Exceptional service in the national interest

Efficient Parallel Software for Tucker Decompositions of Dense Tensors

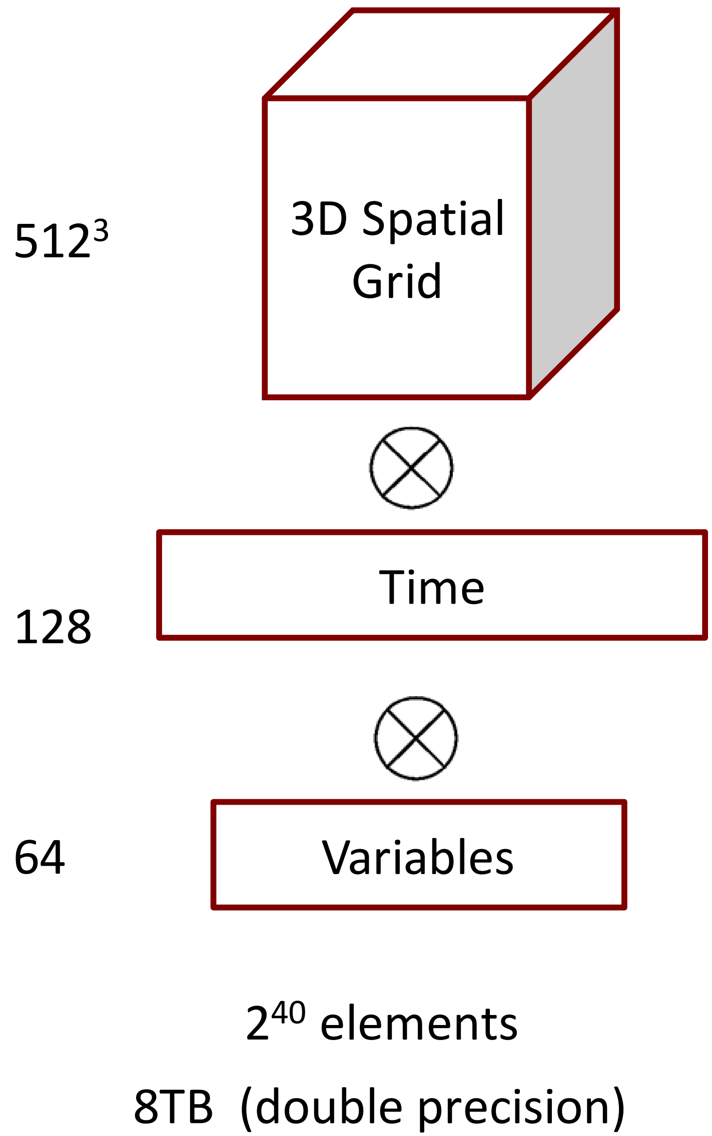
Alicia Klinvex[1], Grey Ballard[2], Tamara G. Kolda[1], Hemanth Kolla[1]

[1]Sandia National Laboratories
[2]Wake Forest University

SIAM CSE '17, Atlanta, March 2, 2017

Sandia National Laboratories is a multi-mission laboratory managed and operated by Sandia Corporation, a wholly owned subsidiary of Lockheed Martin Corporation, for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-AC04-94AL85000. SAND NO SAND2017-2236C

Motivation: Advanced Simulations and Experiments Deluged by Data

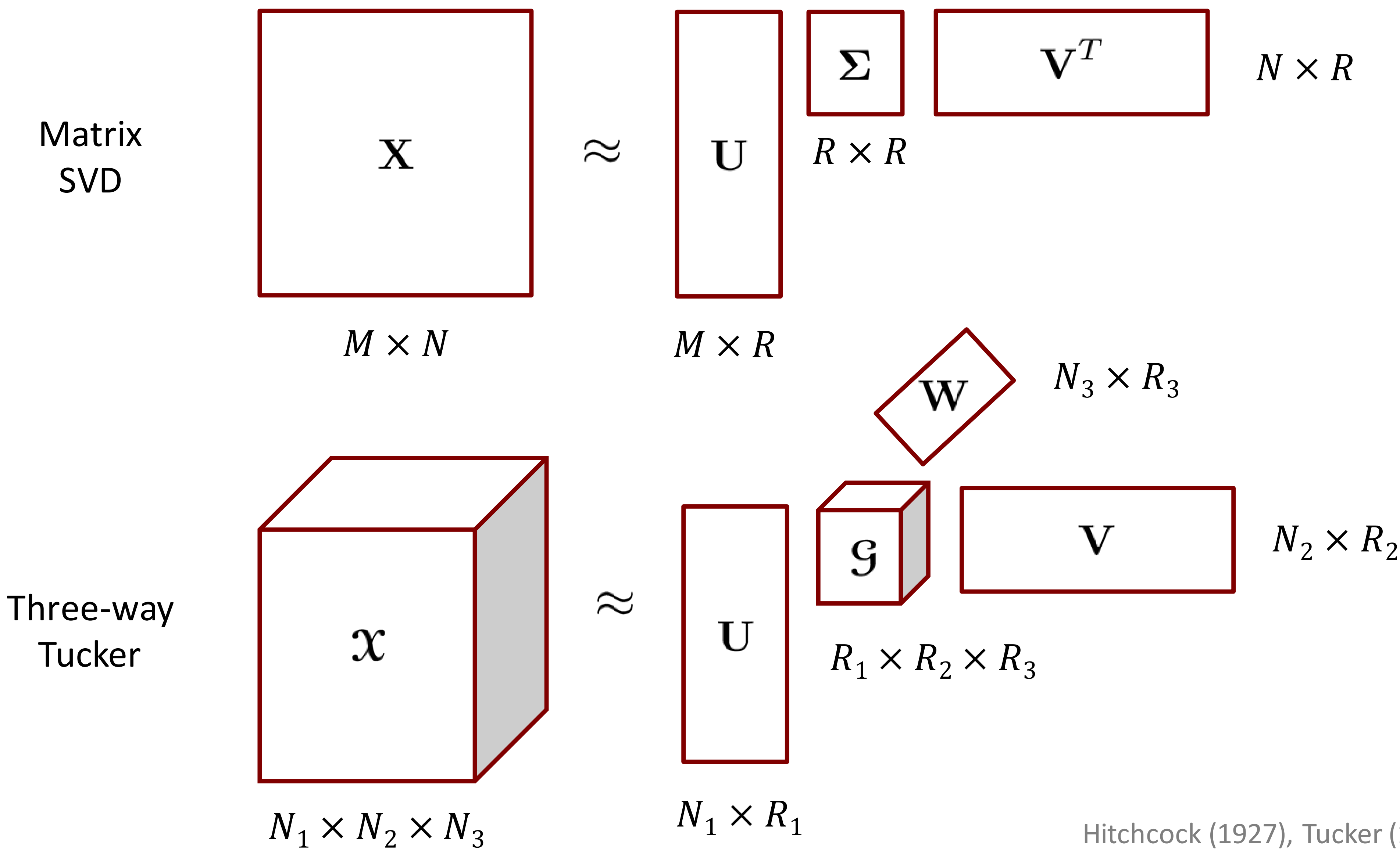


- Combustion simulations
  - S3D code uses direct numerical simulation
  - Gold standard for comparisons, but…
  - Single experiment produces terabytes of data
  - Storage limits spatial, temporal resolutions
  - Difficult to analyze or transfer data
- Goal: to compress this data using math

Outline

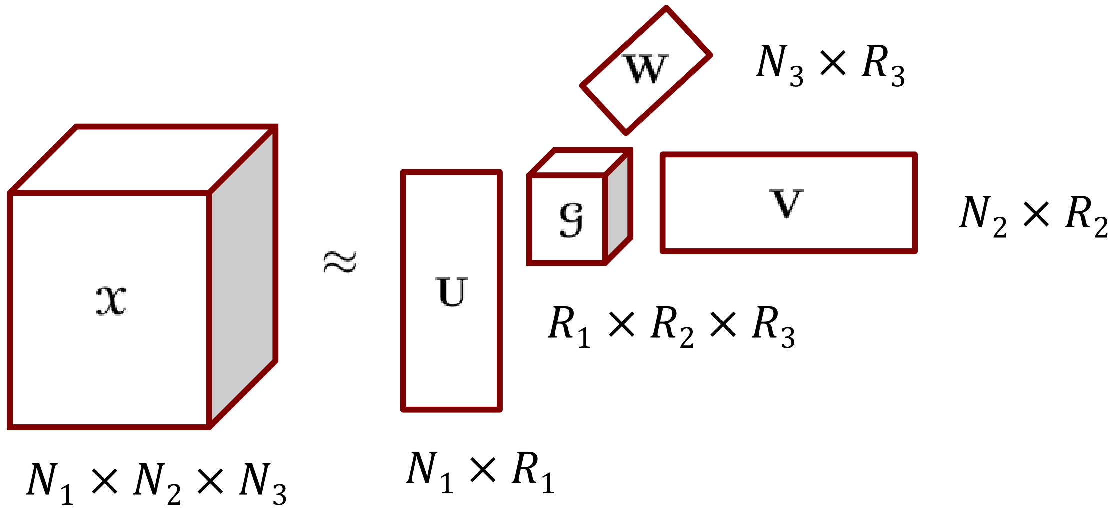
- Tucker decomposition definition
- ST-HOSVD algorithm
- TuckerMPI implementation
- Combustion simulation results

Tucker Compression: Extends the Matrix SVD to Multiway Arrays



Hitchcock (1927), Tucker (1966)

Tucker Compression (3-way)



$$\mathcal{X} \approx \mathcal{G} \times_1 \mathbf{U} \times_2 \mathbf{V} \times_3 \mathbf{W}$$

$\mathcal{G}$ = "Core Tensor" = Reduced representation, determined by factor matrices

$\mathbf{U}, \mathbf{V}, \mathbf{W}$ = "Factor Matrices" = Orthogonal matrices spanning high-variance subspaces

$\times_k$ = Tensor-times-matrix in mode k

Tucker Compression (d-way)

$$\underbrace{\mathcal{X}}_{N_1 \times N_2 \cdots \times N_d} \approx \underbrace{\mathcal{G}}_{R_1 \times R_2 \cdots \times R_d} \times_1 \underbrace{\mathbf{U}^{(1)}}_{N_1 \times R_1} \times_2 \underbrace{\mathbf{U}^{(2)}}_{N_2 \times R_2} \cdots \times_d \underbrace{\mathbf{U}^{(d)}}_{N_d \times R_d}$$

$\mathcal{G}$ = "Core Tensor" = Reduced representation, determined by factor matrices

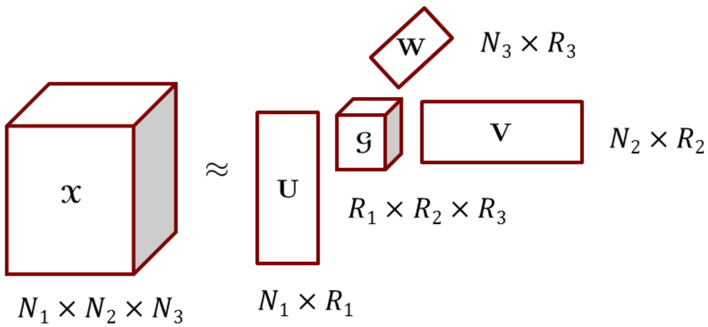
$\mathbf{U}^{(k)}$ = kth "Factor Matrix" = Orthogonal matrix spanning high-variance subspaces

$\times_k$ = Tensor-times-matrix in mode k

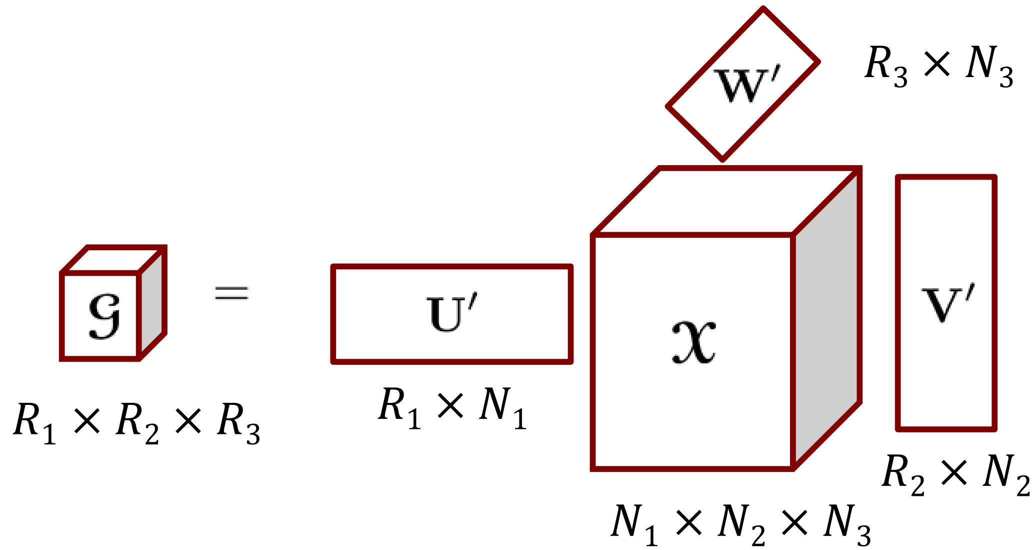
Compression Ratio: $C \approx \prod_{k=1}^{d} \frac{N_k}{R_k}$

Does not include factor matrices

Choosing Tucker Ranks to Retain Accuracy



Find orthogonal matrices $\mathbf{U}, \mathbf{V}, \mathbf{W}$ that reduce the size of tensor but retain its "mass"



For a given relative error ϵ, choose projection ranks R_1, R_2, and R_3 such that:

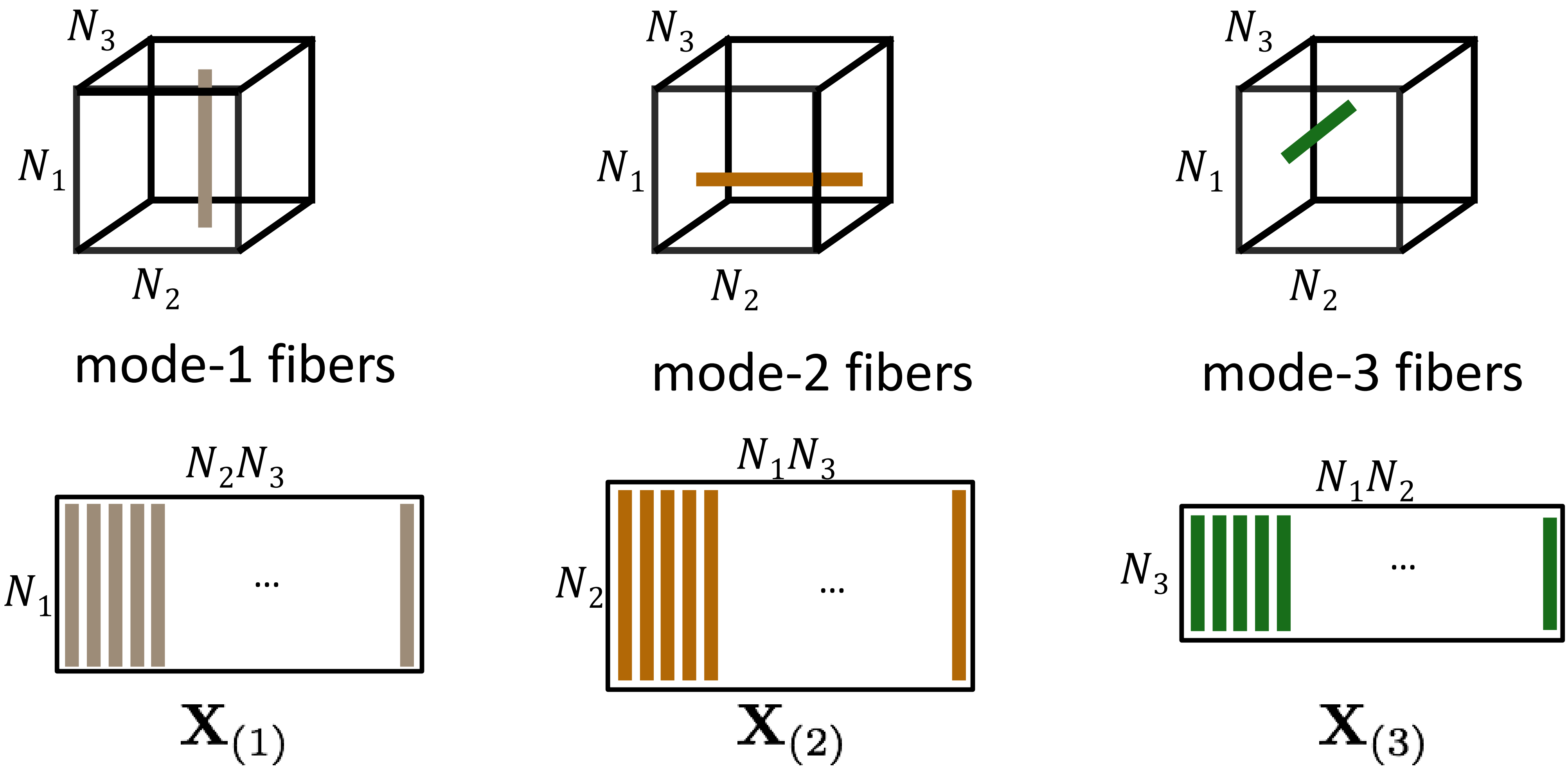
$$\|\mathcal{X} - (\mathcal{G} \times_1 \mathbf{U} \times_2 \mathbf{V} \times_3 \mathbf{W})\| \leq \epsilon \|\mathcal{X}\|$$

Core tensor satisfies: $\mathcal{G} = \mathcal{X} \times_1 \mathbf{U}' \times_2 \mathbf{V}' \times_3 \mathbf{W}'$

$$\|\mathcal{X}\|^2 - \|\mathcal{G}\|^2 \leq \epsilon^2 \|\mathcal{X}\|^2$$

Mass retention

Matricization/Unfolding: Rearranging a Tensor as a Matrix

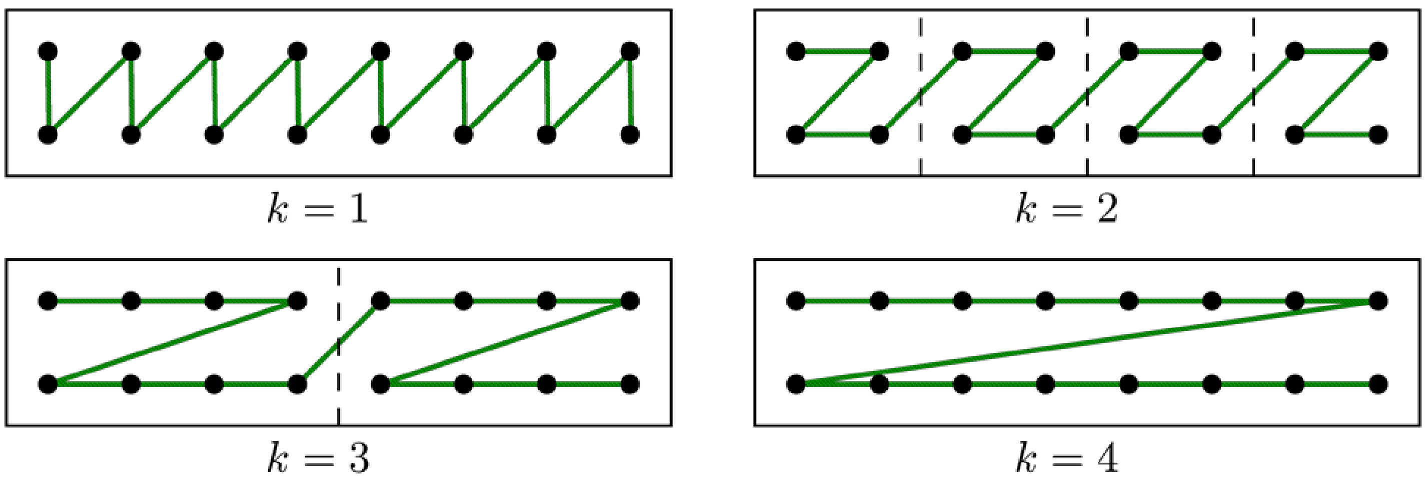


Mathematically convenient expression, but do not want to explicitly form the unfolded matrix due to the expense of the *data movement*.

No Data Movement in Local Unfolding – Block Structure

- Assume local data is stored so that mode-1 unfolding is in column major order
- All unfoldings are blocked, with different block sizes
- Rather than rearrange data (standard practice), exploit structure with block operations

Local Layout: 2 x 2 x 2 x 2 (4-way Tensor)



W. Austin, G. Ballard, and T. G. Kolda, ***Parallel Tensor Compression for Large-Scale Scientific Data***, IPDPS'16 (arXiv:1510.06689)

Algorithm: ST-HOSVD (3-way)

Vannieuwenhoven, Vandebril, Meerbergen (SISC 2012)

1. Choose $\mathbf{U}$ with projection rank R_1 such that: $\|\mathbf{X}_{(1)}\|^2 - \|\mathbf{U}'\mathbf{X}_{(1)}\|^2 \leq \epsilon^2 \|\mathcal{X}\|^2/3$
   a) Compute gram matrix: $\mathbf{X}_{(1)}\mathbf{X}_{(1)}'$
   b) Use eigendecomposition of N_1 x N_1 matrix to choose R_1
   c) Set $\mathbf{U}$ = R_1 leading eigenvectors of gram matrix
2. Shrink to size R_1 x N_2 x N_3: $\mathcal{Y} = \mathcal{X} \times_1 \mathbf{U}'$
3. Choose $\mathbf{V}$ with projection rank R_2 such that: $\|\mathbf{Y}_{(2)}\|^2 - \|\mathbf{V}'\mathbf{Y}_{(2)}\|^2 \leq \epsilon^2 \|\mathcal{X}\|^2/3$
   a) Compute gram matrix: $\mathbf{Y}_{(2)}\mathbf{Y}_{(2)}'$
   b) Use eigendecomposition of N_2 x N_2 matrix to choose R_2
   c) Set $\mathbf{V}$ = R_2 leading eigenvectors of gram matrix
4. Shrink to size R_1 x R_2 x N_3: $\mathcal{Z} = \mathcal{Y} \times_2 \mathbf{V}'$
5. Choose $\mathbf{W}$ with projection rank R_3 such that: $\|\mathbf{Z}_{(3)}\|^2 - \|\mathbf{W}'\mathbf{Z}_{(3)}\|^2 \leq \epsilon^2 \|\mathcal{X}\|^2/3$
   a) Compute gram matrix: $\mathbf{Z}_{(3)}\mathbf{Z}_{(3)}'$
   b) Use eigendecomposition of N_3 x N_3 matrix to choose R_3
   c) Set $\mathbf{W}$ = R_3 leading eigenvectors of gram matrix
6. Shrink to size R_1 x R_2 x R_3: $\mathcal{G} = \mathcal{Z} \times_3 \mathbf{W}'$

Mass retention

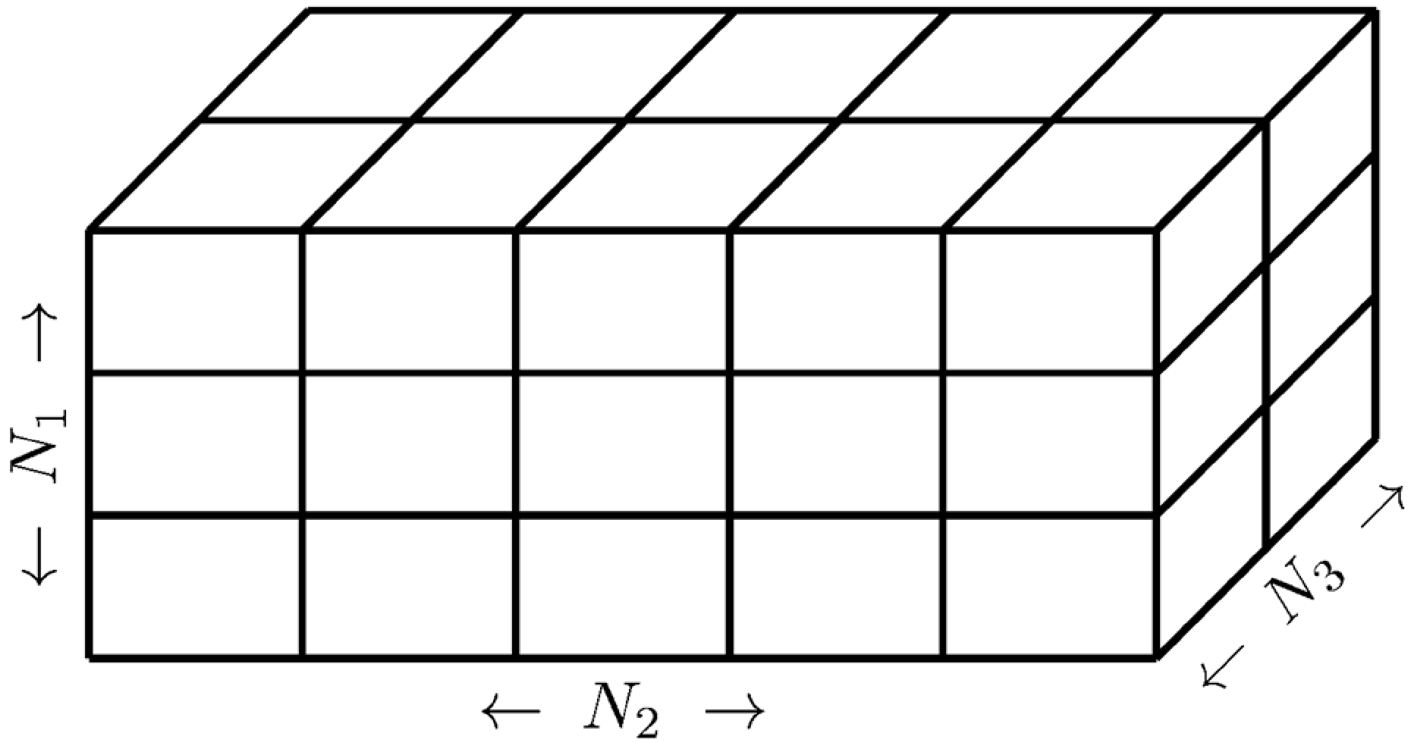
Key Kernels

- Three main steps for mode k
  - **GRAM** – Compute matrix product of unfolded tensor with itself.
    - Unfolded matrix size is N_k x $(R_1 \cdots R_{k\text{-}1} N_{k+1} \cdots N_d)$.
    - Result is N_k x N_k
  - **EVECS** – Compute eigenvalues and eigenvectors of N_k x N_k gram matrix. Use this to determine R_k
    - Call LAPACK, since matrix is small
  - **TTM** – Tensor-times-matrix to shrink mode k from size N_k to size R_k.
    - Input is size R_1 x $\cdots$ x N_k x N_{k+1} x $\cdots$ x N_d
    - Result is size R_1 x $\cdots$ x R_k x N_{k+1} x $\cdots$ x N_d
- These can be viewed as matrix operations

Parallel Tucker Decomposition

For N-way tensor, Cartesian block distribution on N-way processor grid

$$P_1 \times P_2 \times P_3 = 3 \times 5 \times 2$$

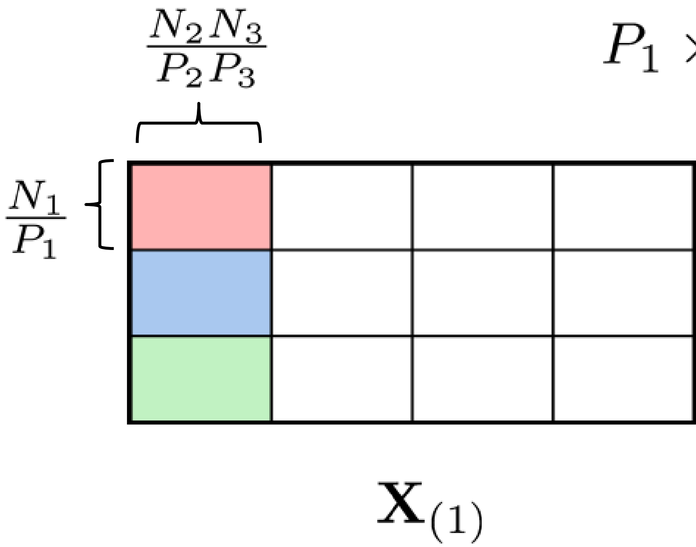


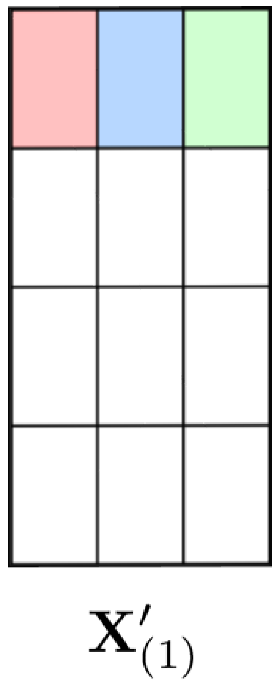
Local block size: $\frac{N_1}{P_1} \times \frac{N_2}{P_2} \times \frac{N_3}{P_3}$

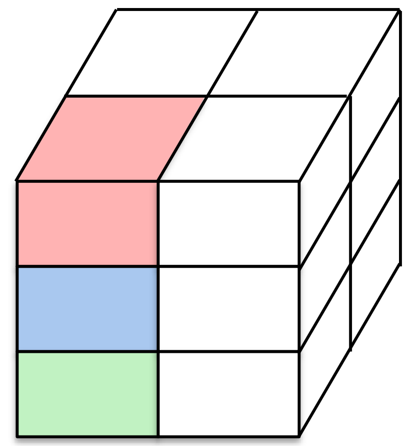
A New Gram Matrix Kernel: Parallel Computation

Sandia National Laboratories

$$P_1 \times P_2 \times P_3 = 3 \times 2 \times 2$$



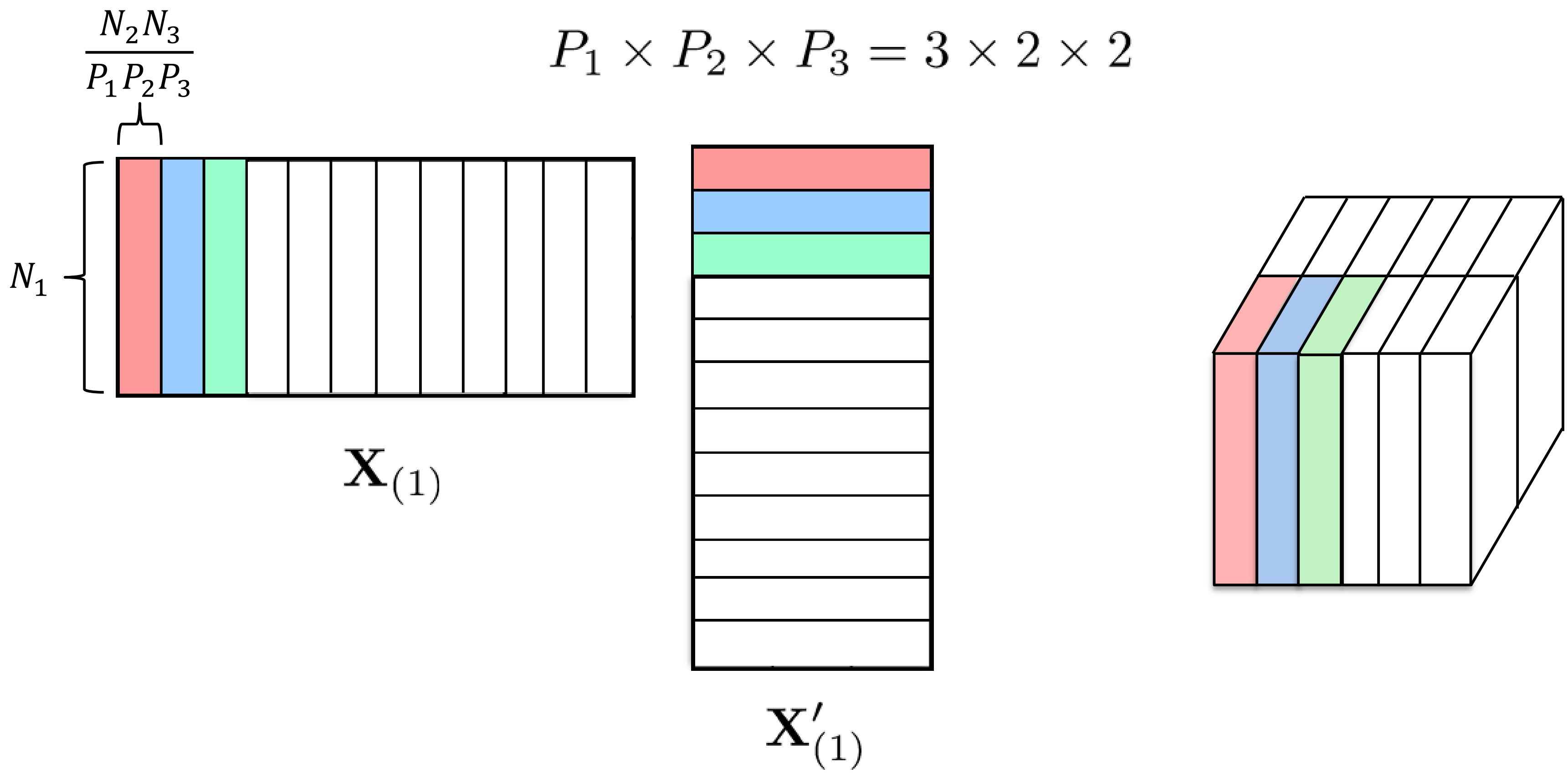




Rearrange data to be block column format

Gram Matrix Kernel: Parallel Computation

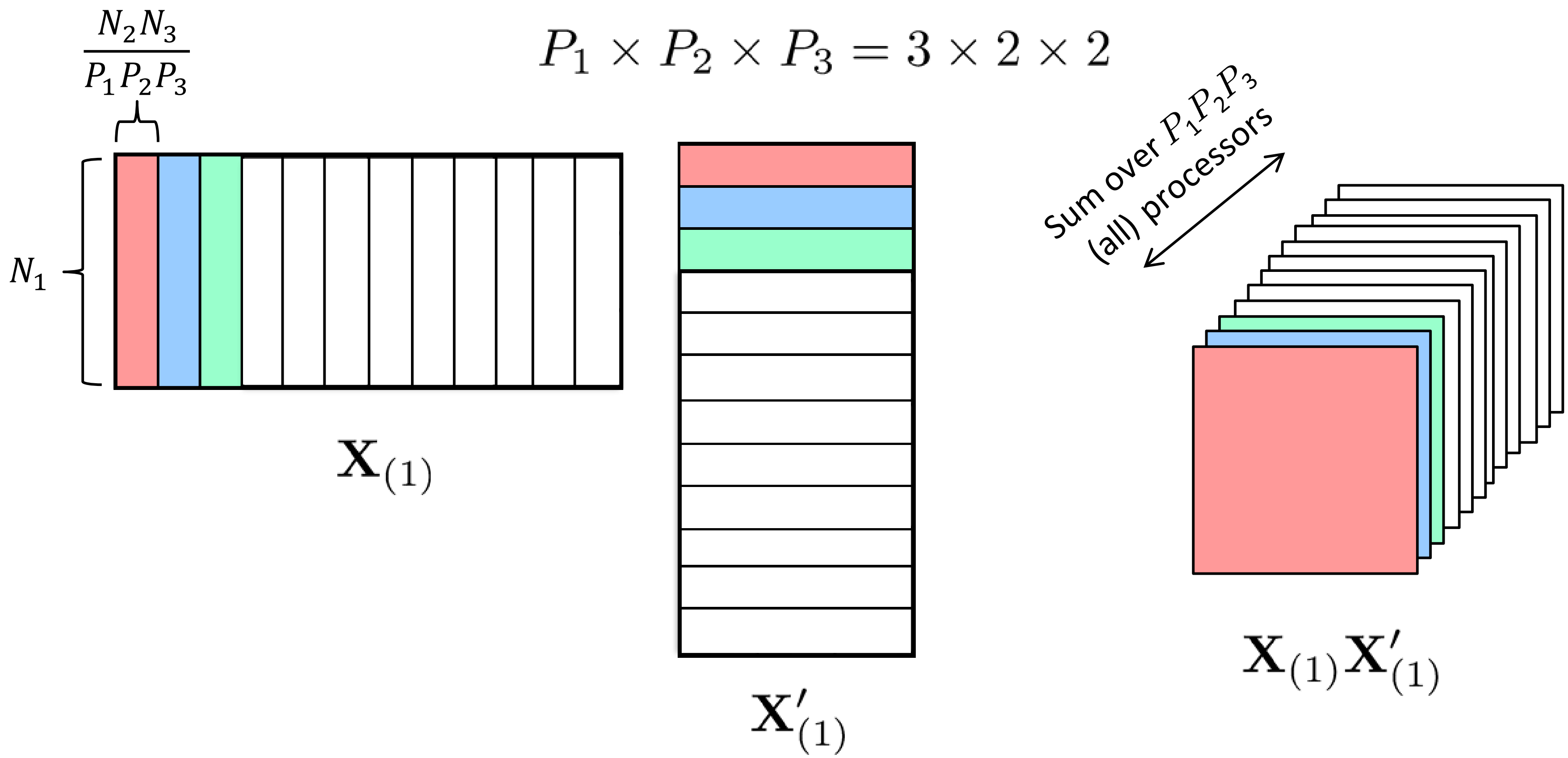
$$P_1 \times P_2 \times P_3 = 3 \times 2 \times 2$$



- Each processor column rearranges its data (with P_1 nodes)

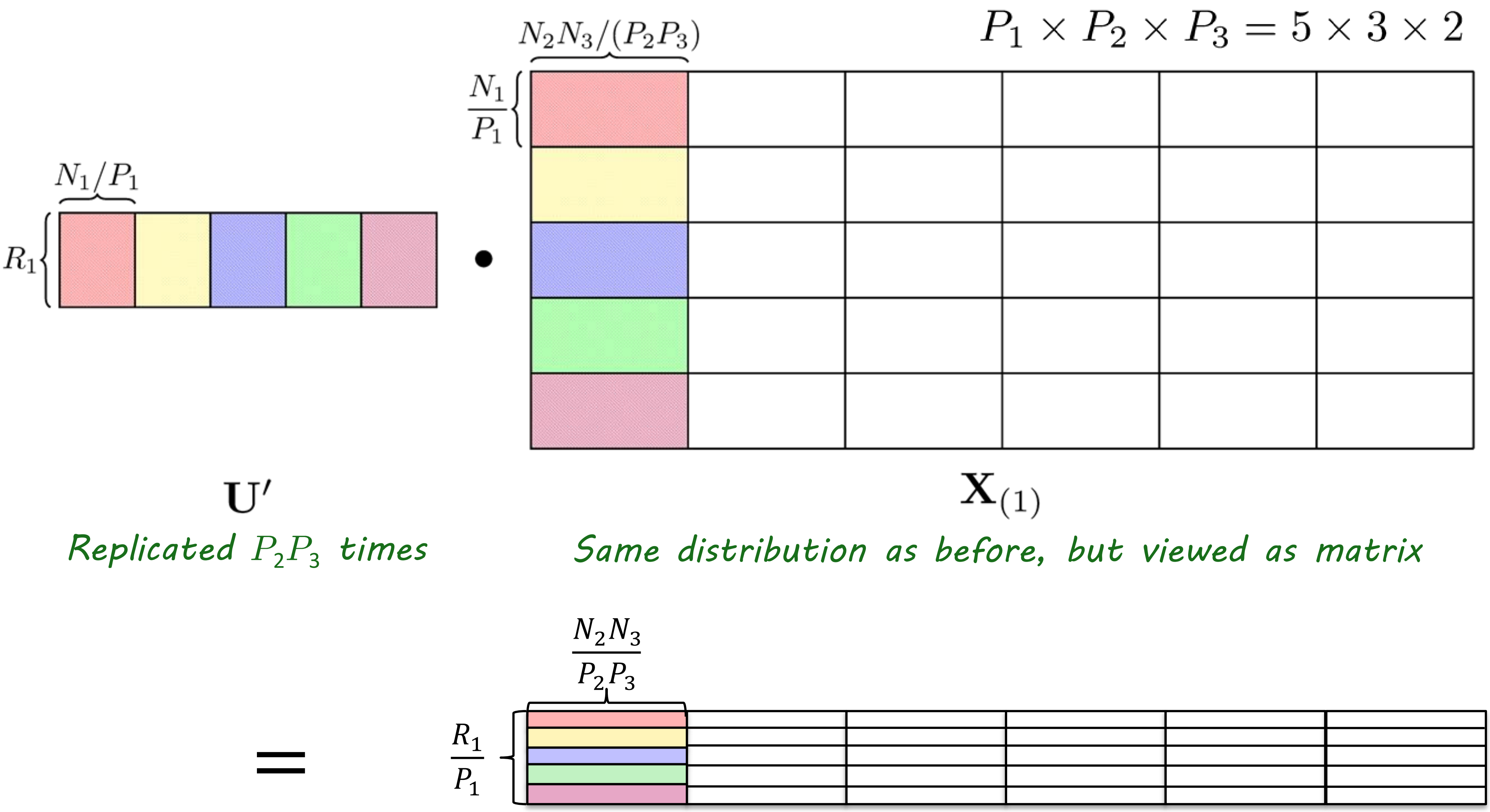
Gram Matrix Kernel: Parallel Computation

$$P_1 \times P_2 \times P_3 = 3 \times 2 \times 2$$

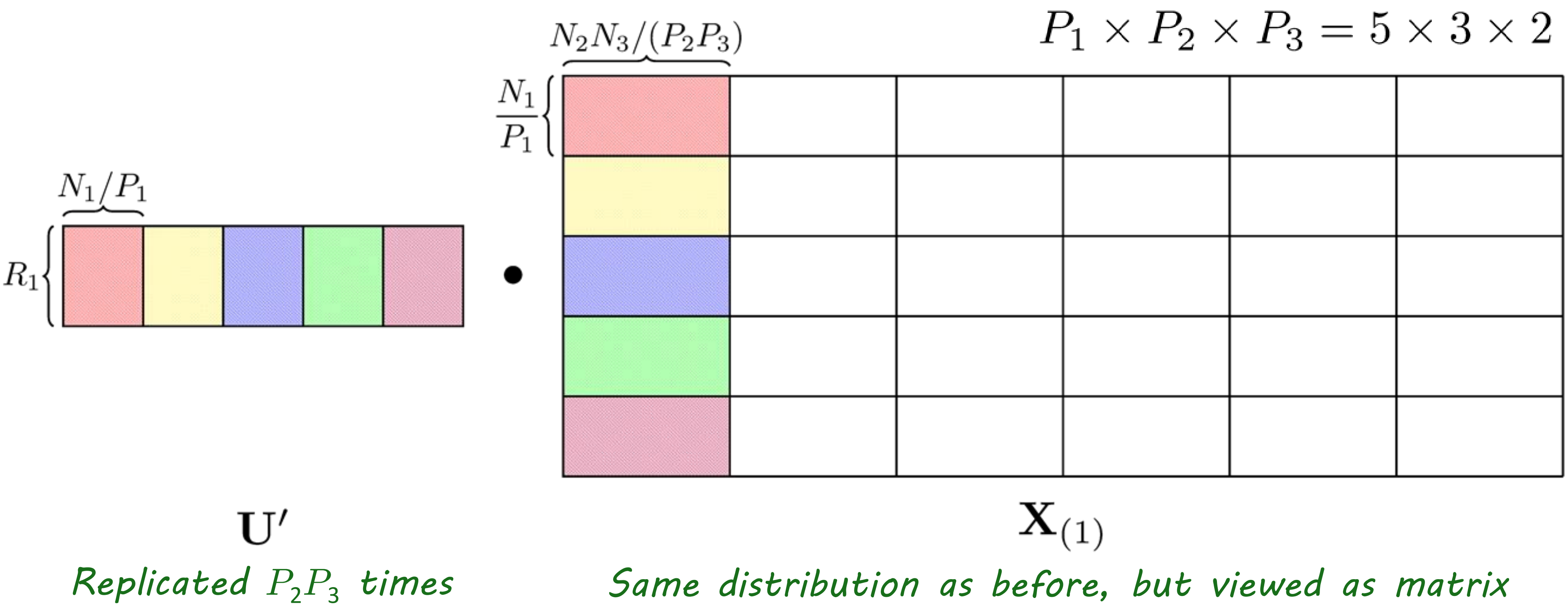


- Each processor column rearranges its data (with P_1 nodes)
- Then computes local outer product
- Sum across all $P_1 P_2 P_3$ groups with all-reduce

Tensor-Times-Matrix Kernel: Parallel Computation



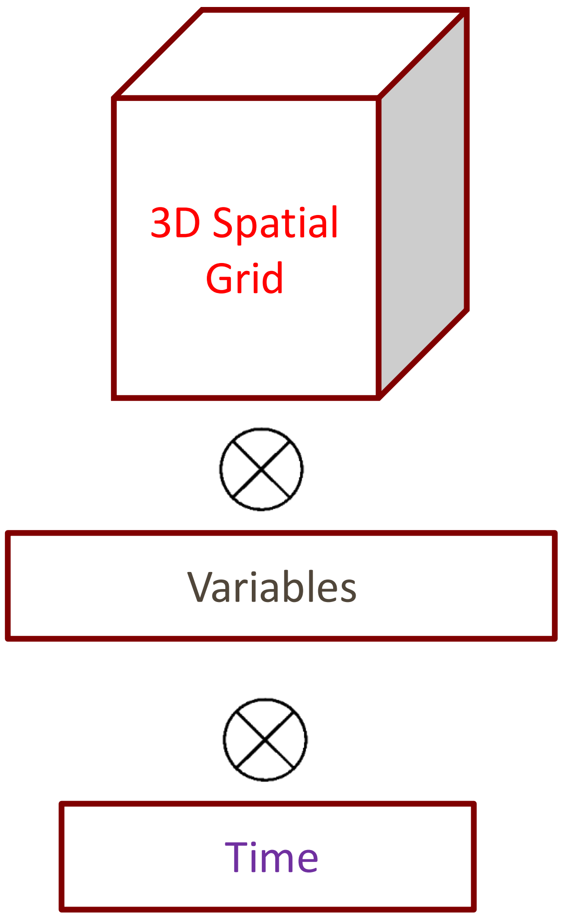
Tensor-Times-Matrix Kernel: Parallel Computation



- Each node locally computes product of its part of $\mathbf{U}$ with its part of $\mathbf{X}_{(1)}$
- Communication is a reduce-scatter on the result within set of P_1 nodes
- Output is distributed same as $\mathbf{X}_{(1)}$ but with a smaller first dimension, i.e., size R_1/P_1

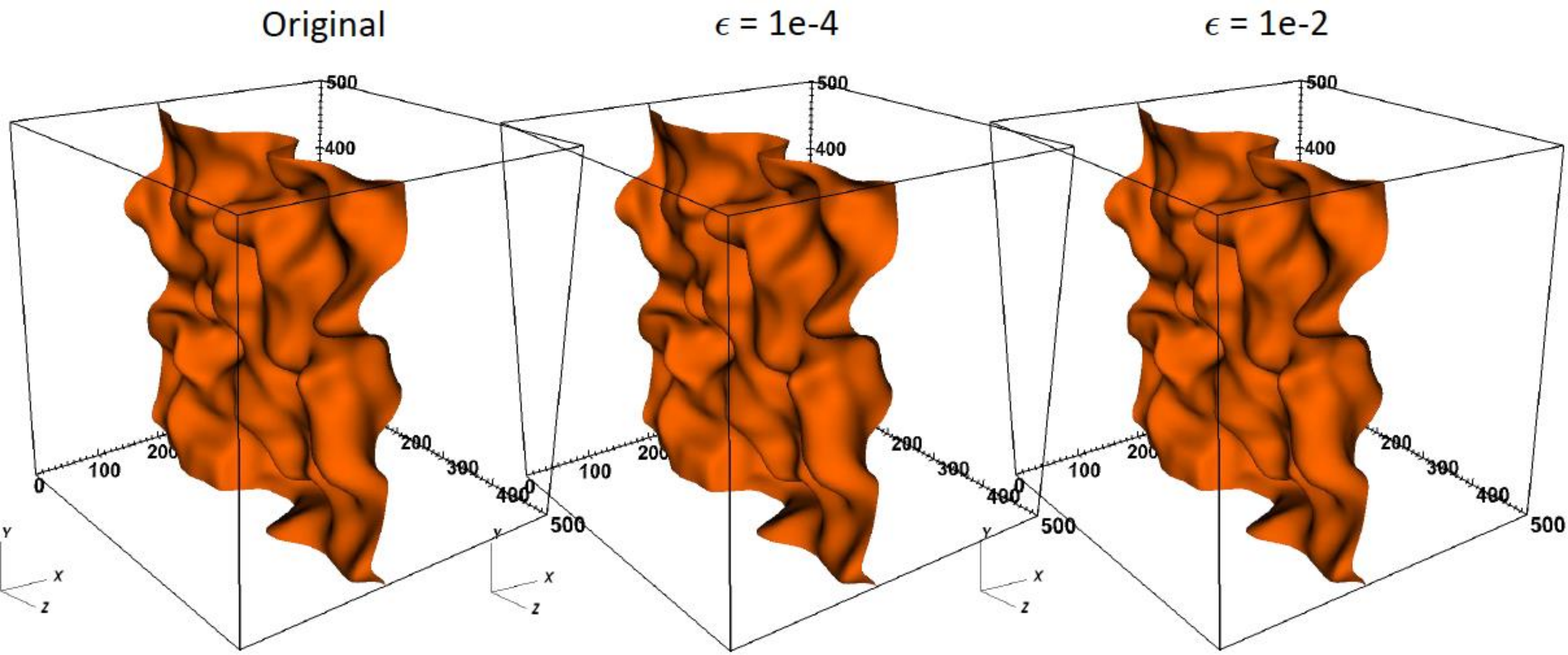
Results

Combustion results



	Original	$\epsilon = 10^{-4}$	$\epsilon = 10^{-2}$
HCCI	672 x 672 x 32 x 626 **67 GB**	330 x 310 x 31 x 199 **(14 X)**	111 x 105 x 22 x 46 **(760 X)**
SP	500 x 500 x 500 x 11 x 400 **4 TB**	95 x 129 x 125 x 7 x 125 **(410 X)**	30 x 38 x 35 x 6 x 11 **(200,000 X)**
JICF	1500 x 2080 x 1500 x 18 x 10 **6 TB**	424 x 387 x 261 x 18 x 10 **(110 X)**	90 x 61 x 48 x 13 x 6 **(40,000 X)**

Original vs reconstructed



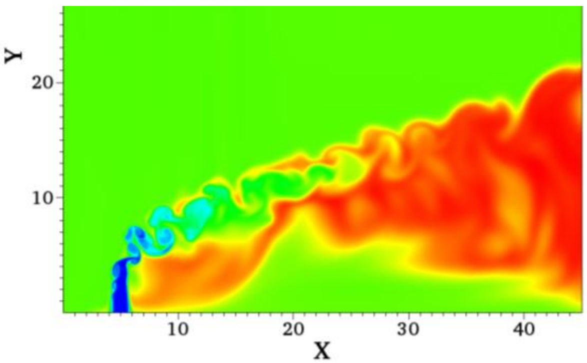
400X Compression

200,000X Compression

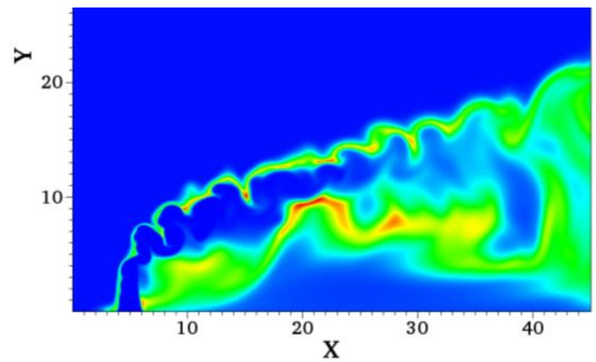
Temperature of 4 TB (SP) dataset at a single timestep

Original vs reconstructed

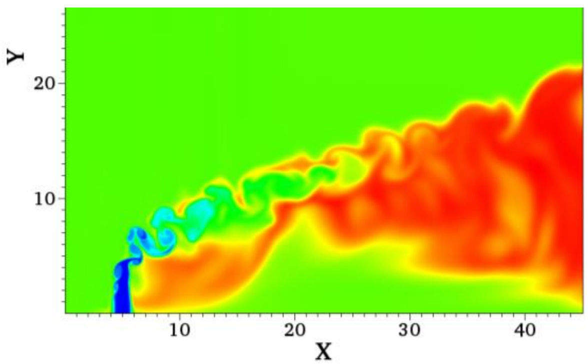
Temperature | OH concentration



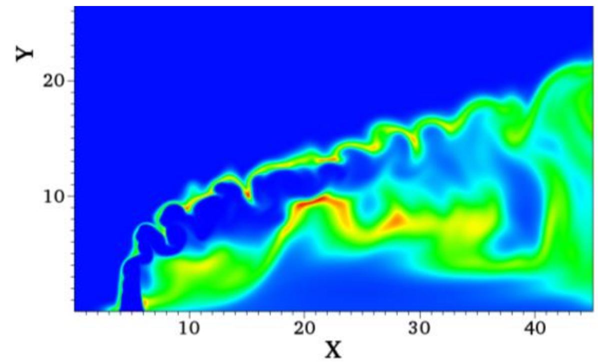
Original

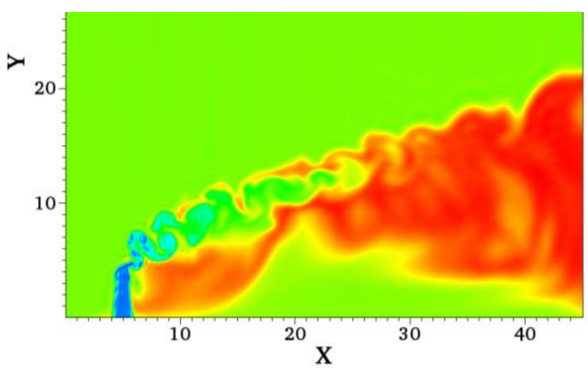


- 6 TB (JICF) dataset
- single timestep
- slice along z direction

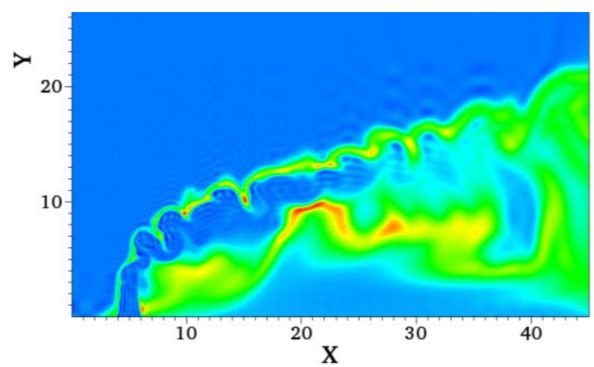


$$\epsilon = 10^{-4}$$
(110X)

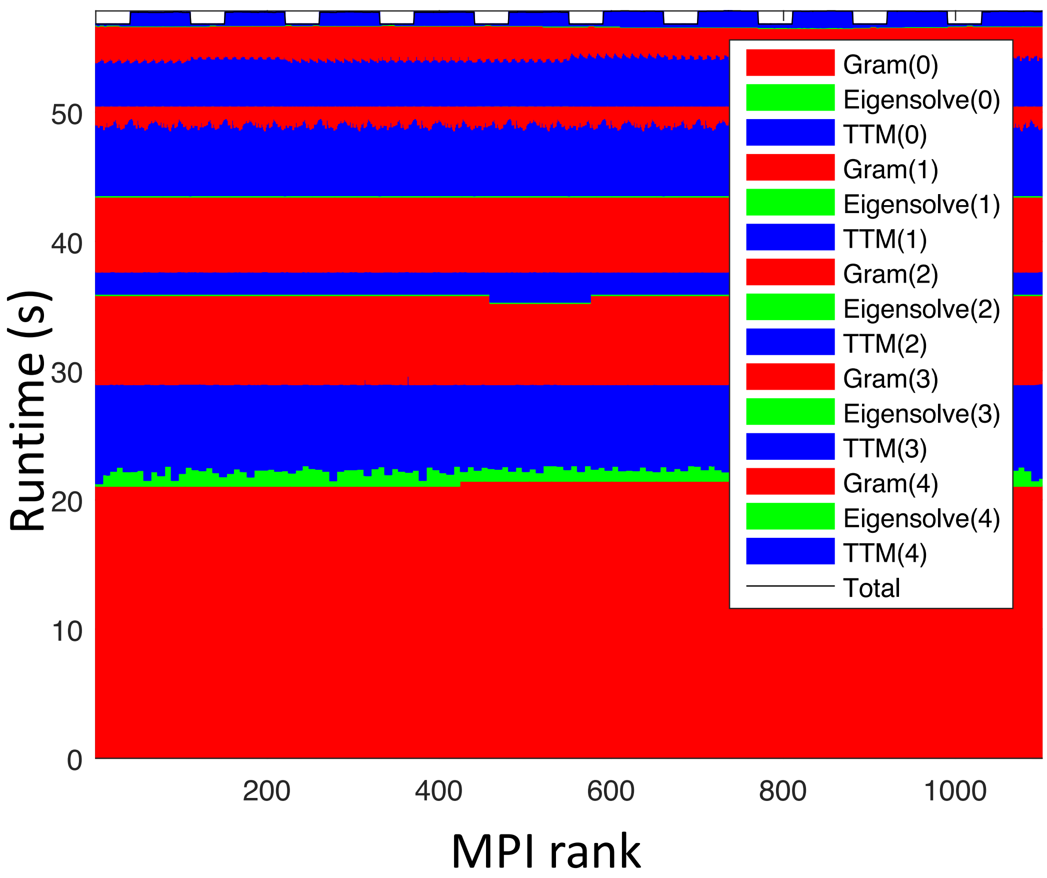




$$\epsilon = 10^{-2}$$
(40,000X)



Parallel TuckerMPI performance



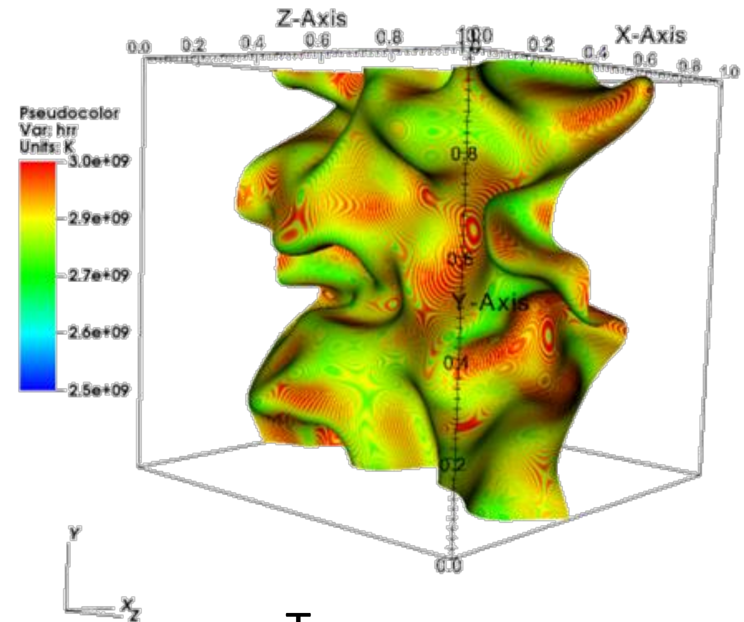
- Total of 55s; 1100 cores.
- 4.4TB -> 10GB (410X).
- Bulk of time is in first mode (GRAM computation).
- **Time for I/O is order of magnitude greater (~450s)**

Key Feature: Need Only Do Partial Reconstruction on Laptops, etc.

Reconstruction requires as much space as the original data!

$$\hat{\mathcal{X}} = \mathcal{G} \times_1 \mathbf{U}^{(1)} \times_2 \mathbf{U}^{(2)} \times_3 \mathbf{U}^{(3)} \times_4 \mathbf{U}^{(4)} \times_5 \mathbf{U}^{(5)}$$

$N_1 \times N_2 \times N_3 \times N_4 \times N_5$



But we can just reconstruct the portion that we need at the moment:

$$\bar{\mathcal{X}} = \mathcal{G} \times_1 \mathbf{U}^{(1)} \times_2 \mathbf{U}^{(2)} \times_3 \mathbf{C}^{(3)T}\mathbf{U}^{(3)} \times_4 \mathbf{C}^{(4)T}\mathbf{U}^{(4)} \times_5 \mathbf{C}^{(5)T}\mathbf{U}^{(5)}$$

$N_1 \times N_2 \times \frac{N_3}{2} \times 1 \times 1$

$$\mathbf{C}^{(3)} = \begin{bmatrix} 1/2 & 0 & \cdots & 0 \\ 1/2 & 0 & \cdots & 0 \\ 0 & 1/2 & \cdots & 0 \\ 0 & 1/2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \end{bmatrix} \qquad \mathbf{C}^{(4)} = \begin{bmatrix} 0 \\ \vdots \\ 1 \\ 0 \\ \vdots \end{bmatrix} \qquad \mathbf{C}^{(5)} = \begin{bmatrix} 0 \\ \vdots \\ 1 \\ 0 \\ \vdots \end{bmatrix}$$

Downsample

Pick single variable

Pick single time step

Partial reconstruction of JICF dataset

- Size of original data: 6 TB
- Reconstructing a single variable at a single timestep along a slice of the z axis (*24 MB*)
- Ran on a single node with 128 GB of RAM

Error	1e-2	1e-4
Core tensor size	156 MB	57 GB
Read	.1 s	139 s
Reconstruct	.1 s	21 s
Write	6 s	.02 s
Required memory	185 MB	58 GB

Software: TuckerMPI

Sandia National Laboratories

`git@gitlab.com:tensors/TuckerMPI.git`

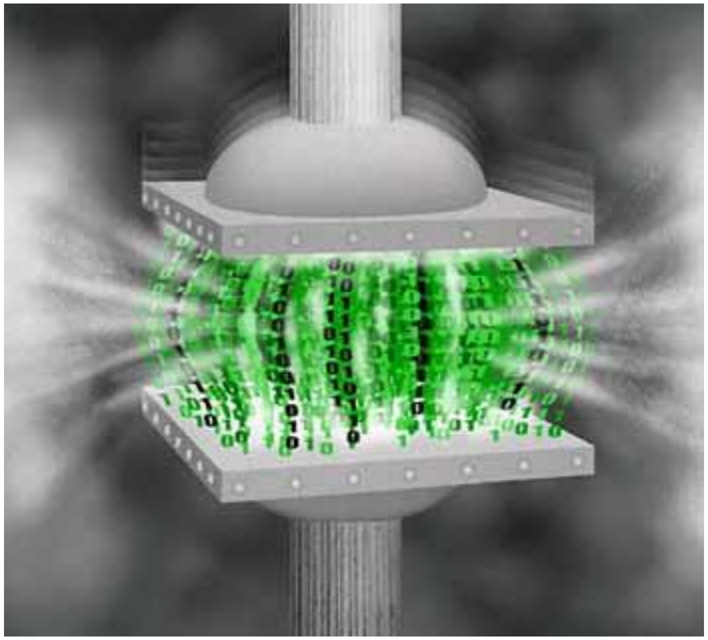
Alicia Klinvex, Woody Austin, Grey Ballard, Hemanth Kolla, Tammy Kolda

- Open source code for computing Tucker compression
- MPI/BLAS/LAPACK/C++11
- Still in development but available for testing
- Looking for new applications and users
- Interested in partnering

Tensor Tucker Decomposition for Compression for Scientific Data

Sandia National Laboratories

- First parallel implementation of Tucker decomposition
  - 5-way data using regular grid
  - Process 4TB data in < 1 min
- Up to 200,000X compression on real-world data
  - Specify desired relative RMSE
  - Discovers latent multi-linear structure
  - Enables "smart" compression rather than discarding data that may be useful
- More work to do…
  - In situ computations
  - Real-time visualization for computational steering, etc.
  - Adaptive and non-uniform grids
  - Experimental data
  - Randomization
  - Extensive testing



http://www.analyticbridge.com/profiles/blogs/how-much-is-big-data-compressible-an-interesting-theorem

Alicia Klinvex
amklinv@sandia.gov

W. Austin, G. Ballard, and T. G. Kolda, ***Parallel Tensor Compression for Large-Scale Scientific Data***, IPDPS'16 (arXiv:1510.06689)