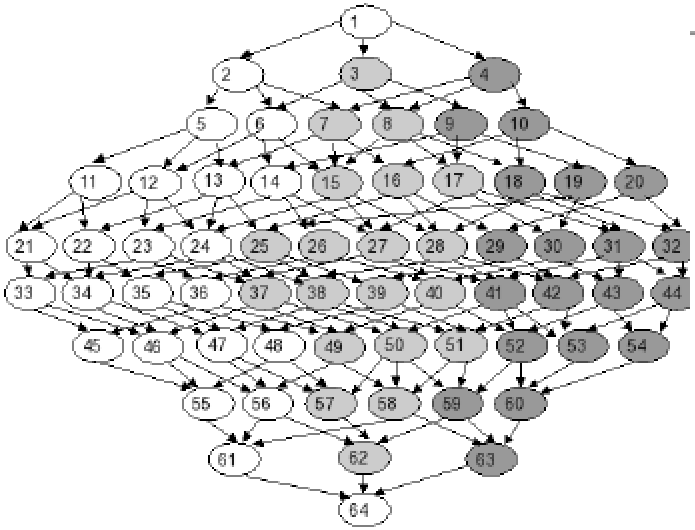
University of Hertfordshire UH

Tensor Partitioning Example for Multiple Sequence Alignment

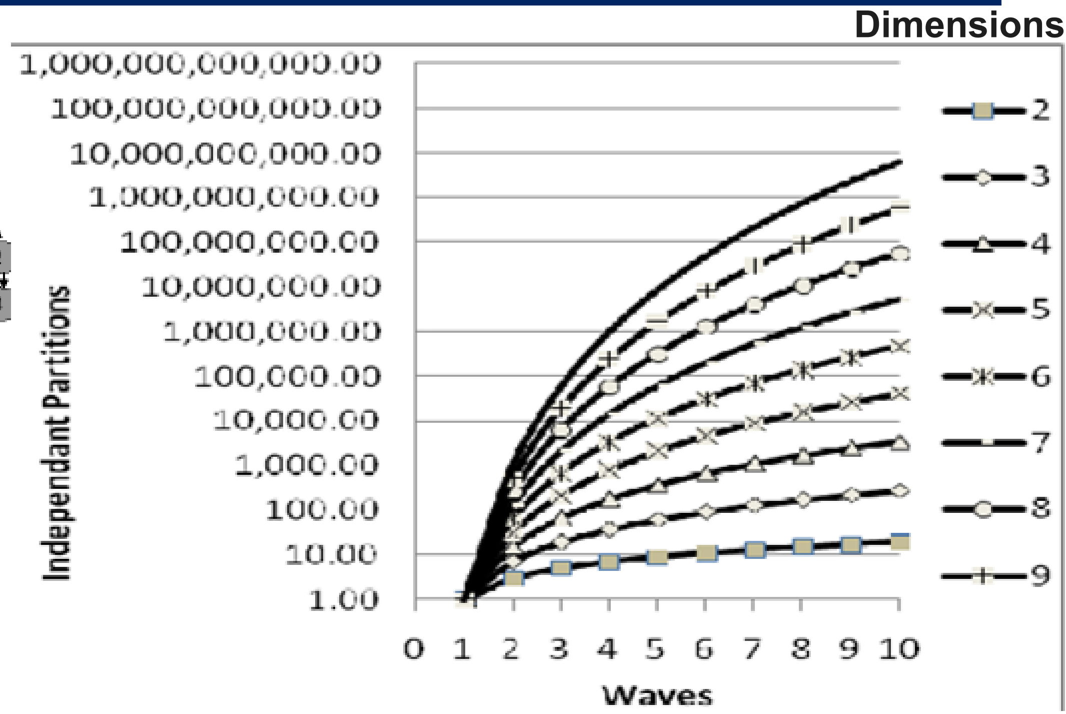
Higher Dimensions

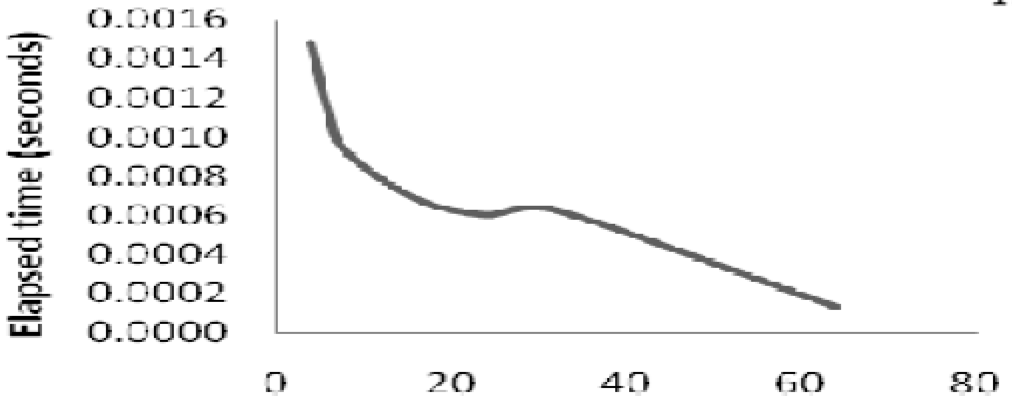


a.



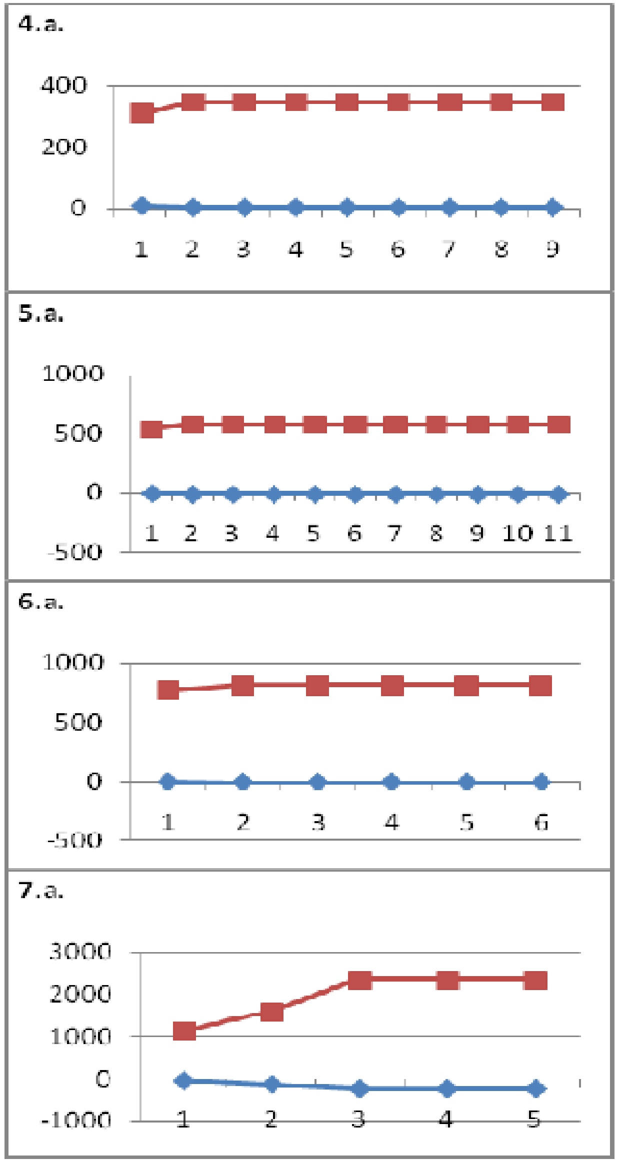
b.



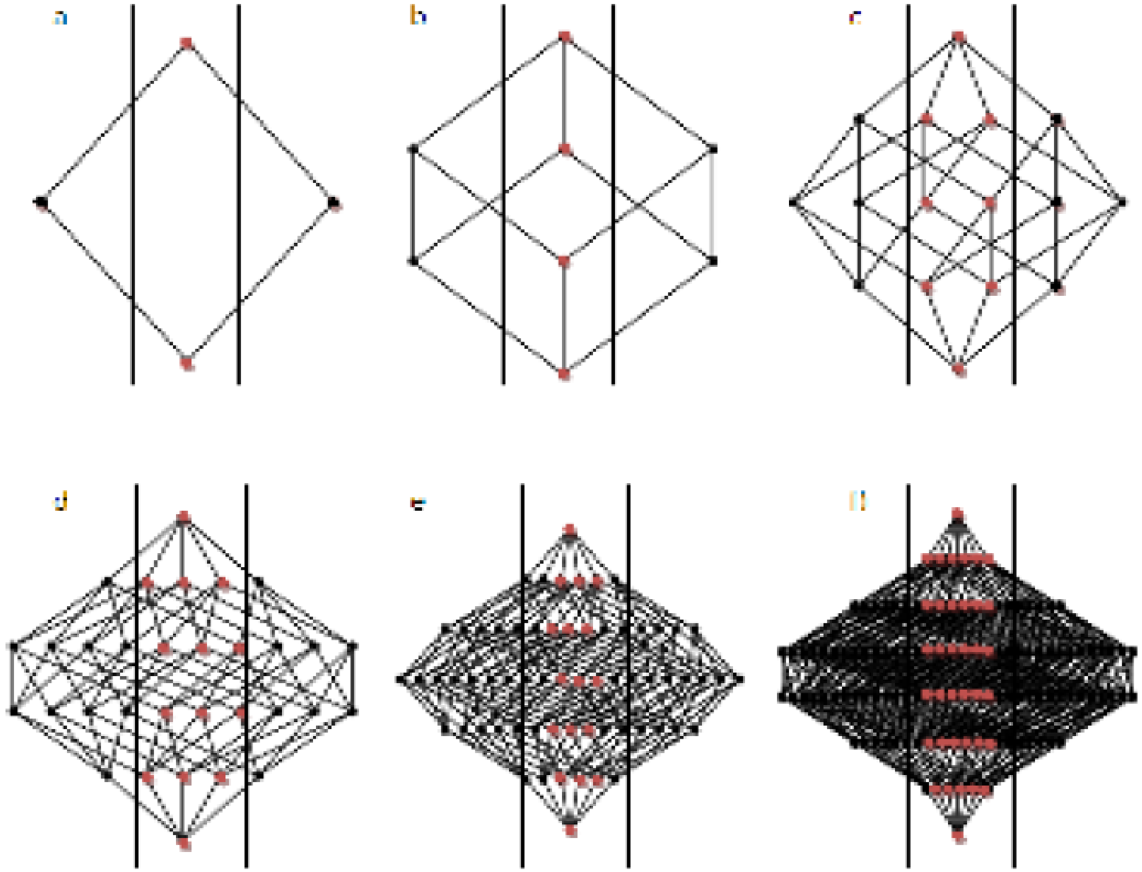


M. Helal, H. El-Gindy, L. Mullin, and B. Gaeta, ‘Parallelizing Optimal Multiple Sequence Alignment by Dynamic Programming’, Dec. 2008, pp. 669–674, doi: 10.1109/ISPA.2008.93.

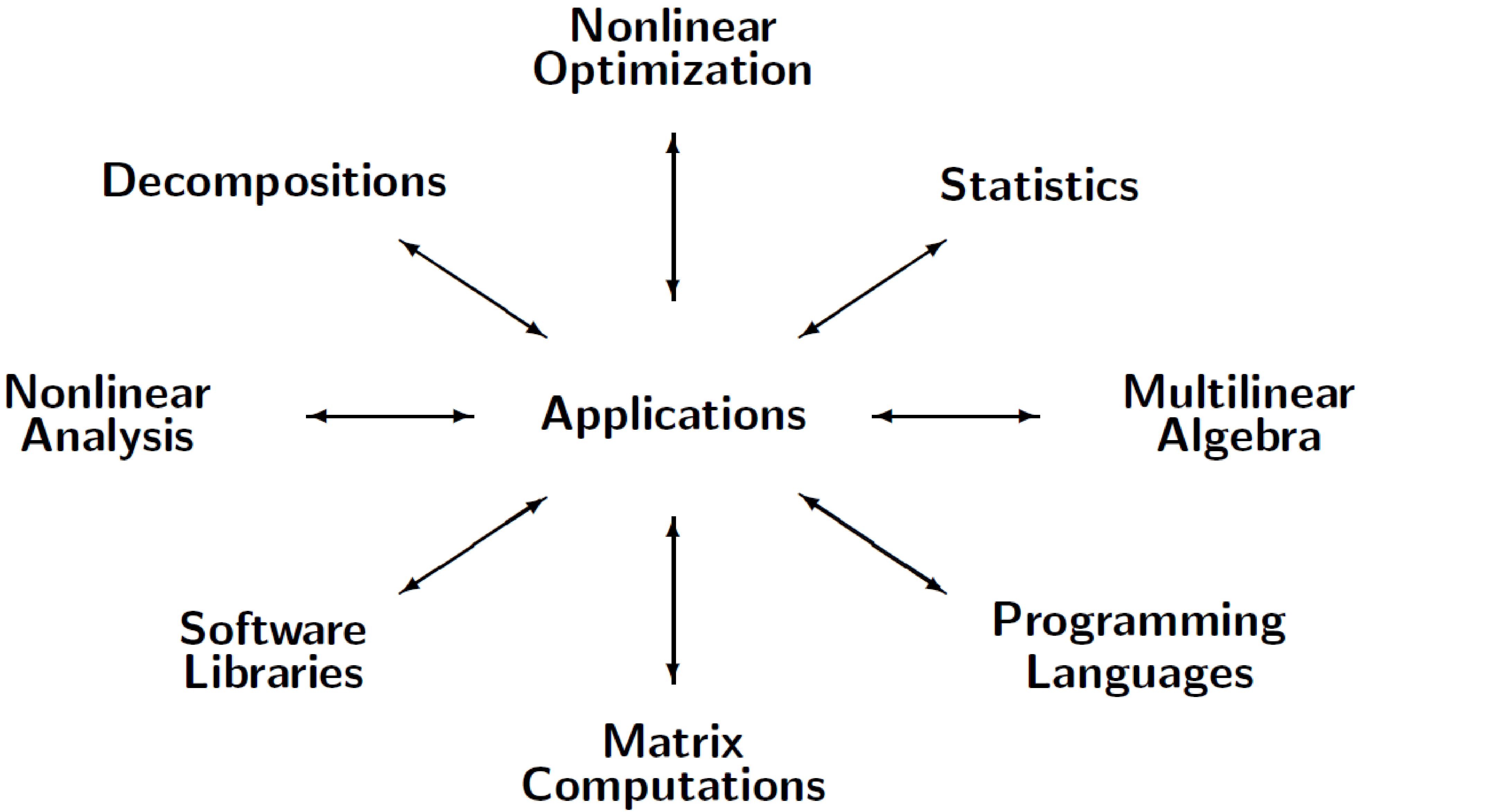
Tensor Partitioning Example for Multiple Sequence Alignment Search Space Reduction



Performance results for the conducted experiments illustrating the alignment scoring accuracy (Red: Entropy; Blue: Sum of Pairs score) over the change of the ε value on the x-axis.



M. Helal, L. Mullin, J. Potter, and V. Sintchenko, ‘Search Space Reduction Technique for Distributed Multiple Sequence Alignment’, Oct. 2009, pp. 219–226, doi: 10.1109/NPC.2009.43.


Nonlinear Optimization
Decompositions
Statistics
Nonlinear Analysis
Applications
Multilinear Algebra
Software Libraries
Programming Languages
Matrix Computations

Challenges and Future Trends

- Hierarchical code that works invariant of dimension and shape (attempted this in my MSc and PhD experiments). More modular APIs for analytics are required. Others have developed libraries for various analytics such as tensorly interface to PyTorch, Keras and TensorFlow, tensorbox toolbox in matlap,
- Automating code generation such as the Matrix/Vector correctness proof and partitioning code generated in Spark - FLAME code-skeleton generator (http://edx-org-utaustinx.s3.amazonaws.com/UT501x/Spark/index.html)
- Developing multilinear extensions of graph-embedding algorithms such as Isomap.
- Since many tensor decomposition approaches are iterative and not closed formula, more work on optimising the initialisation, projection order and the stopping criteria.
- Developing tensor LAPACK with cookbooks describing literature on the suitability or optimality of one model over another.
- These packages require non-functional requirements such as portability, reusability, reliability, correctness, and modularity especially on massively parallel multi-core architectures. Although the deepening memory hierarchy and architectural heterogeneity would be challenging.
- Addressing the issue of floating point stability in tensor computations

For more information:

Please check my research page on my website, and hopefully I will update it as I go:

http://www.manalhelal.com/research/

Any Questions?

University of Hertfordshire UH