Software for tensor computations: What is happening???

Paolo Bientinesi
pauldj@cs.umu.se

March 7–11, 2022
Dagstuhl Seminar "Tensor Computations: Applications and Optimization"
Schloss Dagstuhl

HPAC High Performance and Automatic Computing
UMEÅ UNIVERSITY
HPC2N

Ongoing survey

"The Landscape Of Software For Tensor Computations"
with C. Psarras, L. Karlsson, J. Li

`https://arxiv.org/pdf/2103.13756.pdf`

DatM: Data Manipulation **EWOps**: Element-Wise Operations **Con**: Contractions
SpecCon: Specific Contractions **Decomp**: Decompositions

ID	Package Name	Functionality					Type	Platform	Language
		DatM	EWOps	SpecCon	Con	Decomp			
0	Acrotensor	–	–	✓	✓	–	D	C, G	C++
1	AdaTM	–	–	✓	–	✓	S	C	C
2	Boost.uBlas.Tensor	✓	✓	✓	✓	–	D	C	C++
3	COGENT	–	–	✓	✓	–	D	G	Python → C
4	COMET	–	–	✓	✓	–	S	C	C++ → C++
5	CoTenGra	–	–	✓	✓	–	D	C, D, G	Python
6	CP-CALS	–	–	✓	–	✓	D	C, G	C++, Mat[i]
7	CSTF	–	–	–	–	✓	S	D	Scala
8	CuTensor	✓	✓	✓	✓	–	D	G	C, CUDA
9	cuTT	✓	–	–	–	–	D	G	C++, CUDA
10	Cyclops	✓	✓	✓	✓	–	S	C, D, G	C++
11	DFacTo	–	–	–	–	✓	S	C, D	C++
12	Eigen Tensor	✓	✓	✓	✓	–	D	C, G	C++
13	ExaTN	✓	✓	✓	✓	✓	D	C, D, G	C++, Py[i]
14	Fastor	✓	✓	✓	✓	–	D	C	C++
15	FTensor	✓	✓	✓	✓	–	D	C	C++
16	Genten	–	–	–	–	✓	D, S	C, G	C++
17	GigaTensor	–	–	–	–	✓	S	C, D	Unknown

ID	Package Name	Functionality					Type	Platform	Language
		DatM	EWOps	SpecCon	Con	Decomp			
18	HPTT	✓	–	–	–	–	D	C	C++, Pythor
19	ITensor	–	✓	✓	✓	✓	D, BS	C, G[x]	C++, Julia
20	libtensor	–	–	✓	✓	–	D, BS	C	C++
21	Ltensor	–	–	✓	✓	–	D	C	C++
22	MATLAB	✓	✓	–	–	–	D	C	Matlab
23	MultiArray	✓	–	–	–	–	D	C	C++
24	multiway	–	–	–	–	✓	D	C	R
25	N-way toolbox	–	–	–	–	✓	D	C	Matlab
26	NCON	–	–	✓	✓	–	D	C	Matlab
27	netcon	–	–	✓	✓	–	D	C	Matlab
28	NumPy	✓	✓	✓	✓	–	D	C	Python
29	Ocean	✓	✓	–	–	–	D	C, G	C, Py[i]
30	ParCube	–	–	–	–	✓	S	C	Matlab
31	ParTensor	–	–	–	–	✓	D	C, G	C++
32	ParTI!	✓	✓	✓	–	✓	S	C, G	C, CUDA, M
33	PLANC	–	–	–	–	✓	D	C, D	C++
34	PLS toolbox	–	–	–	–	✓	D	C	Matlab
35	Pytensor	✓	✓	✓	✓	✓	D, S	C	Python

ID	Package Name	Functionality					Type	Platform	Language
		DatM	EWOps	SpecCon	Con	Decomp			
36	PyTorch	✓	✓	✓	✓	–	D, S	C, G	Python, C+
37	quimb	–	–	✓	✓	–	D	C, D, G	Python
38	rTensor	✓	✓	✓	–	✓	D	C	R
39	rTensor (randomized)	–	–	–	–	✓	D	C	Python
40	scikit-tensor	✓	✓	✓	–	✓	D, S	C	Python
41	Scikit-TT	–	–	–	–	✓	D	C	Python
42	SPALS	–	–	–	–	✓	S	C	C++
43	SPARTan	–	–	–	–	✓	S	C	Matlab
44	SPLATT	–	–	✓	–	✓	S	C, D	C, C++, Oct
45	SuSMoST	–	–	–	–	✓	D	C	Python
46	T3F	✓	✓	–	–	✓	D	C, G	Python
47	TACO	✓	✓	✓	✓	–	D, S	C, G	C++, C++ →
48	TAL_SH	✓	✓	✓	✓	–	D	C, G	C, C++, For
49	TBlis	✓	✓	✓	✓	–	D	C	C++
50	TCCG	–	–	✓	✓	–	D	C	C++
51	TCL	–	–	✓	✓	–	D	C	C++, Pythor
52	TDALAB	–	–	–	–	✓	D, S	C	Matlab, GU
53	TeNPy	–	–	–	–	✓	D	C	Python

ID	Package Name	Functionality					Type	Platform	Language
		DatM	EWOps	SpecCon	Con	Decomp			
54	Tensor Fox	—	—	—	—	✓	D, S	C	Python, Ma
55	Tensor package	—	—	—	—	✓	D	C	Matlab
56	Tensor Toolbox	✓	✓	✓	✓	✓	D, S	C	Matlab
57	tensor_decomposition	—	—	—	—	✓	D	C, D	Python
58	TensorBox	—	—	—	—	✓	D, S	C	Matlab
59	TensorD	—	—	—	—	✓	D	C, G	Python
60	TensorFlow	✓	✓	✓	✓	—	D, S	C, D, G	C++, Pythor
61	TensorLab	—	—	—	—	✓	D, S	C	Matlab
62	TensorLy	✓	✓	✓	✓	✓	D	C, G	Python
63	TensorNetwork	—	—	✓	✓	—	D, S	C, G	Python
64	TensorOperations.jl	✓	✓	✓	✓	—	D	C, G	Julia
65	TensorTrace	—	—	✓	✓	—	D	C	GUI → Py,
66	Three-Way	—	—	✓	✓	✓	D	C	R
67	TiledArray	✓	✓	✓	✓	—	D, BS	C, D	C++
68	tncontract	—	—	✓	✓	—	D	C	Python
69	TNR	—	—	—	—	✓	D	C	Matlab
70	TorchMPS	—	—	✓	✓	—	D	C	Python
71	TT-Toolbox	✓	✓	—	—	✓	D	C, D^x, G^x	Matlab, Pyt

ID	Package Name	Functionality					Type	Platform	Language
		DatM	EWOps	SpecCon	Con	Decomp			
72	TTC	✓	–	–	–	–	D	C	Python → C
73	TTV	–	–	✓	–	–	D	C	C++
74	TVM	–	✓	–	–	–	D, S	C, G	Python
75	Uni10	✓	✓	✓	✓	–	D	C, G^x	C++
76	xerus	–	–	✓	✓	✓	D, S	C,	C++

That is A LOT of libraries for tensor operations!!!

That is A LOT of libraries for tensor operations!!!

- The survey is by no means complete.
  We are still finding libs, and
  NEW libs are released regularly!

That is A LOT of libraries for tensor operations!!!

- The survey is by no means complete.
  We are still finding libs, and
  NEW libs are released regularly!

- Even considering different languages, different data types (sparse vs. dense), different tensor orders, different types of parallelism . . .

  STILL, this is an enormous duplication of effort and functionality!!
  Also, many (severely) suboptimal implementations.

That is A LOT of libraries for tensor operations!!!

- The survey is by no means complete.
  We are still finding libs, and
  NEW libs are released regularly!

- Even considering different languages, different data types (sparse vs. dense), different tensor orders, different types of parallelism ...

  STILL, this is an enormous duplication of effort and functionality!!
  Also, many (severely) suboptimal implementations.

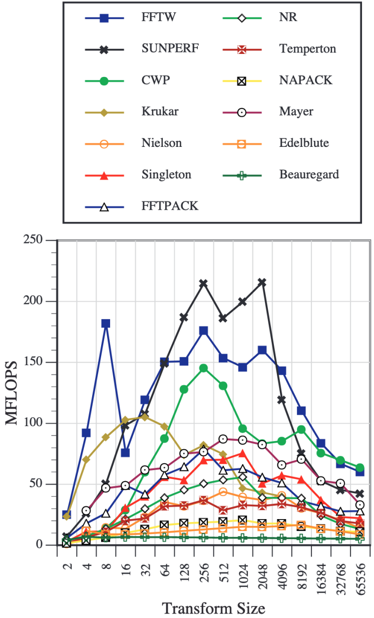
- 1) Why?!
  2) Is there a way out?

Have we seen this before?

Have we seen this before?

- Libraries for computing FFTs, prior to FFTW

  FFT: 1 single op, many different algs, different datatypes, 2 languages;
  ⇒ 1 small team (users) took over



"The Fastest Fourier Transform in the West", MIT-LCS-TR-728

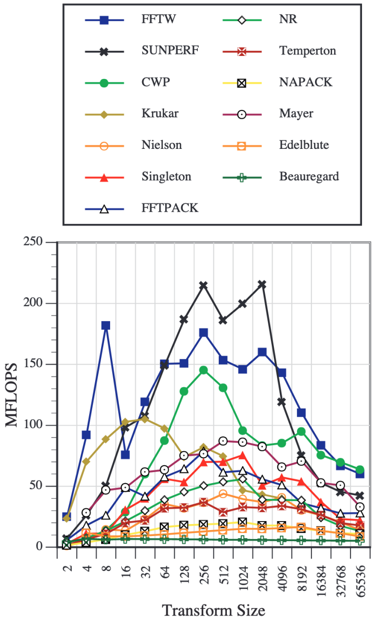
Have we seen this before?

- Libraries for computing FFTs, prior to FFTW

  FFT: 1 single op, many different algs, different datatypes, 2 languages;
  $\Rightarrow$ 1 small team (users) took over

- Libraries for message passing, prior to MPI

  Message passing: Lots of operations, lots of algs, 2 languages;
  $\Rightarrow$ community effort, HPC + vendors, 3-year incubation time



"The Fastest Fourier Transform in the West", MIT-LCS-TR-728

Have we seen this before?

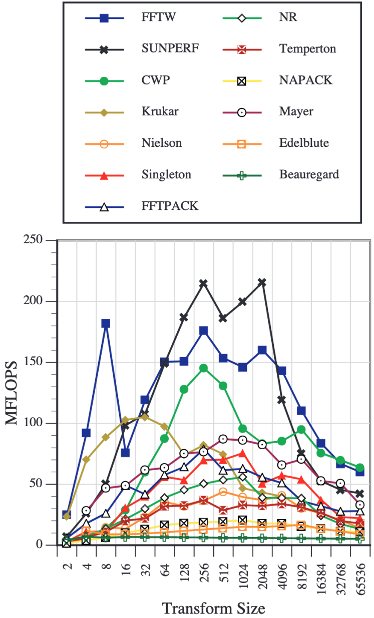
- Libraries for computing FFTs, prior to FFTW

  FFT: 1 single op, many different algs, different datatypes, 2 languages;
  $\Rightarrow$ 1 small team (users) took over

- Libraries for message passing, prior to MPI

  Message passing: Lots of operations, lots of algs, 2 languages;
  $\Rightarrow$ community effort, HPC + vendors, 3-year incubation time

- Different routes to "convergence"
  What about Tensorland?



"The Fastest Fourier Transform in the West", MIT-LCS-TR-728

Matrices vs. Tensors

Historical overview

Linear Algebra Libraries: 1970s

"Basic Linear Algebra Subprograms for FORTRAN usage", ACM TOMS, 1979

BLAS-1

Linear Algebra Libraries: 1980s

BLAS-2: Mat-vec ops, ACM TOMS 1988. BLAS-3: mat-mat ops, ACM TOMS 1990

BLAS-1, BLAS-2, BLAS-3

Linear Algebra Libraries: 1990s

Solvers & eigensolvers, 1992

LAPACK

BLAS-1, BLAS-2, BLAS-3

Linear Algebra Libraries: 1990s

Distributed Memory, 1995, 1997

ScaLAPACK, PLAPACK, . . .

LAPACK

BLAS-1, BLAS-2, BLAS-3

Linear Algebra Libraries: 1990s

Dense & Sparse, 1997

PETSc, . . .

ScaLAPACK, PLAPACK, . . .

LAPACK

BLAS-1, BLAS-2, BLAS-3

Linear Algebra Libraries

and then more!

PETSc, Trilinos, . . .

ScaLAPACK, PLAPACK, Elemental, . . .

LAPACK, Plasma, SuperMatrix, Magma, . . .

BLAS-1, BLAS-2, BLAS-3, ATLAS, BTO-BLAS, BLIS, . . .

(Dense) Linear Algebra Libraries

Salient features

- Community effort. Standardized interface
- Careful organization: support routines, linear-systems, eigen-decompositions
- Clear layering: functionality, parallelism

(Dense) Linear Algebra Libraries

Salient features

- Community effort. Standardized interface
- Careful organization: support routines, linear-systems, eigen-decompositions
- Clear layering: functionality, parallelism
- Preferred outlet: ACM TOMS
- Performance & HW-driven development

(Dense) Linear Algebra Libraries

Salient features

- Community effort. Standardized interface
- Careful organization: support routines, linear-systems, eigen-decompositions
- Clear layering: functionality, parallelism
- Preferred outlet: ACM TOMS
- Performance & HW-driven development

But

- Rigid interface
- Black-box nature
- Often sub-optimal at small scale

$$K_k := P_k^b H^T (H P_k^b H^T + R)^{-1}; \quad x_k^a := x_k^b + K_k(z_k - H x_k^b); \quad P_k^a := (I - K_K H)\, P_k^b$$

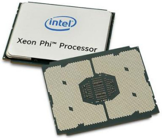
$$\begin{cases} C_\dagger := PCP^T + Q \\ K := C_\dagger H^T (H C_\dagger H^T)^{-1} \end{cases}$$

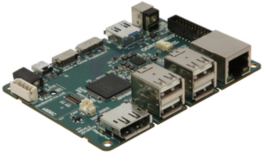
$$\Lambda := S(S^T AWAS)^{-1} S^T; \ \Theta := \Lambda AW; \ M_k := X_k A - I$$
$$X_{k+1} := X_k - M_k \Theta - (M_k \Theta)^T + \Theta^T (A X_k A - A) \Theta$$

$$x := A(B^T B + A^T R^T \Lambda R A)^{-1} B^T B A^{-1} y$$

...

$$E := Q^{-1} U (I + U^T Q^{-1} U)^{-1} U^T$$





MUL ADD MOV

MOVAPD

VFMADDPD ...

$$K_k := P_k^b H^T (H P_k^b H^T + R)^{-1}; \quad x_k^a := x_k^b + K_k(z_k - H x_k^b); \quad P_k^a := (I - K_K H)\, P_k^b$$

$$\begin{cases} C_\dagger := PCP^T + Q \\ K := C_\dagger H^T (H C_\dagger H^T)^{-1} \end{cases}$$

$$\Lambda := S(S^T AWAS)^{-1} S^T; \; \Theta := \Lambda AW; \; M_k := X_k A - I$$
$$X_{k+1} := X_k - M_k \Theta - (M_k \Theta)^T + \Theta^T (A X_k A - A) \Theta$$

$$x := A(B^T B + A^T R^T \Lambda R A)^{-1} B^T B A^{-1} y \qquad \ldots \qquad E := Q^{-1} U (I + U^T Q^{-1} U)^{-1} U^T$$

$y := \alpha x + y$ | $LU = A$ | $\cdots$ | $C := \alpha AB + \beta C$

$X := A^{-1} B$ | $C := AB^T + BA^T + C$ | $X := L^{-1} M L^{-T}$ | $QR = A$

... BLAS LAPACK ...

MUL | ADD | MOV

MOVAPD

VFMADDPD ...

$$K_k := P_k^b H^T (H P_k^b H^T + R)^{-1}; \quad x_k^a := x_k^b + K_k(z_k - H x_k^b); \quad P_k^a := (I - K_K H)\, P_k^b$$

$$\begin{cases} C_\dagger := PCP^T + Q \\ K := C_\dagger H^T (H C_\dagger H^T)^{-1} \end{cases}$$

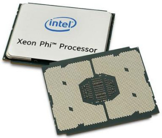
$$\Lambda := S(S^T AWAS)^{-1} S^T; \; \Theta := \Lambda AW; \; M_k := X_k A - I$$
$$X_{k+1} := X_k - M_k \Theta - (M_k \Theta)^T + \Theta^T (A X_k A - A) \Theta$$

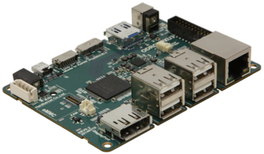
$$x := A(B^T B + A^T R^T \Lambda R A)^{-1} B^T B A^{-1} y \quad \ldots \quad E := Q^{-1} U (I + U^T Q^{-1} U)^{-1} U^T$$

$y := \alpha x + y$ | $LU = A$ | $\cdots$ | $C := \alpha AB + \beta C$

$X := A^{-1} B$ | $C := AB^T + BA^T + C$ | $X := L^{-1} M L^{-T}$ | $QR = A$

... **BLAS** **LAPACK** ...





MUL | ADD | MOV

MOVAPD

VFMADDPD ...

$$K_k := P_k^b H^T (H P_k^b H^T + R)^{-1}; \quad x_k^a := x_k^b + K_k(z_k - H x_k^b); \quad P_k^a := (I - K_K H) P_k^b$$

$$\begin{cases} C_\dagger := PCP^T + Q \\ K := C_\dagger H^T (H C_\dagger H^T)^{-1} \end{cases}$$

$$\Lambda := S(S^T AWAS)^{-1} S^T; \; \Theta := \Lambda A W; \; M_k := X_k A - I$$

$$X_{k+1} := X_k - M_k \Theta - (M_k \Theta)^T + \Theta^T (A X_k A - A) \Theta$$

$$x := A(B^T B + A^T R^T \Lambda R A \ldots$$

$$\ldots^{-1} U (I + U^T Q^{-1} U)^{-1} U^T$$

LINEAR ALGEBRA MAPPING PROBLEM (“LAMP”)

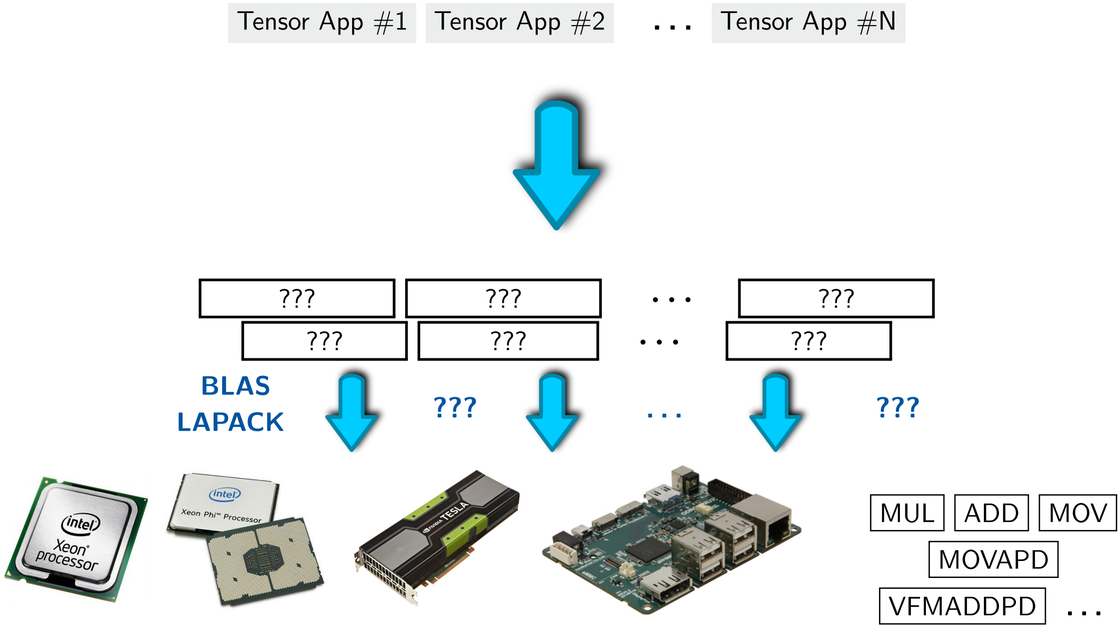
$y := \alpha x + y$	$LU = A$	$\cdots$	$C := \alpha AB + \beta C$
$X := A^{-1} B$	$C := AB^T + BA^T + C$	$X := L^{-1} M L^{-T}$	$QR = A$

C. Psarras, H. Barthels, P. Bientinesi, [arXiv:1911.09421]
“The Linear Algebra Mapping Problem. Current state of linear algebra languages and libraries”.

A. Sankaran, N.A. Alashti, C. Psarras, P. Bientinesi, [arXiv:2202.09888]
“Benchmarking the Linear Algebra Awareness of TensorFlow and PyTorch”.

H. Barthels, C. Psarras, P. Bientinesi, [arXiv:1912.12924]
“Linnea: Automatic Generation of Efficient Linear Algebra Programs”, ACM TOMS, 2021.

Tensors


Tensor App #1
Tensor App #2
...
Tensor App #N
???
???
...
???
???
???
...
???
BLAS
LAPACK
???
...
???
intel
Xeon processor
intel
Xeon Phi™ Processor
TESLA
MUL
ADD
MOV
MOVAPD
VFMADDPD
...

but. . .

Comp. Physics

Comp. Chemistry

Data Science

Machine Learning

??? … ???

??? ???

??? … ???

??? ???

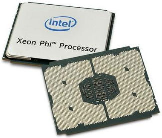
BLAS

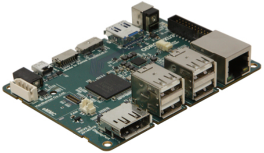
LAPACK

???

. . .

???





MUL ADD MOV

MOVAPD

VFMADDPD . . .

Tensor computations

- (At least) Two separate worlds

[1] With notable differences.

Tensor computations

- (At least) Two separate worlds
  - Computational physics / chemistry

    Tensor = Multi-linear operator

    Contractions = Generalization of matrix-matrix product

[1] With notable differences.

Tensor computations

- (At least) Two separate worlds
  - Computational physics / chemistry

    Tensor = Multi-linear operator

    Contractions = Generalization of matrix-matrix product
  - Data science

    Tensor = Collection of data

    Decompositions = Generalization of matrix factorizations[1]

[1] With notable differences.

Tensor computations

- (At least) Two separate worlds
  - Computational physics / chemistry

    Tensor = Multi-linear operator

    Contractions = Generalization of matrix-matrix product
  - Data science

    Tensor = Collection of data

    Decompositions = Generalization of matrix factorizations[1]
- Terminology and notation vary (and conflict) even within one world
- Very few software efforts cut across the boundary

[1] With notable differences.

Tensors, presently

- No “Tensor BLAS” – no collections of building blocks
- Lack of reference implementations
- No agreement on interface(s)

Tensors, presently

- No "Tensor BLAS" – no collections of building blocks
- Lack of reference implementations
- No agreement on interface(s)
- No community effort!
  A jungle of independent libraries and packages, in a variety of languages
  Massive redundancy: replication of effort, low performance

Tensors, presently

- No “Tensor BLAS” – no collections of building blocks
- Lack of reference implementations
- No agreement on interface(s)
- No community effort!
  A jungle of independent libraries and packages, in a variety of languages
  Massive redundancy: replication of effort, low performance
- Application-driven development:
  Publications scattered among diffents fields
  Re-invention of the wheel

Representative operations – building blocks candidates

Data layout operations

- Reshape
- Permute / transpose
- Sort (sparse)
- Convert data layout
- Partition
- Distribute
- ...

Arithmetic operations

- Add, subtract, scale
- Inner product
- Norms
- Element-wise operations
- Tensor-times-vector (TTV)
- Tensor-times-matrix (TTM)
- MTTKRP
- Contractions
- ...

Decompositions

- CP (CANDECOMP/PARAFAC)
- Tucker
- INDSCAL
- PARAFAC2
- CANDELINC
- DEDICOM
- PARATUCK2
- ...

Representative operations – building blocks candidates

Data layout operations

- Reshape
- Permute / transpose
- Sort (sparse)
- Convert data layout
- Partition
- Distribute
- ...

Arithmetic operations

- Add, subtract, scale
- Inner product
- Norms
- Element-wise operations
- Tensor-times-vector (TTV)
- Tensor-times-matrix (TTM)
- MTTKRP
- Contractions
- ...

Decompositions

- CP (CANDECOMP/PARAFAC)
- Tucker
- INDSCAL
- PARAFAC2
- CANDELINC
- DEDICOM
- PARATUCK2
- ...

Where to draw the boundaries?
E.g., where does the "T-BLAS" end and the "T-LAPACK" begin?

2-level "solution"

Chromatography-MS

CP/PARAFAC | Tucker | PARAFAC2

Transposition | Contraction | ... | Alternating LS

Khatri-Rao | SpMTTKRP | ... | TTV, TTM

HPTT **TCL** **...** **BLAS**

MUL | ADD | MOV

MOVAPD

VFMADDPD ...

Algorithms for CP (PARAFAC) decomposition

Hence all the different libraries

- **Algebraic algorithms**
  - Generalized Rank Annihilation Method
  - Direct TriLinear Decomposition
  - The "algebraic algorithm" by Domanov and De Lathauwer
  - The "simpler algorithm" by Pimentel-Alarcón
  - ...
- **Alternating optimization algorithms**
  - Alternating Least Squares
  - Fast ALS
  - Hierarchical ALS
  - Regularized ALS
  - ...
- **All-at-once optimization algorithms**
  - Gradient descent
  - (Damped) Gauss–Newton
  - Nonlinear CG, GMRES
  - Quasi-Netwon (e.g., L-BFGS)
  - ...
- **Enhancements**
  - Line search
  - Compression
  - Randomization
  - Transient constraints
  - ...

Algorithms for CP (PARAFAC) decomposition

Hence all the different libraries

- **Algebraic algorithms**
  - Generalized Rank Annihilation Method
  - Direct TriLinear Decomposition
  - The “algebraic algorithm” by Domanov and De Lathauwer
  - The “simpler algorithm” by Pimentel-Alarcón
  - ...
- **Alternating optimization algorithms**
  - Alternating Least Squares
  - Fast ALS
  - Hierarchical ALS
  - Regularized ALS
  - ...
- **All-at-once optimization algorithms**
  - Gradient descent
  - (Damped) Gauss–Newton
  - Nonlinear CG, GMRES
  - Quasi-Netwon (e.g., L-BFGS)
  - ...
- **Enhancements**
  - Line search
  - Compression
  - Randomization
  - Transient constraints
  - ...

Is a computational hierarchy even possible? Or is optimality achieved via specialized kernels?

Also . . .   one vs. many problem instances

Also . . . one vs. many problem instances

Coupled-Cluster methods

$$
\begin{aligned}
\tau_{ij}^{ab} &= t_{ij}^{ab} + \frac{1}{2} P_b^a P_j^i t_i^a t_j^b, \\
\tilde{F}_e^m &= f_e^m + \sum_{fn} v_{ef}^{mn} t_n^f, \\
\tilde{F}_e^a &= (1-\delta_{ae}) f_e^a - \sum_m \tilde{F}_e^m t_m^a - \frac{1}{2} \sum_{mnf} v_{ef}^{mn} t_{mn}^{af} + \sum_{fn} v_{ef}^{an} t_n^f, \\
\tilde{F}_i^m &= (1-\delta_{mi}) f_i^m + \sum_e \tilde{F}_e^m t_i^e + \frac{1}{2} \sum_{nef} v_{ef}^{mn} t_{in}^{ef} + \sum_{fn} v_{if}^{mn} t_n^f,
\end{aligned}
$$

$$
\begin{aligned}
\tilde{W}_{ei}^{mn} &= v_{ei}^{mn} + \sum_f v_{ef}^{mn} t_i^f, \\
\tilde{W}_{ij}^{mn} &= v_{ij}^{mn} + P_j^i \sum_e v_{ie}^{mn} t_j^e + \frac{1}{2} \sum_{ef} v_{ef}^{mn} \tau_{ij}^{ef}, \\
\tilde{W}_{ie}^{am} &= v_{ie}^{am} - \sum_n \tilde{W}_{ei}^{mn} t_n^a + \sum_f v_{ef}^{ma} t_i^f + \frac{1}{2} \sum_{nf} v_{ef}^{mn} t_{in}^{af}, \\
\tilde{W}_{ij}^{am} &= v_{ij}^{am} + P_j^i \sum_e v_{ie}^{am} t_j^e + \frac{1}{2} \sum_{ef} v_{ef}^{am} \tau_{ij}^{ef}, \\
z_i^a &= f_i^a - \sum_m \tilde{F}_i^m t_m^a + \sum_e f_e^a t_i^e + \sum_{em} v_{ei}^{ma} t_m^e + \sum_{em} v_{im}^{ae} \tilde{F}_e^m + \frac{1}{2} \sum_{efm} \cdots \\
z_{ij}^{ab} &= v_{ij}^{ab} + P_j^i \sum_e v_{ie}^{ab} t_j^e + P_b^a P_j^i \sum_{me} \tilde{W}_{ie}^{am} t_{mj}^{eb} - P_b^a \sum_m \tilde{W}_{ij}^{am} t_m^b + P \cdots
\end{aligned}
$$

credits to D. Matthews, E. Solomonik, J. Stanton, and J. Gauss

Also . . . one vs. many problem instances

Coupled-Cluster methods

$$\begin{aligned}
\tau_{ij}^{ab} &= t_{ij}^{ab} + \frac{1}{2} P_b^a P_j^i t_i^a t_j^b, \\
\tilde{F}_e^m &= f_e^m + \sum_{fn} v_{ef}^{mn} t_n^f, \\
\tilde{F}_e^a &= (1-\delta_{ae}) f_e^a - \sum_m \tilde{F}_e^m t_m^a - \frac{1}{2} \sum_{mnf} v_{ef}^{mn} t_{mn}^{af} + \sum_{fn} v_{ef}^{an} t_n^f, \\
\tilde{F}_i^m &= (1-\delta_{mi}) f_i^m + \sum_e \tilde{F}_e^m t_i^e + \frac{1}{2} \sum_{nef} v_{ef}^{mn} t_{in}^{ef} + \sum_{fn} v_{if}^{mn} t_n^f,
\end{aligned}$$

$$\begin{aligned}
\tilde{W}_{ei}^{mn} &= v_{ei}^{mn} + \sum_f v_{ef}^{mn} t_i^f, \\
\tilde{W}_{ij}^{mn} &= v_{ij}^{mn} + P_j^i \sum_e v_{ie}^{mn} t_j^e + \frac{1}{2} \sum_{ef} v_{ef}^{mn} \tau_{ij}^{ef}, \\
\tilde{W}_{ie}^{am} &= v_{ie}^{am} - \sum_n \tilde{W}_{ei}^{mn} t_n^a + \sum_f v_{ef}^{ma} t_i^f + \frac{1}{2} \sum_{nf} v_{ef}^{mn} t_{in}^{af}, \\
\tilde{W}_{ij}^{am} &= v_{ij}^{am} + P_j^i \sum_e v_{ie}^{am} t_j^e + \frac{1}{2} \sum_{ef} v_{ef}^{am} \tau_{ij}^{ef}, \\
z_i^a &= f_i^a - \sum_m \tilde{F}_i^m t_m^a + \sum_e f_e^a t_i^e + \sum_{em} v_{ei}^{ma} t_m^e + \sum_{em} v_{im}^{ae} \tilde{F}_e^m + \frac{1}{2} \sum_{efm} \\
z_{ij}^{ab} &= v_{ij}^{ab} + P_j^i \sum_e v_{ie}^{ab} t_j^e + P_b^a P_j^i \sum_{me} \tilde{W}_{ie}^{am} t_{mj}^{eb} - P_b^a \sum_m \tilde{W}_{ij}^{am} t_m^b + P
\end{aligned}$$

credits to D. Matthews, E. Solomonik, J. Stanton, and J. Gauss

Finite Element 3D diffusion operator

```
TE.BeginMultiKernelLaunch();
TE("T2_e_i1_i2_k3 = B_k3_i3 X_e_i1_i2_i3", T2, B, X);
TE("T1_e_i1_k2_k3 = B_k2_i2 T2_e_i1_i2_k3", T1, B, T2);
TE("U1_e_k1_k2_k3 = G_k1_i1 T1_e_i1_k2_k3", U1, G, T1);
TE("T1_e_i1_k2_k3 = G_k2_i2 T2_e_i1_i2_k3", T1, G, T2);
TE("U2_e_k1_k2_k3 = B_k1_i1 T1_e_i1_k2_k3", U2, B, T1);
TE("T2_e_i1_i2_k3 = G_k3_i3 X_e_i1_i2_i3", T2, G, X);
TE("T1_e_i1_k2_k3 = B_k2_i2 T2_e_i1_i2_k3", T1, B, T2);
TE("U3_e_k1_k2_k3 = B_k1_i1 T1_e_i1_k2_k3", U3, B, T1);
TE("Z_m_e_k1_k2_k3 = U_n_e_k1_k2_k3 D_e_m_n_k1_k2_k3", Z, U,
TE("T1_e_i3_k1_k2 = B_k3_i3 Z1_e_k1_k2_k3", T1, B, Z1);
TE("T2_e_i2_i3_k1 = B_k2_i2 T1_e_i3_k1_k2", T2, B, T1);
TE("Y_e_i1_i2_i3 = G_k1_i1 T2_e_i2_i3_k1", Y, G, T2);
TE("T1_e_i3_k1_k2 = B_k3_i3 Z2_e_k1_k2_k3", T1, B, Z2);
TE("T2_e_i2_i3_k1 = G_k2_i2 T1_e_i3_k1_k2", T2, G, T1);
TE("Y_e_i1_i2_i3 += B_k1_i1 T2_e_i2_i3_k1", Y, B, T2);
TE("T1_e_i3_k1_k2 = G_k3_i3 Z3_e_k1_k2_k3", T1, G, Z3);
TE("T2_e_i2_i3_k1 = B_k2_i2 T1_e_i3_k1_k2", T2, B, T1);
TE("Y_e_i1_i2_i3 += B_k1_i1 T2_e_i2_i3_k1", Y, B, T2);
TE.EndMultiKernelLaunch();
```

credits to A. Fisher – https://github.com/LLNL/acrotensor

Also . . . one vs. many problem instances

- BLAS and LAPACK built and optimized for single problems
  Only much much later "streaming BLAS", "batched BLAS"

Also . . . one vs. many problem instances

- BLAS and LAPACK built and optimized for single problems
  Only much much later "streaming BLAS", "batched BLAS"

- In Tensorland: multiple contractions, multiple indices contracted, tensor networks, multiple decompositions in one workflow, . . .

  Huge opportunities for multi-instance / multi-problem optimizations.

Also . . . one vs. many problem instances

- BLAS and LAPACK built and optimized for single problems
  Only much much later "streaming BLAS", "batched BLAS"

- In Tensorland: multiple contractions, multiple indices contracted, tensor networks, multiple decompositions in one workflow, . . .

  Huge opportunities for multi-instance / multi-problem optimizations.

Example:

C. Psarras, L. Karsson, R. Bro, P. Bientinesi, [arXiv:2010.04678v2]
"Concurrent Alternating Least Squares for multiple simultaneous Canonical Polyadic Decompositions", ACM TOMS.

Summary & questions

	Matrices	Tensors

Summary & questions

	Matrices	Tensors
Driver	performance, HW	applications

Summary & questions

	Matrices	**Tensors**
Driver	performance, HW	applications
Community effort	BLAST/LAPACK/...	group by group

Summary & questions

	Matrices	**Tensors**
Driver	performance, HW	applications
Community effort	BLAST/LAPACK/...	group by group
Industry	wide support	some support

Summary & questions

	Matrices	**Tensors**
Driver	performance, HW	applications
Community effort	BLAST/LAPACK/...	group by group
Industry	wide support	some support
Standardization	interface, layers, ...	"pointless" – CvL

Summary & questions

	Matrices	Tensors
Driver	performance, HW	applications
Community effort	BLAST/LAPACK/...	group by group
Industry	wide support	some support
Standardization	interface, layers, . . .	“pointless” – CvL
Preferred outlet	ACM TOMS	—

Summary & questions

	Matrices	Tensors
Driver	performance, HW	applications
Community effort	BLAST/LAPACK/...	group by group
Industry	wide support	some support
Standardization	interface, layers, . . .	“pointless” – CvL
Preferred outlet	ACM TOMS	—
Language native support	plenty; BLAS	so

Summary & questions

	Matrices	**Tensors**
Driver	performance, HW	applications
Community effort	BLAST/LAPACK/...	group by group
Industry	wide support	some support
Standardization	interface, layers, . . .	“pointless” – CvL
Preferred outlet	ACM TOMS	—
Language native support	plenty; BLAS	so

- “DEV people”: What does it take to kickstart a community effort? . . . Dagstuhl?

Summary & questions

	Matrices	Tensors
Driver	performance, HW	applications
Community effort	BLAST/LAPACK/...	group by group
Industry	wide support	some support
Standardization	interface, layers, . . .	"pointless" – CvL
Preferred outlet	ACM TOMS	—
Language native support	plenty; BLAS	so

- "DEV people": What does it take to kickstart a community effort? . . . Dagstuhl?
- "APP people": What would it take for you to consider using different libraries, possibly a different language?