

# Efficient Computation Through Tuned Approximation



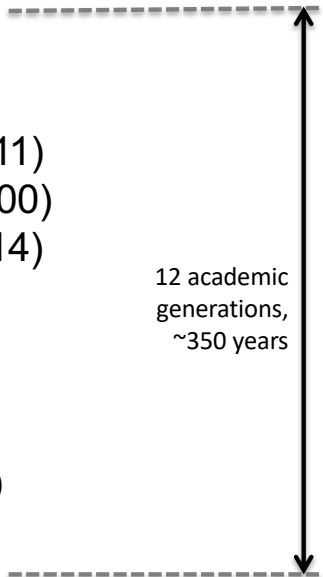
جامعة الملك عبد الله  
للعلوم والتقنية  
King Abdullah University of  
Science and Technology



David Keyes  
and the HiCMA group of KAUST's  
Extreme Computing Research Center

# Jim's descent from the Bernoullis

Jakob Bernoulli (Basel, 1676)  
Johann Bernoulli (Basel, 1694)  
Leonard Euler (Basel, 1726)  
Joseph-Louis Lagrange (Torino, 1811)  
Siméon Poisson (Polytechnique, 1800)  
Michel Chasles (Polytechnique, 1814)  
A. H. Newton (Yale, 1850)  
E. H. Moore (Yale, 1885)  
George Birkoff (Chicago, 1907)  
Marshall Stone (Harvard, 1926)  
Richard Kadison (Chicago, 1950)  
Jim Glimm (Columbia, 1959)  
many of YOU



12 academic  
generations,  
~350 years

# Our common blessing from Jim

## *THEOREM 1*

We are all young.

## *PROOF*

We are all younger than Jim.

Jim is young.

Transitivity.

## BS Eng (1956), PhD Math (1959) at Columbia

“I have always been interested in mathematics, physics and engineering. While I was a student, I felt it very difficult to choose among these interests. As it turns out, I have managed to do all three, mostly by choice of projects and goals that rotated among these choices. The mixture of these topics especially **when combined with computing** is very powerful, as it allows one to address a very wide range of problems.”

# My intersection with Jim

## Linear Augmented Slater-Type Orbital Method for Free Standing Clusters

K. S. KANG,<sup>1</sup> J. W. DAVENPORT,<sup>1,2</sup> J. GLIMM,<sup>1,3</sup> D. E. KEYES,<sup>4</sup> AND M. MCGUIGAN<sup>1</sup>

<sup>1</sup>Computational Science Center, Brookhaven National Laboratory, Upton, New York

<sup>2</sup>Center for Functional Nanomaterials, Brookhaven National Laboratory, Upton, New York

<sup>3</sup>Department of Applied Mathematics and Statistics, Stony Brook University, Stony Brook, New York

<sup>4</sup>Department of Applied Physics and Applied Mathematics, Columbia University,  
New York, New York

Received 10 June 2008; Accepted 2 September 2008

DOI 10.1002/jcc.21138

Published online in Wiley InterScience (www.interscience.wiley.com).

---

**Abstract:** We have developed a Scalable Linear Augmented Slater-Type Orbital (LASTO) method for electronic-structure calculations on free-standing atomic clusters. As with other linear methods we solve the Schrödinger equation using a mixed basis set consisting of numerical functions inside atom-centered spheres and matched onto tail functions outside. The tail functions are Slater-type orbitals, which are localized, exponentially decaying functions. To solve the Poisson equation between spheres, we use a finite difference method replacing the rapidly varying charge density inside the spheres with a smoothed density with the same multipole moments. We use multigrid techniques on the mesh, which yields the Coulomb potential on the spheres and in turn defines the potential inside via a Dirichlet problem. To solve the linear eigen-problem, we use ScaLAPACK, a well-developed package to solve large eigensystems with dense matrices. We have tested the method on small clusters of palladium.

# Jim and SciDAC

Jim led the front tracking pillar of **TSTT** (Terascale Simulation Tools and Technology)

Phil led **APDEC** (Applied Partial Differential Equations Center)

Yours truly led **TOPS** (Terascale Optimal PDE Solvers)



**SciDAC**  
Scientific Discovery  
through  
Advanced Computing

# Speakers at Jim's 70th at Stonybrook, 3-5 Aug 2004

Chi-Wang

Constantine

Gretar

Gui-Qiang

Phil

Snezhana

*Yours truly*

Out of 36 presenters overall  
spanning three days

# Efficient Computation Through Tuned Approximation



جامعة الملك عبد الله  
للعلوم والتقنية  
King Abdullah University of  
Science and Technology



David Keyes  
and the HiCMA group of KAUST's  
Extreme Computing Research Center



## Alternative titles

“How to get four Gordon Bell Prize Finalist nominations out of *one simple idea*”

“Do linear algebra;  
see the world!”



Earl Blossom, 1891-1970

# Conclusions, up front

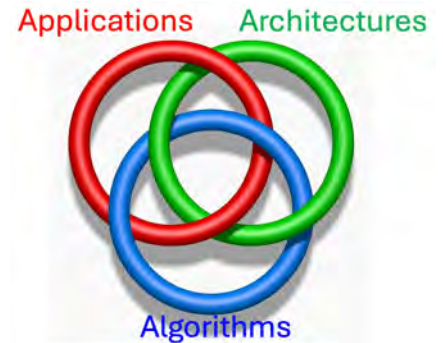
As computational infrastructure demands a growing sector of research budgets and global energy expenditure, we must address the need for greater efficiency

As a community, we have excelled at this historically in three aspects:

- architectures
- applications (redefining *actual outputs of interest*)
- algorithms

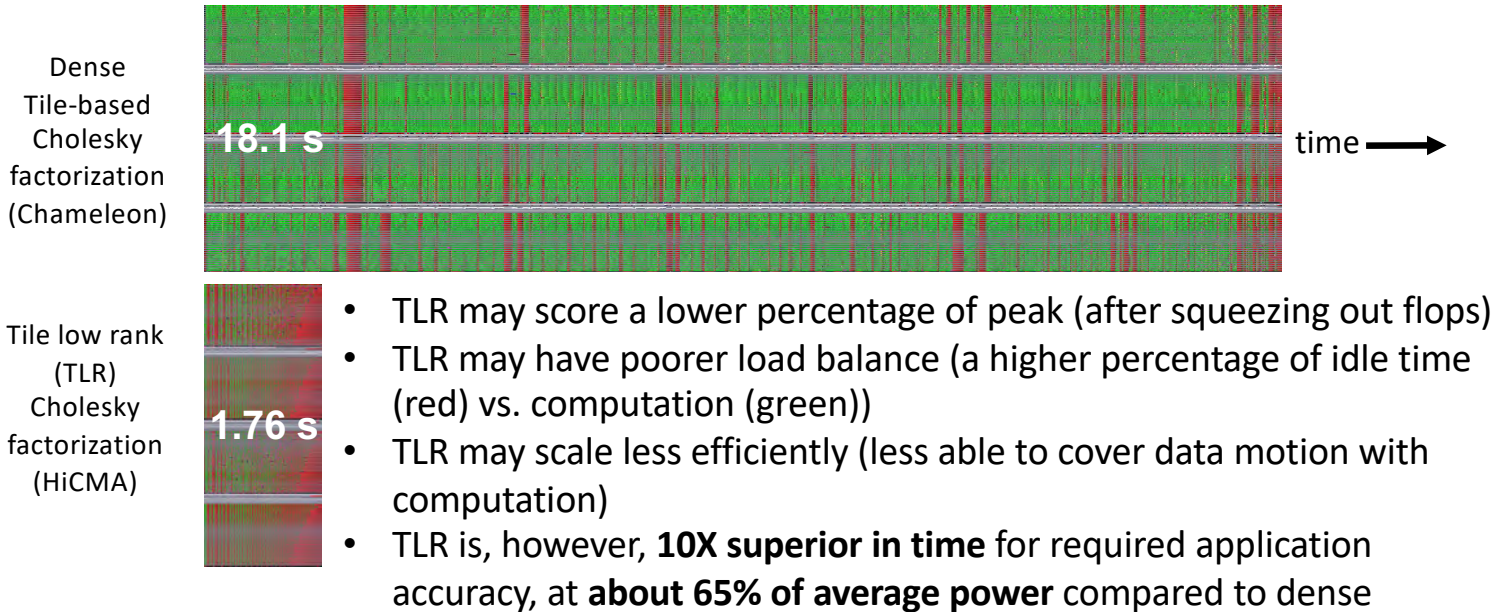
There are *new algorithmic* opportunities in:

- reduced rank representations
- reduced precision representations

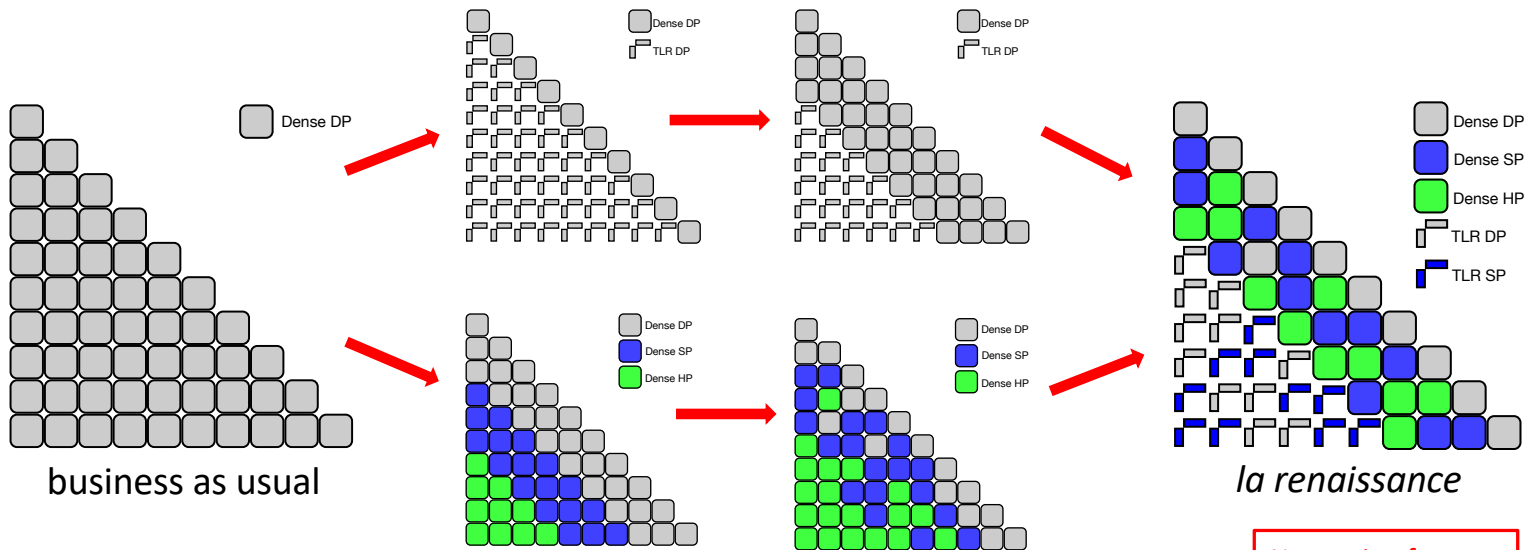


# Our journey in tuned approximation began in 2018 with these time traces for tile low-rank (TLR) Cholesky

... for factorization of a dense 54K covariance matrix on four 32-core nodes of a Cray XC-40



# Computational efficiency through *tuned approximation*: a journey with *tile low rank* and *mixed precision*

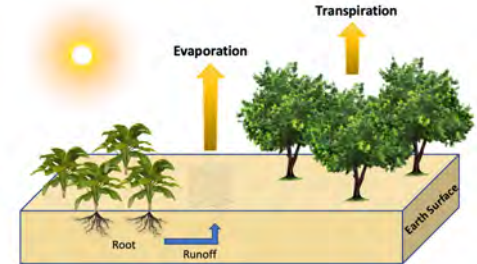


Don't oversolve: maintain just enough accuracy for the application purpose  
Economize on storage: no extra copies of the original matrix

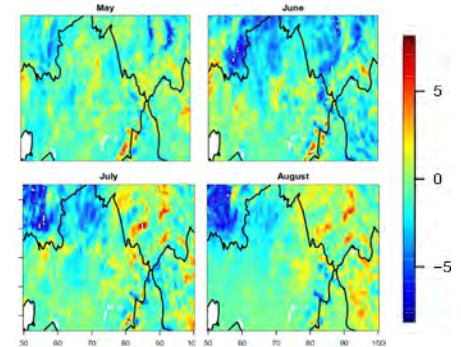
Now using four  
precisions: FP64,  
FP32, FP16 & FP8

# Efficiency (“science per Joule”) improvement in HPC?

- We consider 3 categories of efficiency improvement
  - from architectures, applications, algorithms
- In 2022, 2023, and a pair of 2024 Gordon Bell finalist papers
- Through efficiency improvements in inner linear algebra operations from exploiting
  - rank structure (related to correlation smoothness)
  - precision structure (related to correlation magnitudes)



time series evapotranspiration



# Time-to-solution addresses the energy “elephant”



Frontier (#1 on Top500) delivers about 1 Exaflop/s at about 50 Gigaflop/s per Watt

- **20 MegaWatts consumed continuously**

Representative electricity cost in US is \$ 0.20 per KiloWatt-hour

- **\$ 200 per MegaWatt-hour**

Powering an exaflop/s system costs about \$ 4,000 per hour

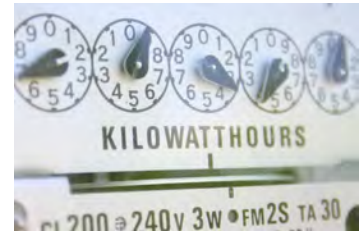
- 10 Kilohour per year (8,760, to be more precise)

→ **\$40 million annual electricity bill for an exaflop/s system**

Carbon footprint of a KiloWatt-hour is about 0.5 kg CO<sub>2</sub>-equivalent

- 10,000 kg CO<sub>2</sub>e hourly carbon footprint for an exaflop/s system
- 100,000 metric tons CO<sub>2</sub>e annually

→ **equivalent to 20,000 typical passenger cars in the USA**



A 10% improvement:  
saves \$4M/year  
takes 2,000 cars off the road

A 10X improvement:  
saves \$36M/year  
takes 18,000 cars off the road

10X is actually  
achievable in many  
use cases

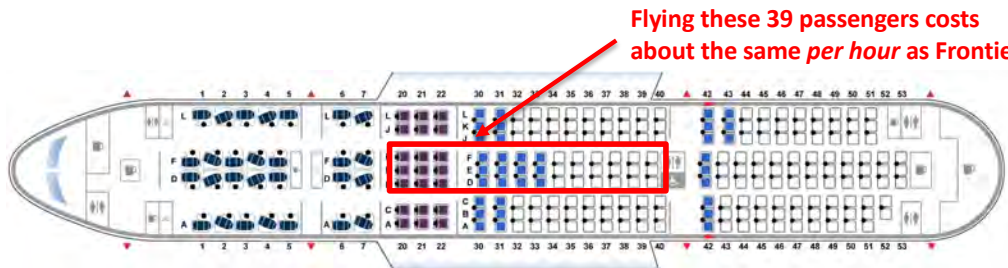
# CO2 production equivalents



“Science per Joule”  
is a matter of  
planetary  
stewardship

# Running on Frontier versus flying commercially

- Carbon footprint of a KiloWatt-hour is 0.5 kg CO<sub>2</sub>-equivalent
  - 10,000 kg CO<sub>2</sub>e hourly carbon footprint for an exaflop/s system (10 metric tons)
- Carbon footprint of one passenger-hour of commercial cruise Mach flight is about 0.25 metric tons CO<sub>2</sub>e
  - 1 hour of exaflop/s is roughly equivalent to 40 passenger-hours of flight

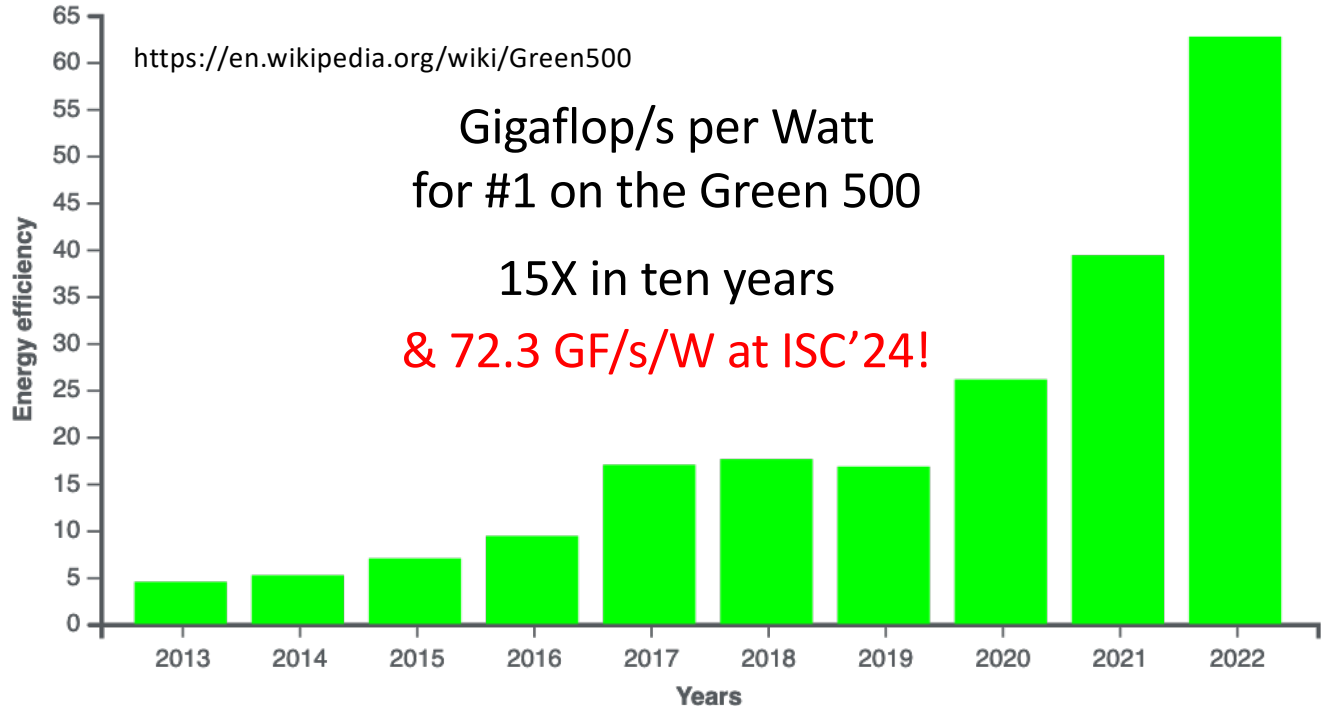


Carbon offset your next flight by efficient programming!

Better yet, please justify my flight here to give this talk 😊



# Architecture efficiency tracked by the Green 500

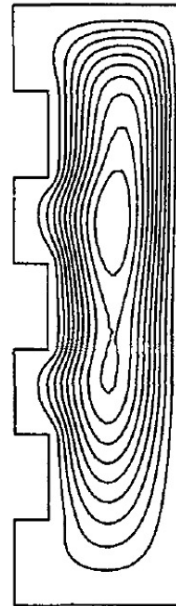


# Application “efficiency” from redefining the objective

Sometimes, the output of interest from a computation is not a solution to high accuracy everywhere, but a *functional* of the solution to a *specified accuracy*, e.g.

- compute the convective heat flux across a fluid-solid boundary, obtainable without globally uniform accuracy
- use low fidelity surrogates in early inner iterations of “outer loop problems”

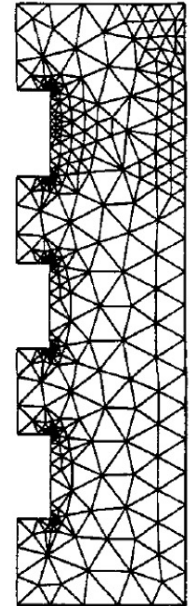
Machiels, Peraire & Patera, *A posteriori FE Output Bounds for the Incompressible NS Equations*, (2001), J. Comp. Phys. **172**:401



temperature  
contour



conservative  
mesh

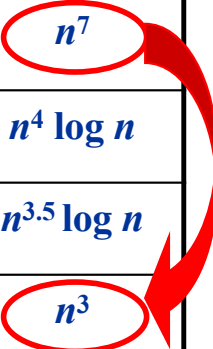


output bound  
mesh (flux to 1%)

# HPC algorithmic efficiency tracked by Poisson solvers

Consider a Poisson solve in a 3D  $n \times n \times n$  box; natural ordering gives bandwidth of  $n^2$

<i>Year</i>	<i>Method</i>	<i>Reference</i>	<i>Storage</i>	<i>Flops</i>
1947	GE (banded)	Von Neumann & Goldstine	$n^5$	$n^7$
1950	Optimal SOR	Young	$n^3$	$n^4 \log n$
1971/77	MILU-CG	Reid/Van der Vorst	$n^3$	$n^{3.5} \log n$
1984	Full MG	Brandt	$n^3$	$n^3$

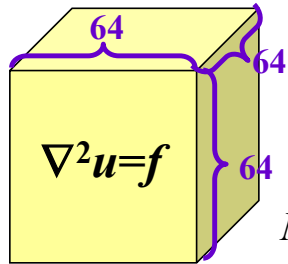


If  $n = 64$ , this implies an overall reduction in flops of  $\sim 16$  million \*

\*Six months is reduced to 1 second (recall:  $3.154 \times 10^7$  seconds per year)

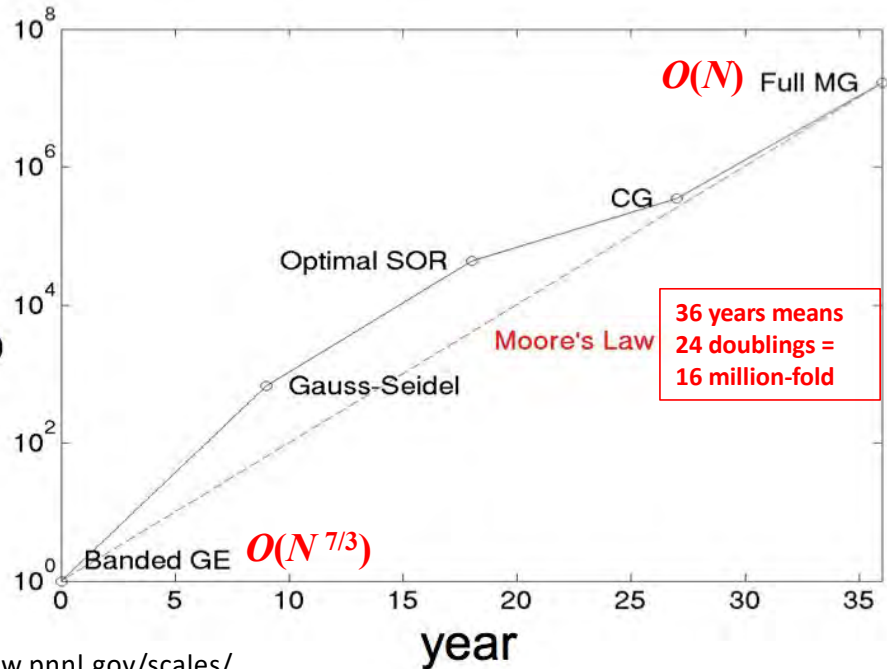
# “Algorithmic Moore’s Law”

HPC progresses even faster in algorithms than in hardware:  
*example of Poisson’s equation in a 3D box with 2nd-order FD*



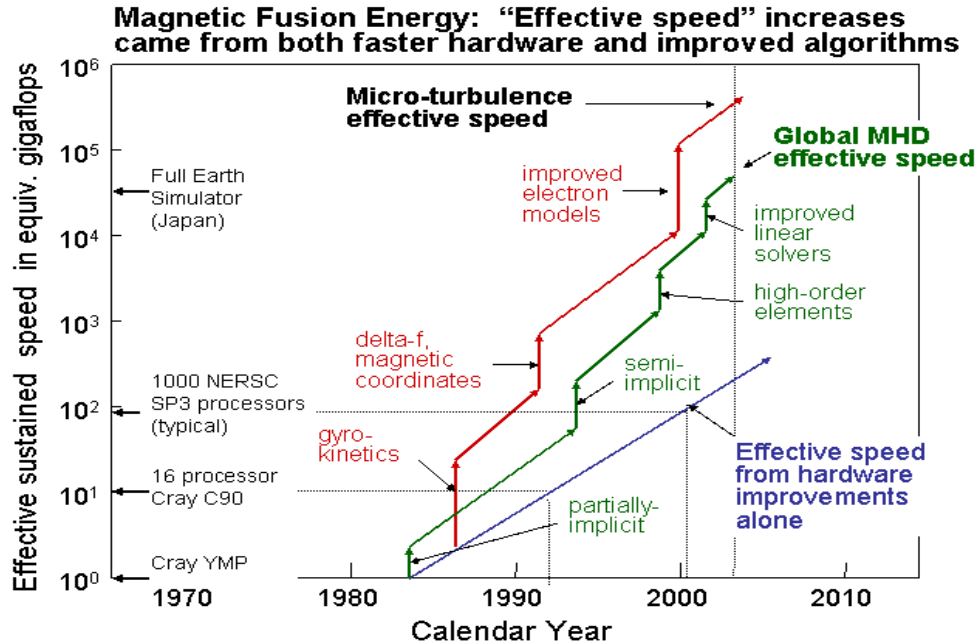
$$N = n^3 = (1/h)^3$$

relative speedup



# “Algorithmic Moore’s Law” for fusion energy simulations

GKT in  
red  
MHD in  
green  
Moore’s  
Law in  
blue



“Semi-implicit”:

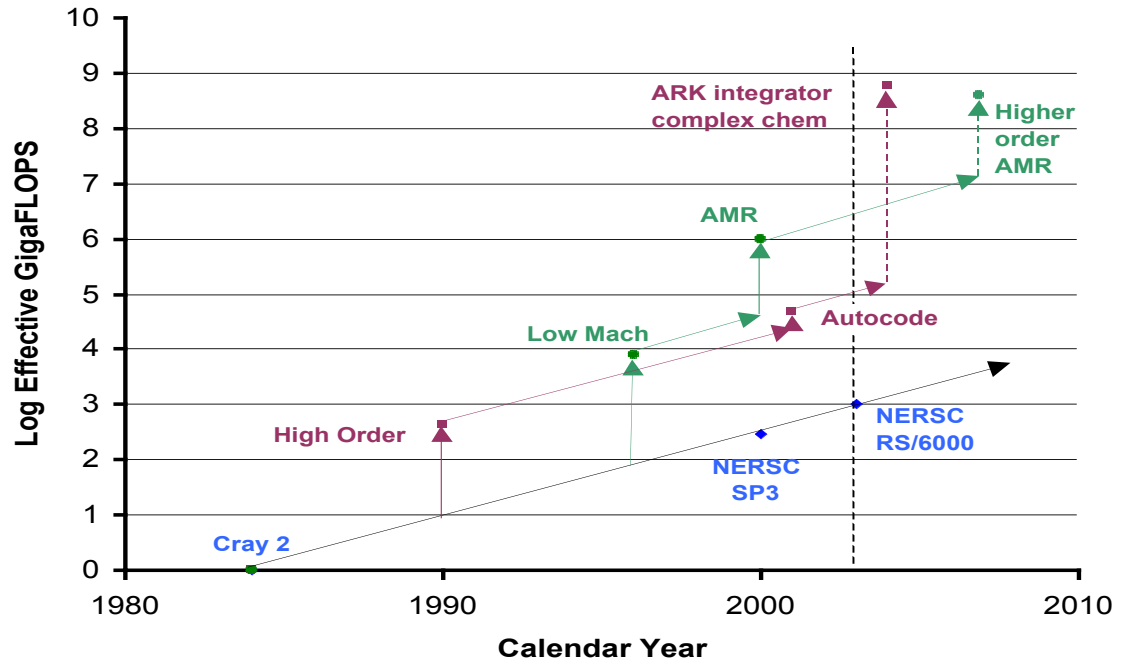
All waves treated implicitly, but still stability-limited by transport

“Partially implicit”:

Fastest waves filtered, but still stability-limited by slower waves

# “Algorithmic Moore’s Law” for combustion simulations

Complex kinetics in maroon  
CFD in green  
Moore’s Law in blue



# Algorithms improve exponents; Moore only adjusts the base

- To scale to extremes, one must start with algorithms with optimal asymptotic complexity,  $O(N \log^p N)$ ,  $p = 0, 1, 2$
- These are typically (not exclusively) recursively hierarchical
- Some such algorithms through the decades:
  - Fast Fourier Transform (1960's):  $N^2 \rightarrow N \log N$
  - Multigrid (1970's):  $N^{4/3} \log N \rightarrow N$
  - Fast Multipole (1980's):  $N^2 \rightarrow N$
  - Sparse Grids (1990's):  $N^d \rightarrow N (\log N)^{d-1}$
  - $\mathcal{H}$  matrices (2000's):  $N^3 \rightarrow k^2 N (\log N)^2$
  - Multilevel Monte Carlo (2000's):  $N^{3/2} \rightarrow N (\log N)^2$
  - Randomized matrix algorithms (2010's):  $N^3 \rightarrow N^2 \log k$
  - ??? (2020's):  $??? \rightarrow ???$  (*your challenge!*)

# Hints for contributions for the 2020's



*You* are going to replace inefficient first-order convergent neural network training methods by, e.g.,

- communication-reducing hierarchically preconditioned second-order methods
- matrix-free nonlinear acceleration methods (e.g., Anderson)



*You* are going to replace inefficient ML inference by, e.g.,

- by pruned, compressed forms of “attention”

“With great computational power comes great algorithmic responsibility.”

– Longfei Gao, ALCF (PhD 2013, KAUST)



# Hints for contributions for the 2020's



*You* are going to attach QPUs to classical supercomputers to farm out tasks that offer exponential speedup over their classical counterparts ( $O(N) \rightarrow O(\log(N))$ )

- when offered sufficiently many sufficiently reliable qubits to confer quantum advantage



*You* are going master hybridized mod-sim/ML/QC workflows

- using few instances of high fidelity, high resolution simulations supplemented by many instances of machine-learned surrogates
- each modality being employed where it is the most energy efficient

“With great computational power comes great algorithmic responsibility.”

– Longfei Gao, ALCF (PhD 2013, KAUST)

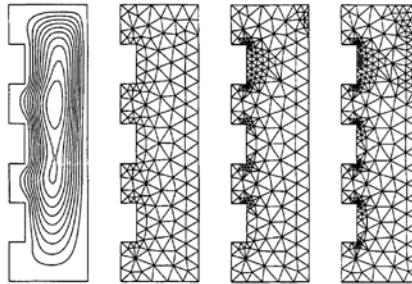
# Science per Joule – summary so far

Improving the “science per Joule” (or per unit time) involves:

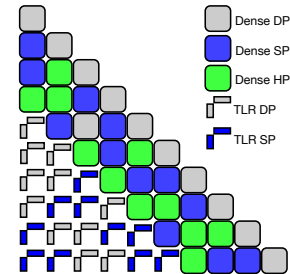
architecture



application



algorithm/software



In a fortunate world, these are orthogonal: the *desired app* can employ the *best algorithm* on the most *efficient hardware*.

# Algorithmic “secret sauce”

Where possible, without losing accuracy:

- Replace default double precision (64-bit IEEE standard) with lower precisions
  - Save storage
  - Save data motion
  - Exploit special-purpose hardware optimized for low precision
- Replace default full rank blocks of discrete linear operators or discrete field data with lower rank blocks
  - Save storage
  - Save data motion
  - Exploit special-purpose hardware optimized for BLAS3

## Two natural questions

- How did double precision become the default for scientific computing in the first place?
  - When should we be cautious of low precision?
- What is the intuition behind low-rank approximation?
  - When should we expect it to be practical?

# Lessons from the 1D Laplacian

Two concepts we need to understand in our pursuit of computational efficiency in linear algebra:

- conditioning (implications on precision)
- rank structure (implications on sparsification)

can be motivated with reference to the 1D Laplacian (to be precise, its negative  $-\Delta$ ), discretized here to second-order in FD, FE, or FV:

$$\begin{bmatrix} 2 & -1 & & & & & \\ -1 & 2 & -1 & & & & \\ & -1 & 2 & -1 & & & \\ & & -1 & 2 & -1 & & \\ & & & -1 & 2 & -1 & \\ & & & & -1 & 2 & -1 \\ & & & & & -1 & 2 \end{bmatrix}$$

# Laplacian ill-conditioned stresses precision

Let  $n = 1/h$  and consider Dirichlet end conditions with  $n-1$  interior points. Then:

$$\lambda_1 = 2 [1 - \cos \pi/n] \sim (\pi/n)^2$$

$$\lambda_{n-1} = 2 [1 - \cos (n-1)\pi/n] \sim 4$$

As  $n$  gets large and the mesh resolves more Fourier components, the condition number grows like the square of the matrix dimension (inverse mesh parameter):

$$\kappa = \lambda_{n-1} / \lambda_1 \sim (4/\pi^2) n^2$$

In single precision real arithmetic,  $\kappa$  approaches the reciprocal of macheps ( $10^{-7}$ ) for an  $n$  as small as  $2^{10}$  ( $\sim 10^3$ ). Laplacian-like operators arise throughout modeling and simulation (diffusion, electrostatics, gravitation, stress, graphs, etc.), implying  $O(1)$  error in the result, so HPC has traditionally demanded double precision by default. GPUs were accepted only when they offered hardware DP (2008, NVIDIA GTX 280).

For the biharmonic, even double precision gives out at  $n = 2^{10}$ . Some multiscale codes require quadruple precision, often available only in software.

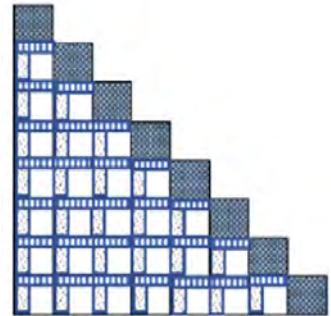


# A renaissance in numerical linear algebra (1)

It turns out that many formally dense matrices arising from

- **integral equations** with smooth Green's functions
- **covariances** in statistics
- **Schur complements** within discretizations of PDEs
- **Hessians** from PDE-constrained optimization
- **nonlocal operators** from fractional differential equations
- **radial basis functions** from unstructured meshing
- **kernel matrices** from GWAS & machine learning applications

have exploitable low-rank structure in “most” their off-diagonal blocks (if well ordered)



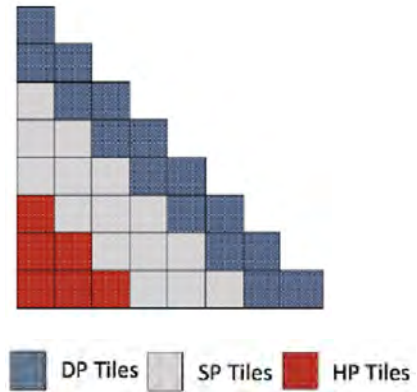


## A renaissance in numerical linear algebra (2)

It turns out that many matrices arising in applications have blocks of **relatively small norm** and can be replaced with **reduced precision**.

Mixed precision algorithms have a long history, e.g., iterative refinement (1963, Wilkinson), where multiple copies of the matrix are kept in different precisions for different purposes.

There are many such new algorithms; see Higham & Mary, *Mixed precision algorithms in numerical linear algebra*, Acta Numerica (2022).



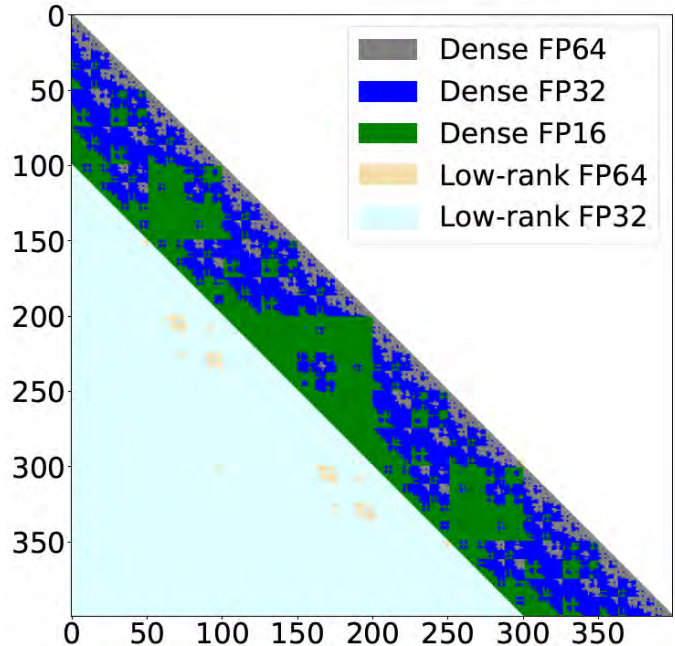
# A renaissance in numerical linear algebra (3)

Moreover, these ideas can be combined, as in this 1M x 1M dense symmetric covariance matrix:

- Original in DP: 4 TB
- Replacement: 0.915 TB

Smaller workingsets mean larger problems fit in GPUs and last-level caches on CPUs, for data movement savings

- Also, net computational savings
- Data structures and programs are more complex



# Rank: a tuning knob

- Replace dense blocks with reduced rank representations, whether “born dense” or as arising during matrix operations
  - use high accuracy (high rank) to build “exact” solvers
  - use low accuracy (low rank) to build preconditioners
- Consider hardware parameters in tuning block sizes and maximum rank parameters, to complement mathematical considerations
  - e.g., cache sizes, warp sizes
- Select from already broad and ever broadening algorithmic menu to form low-rank blocks (next slide)
  - traditionally a flop-intensive vendor-optimized GEMM-based flat algorithm
- Implement in “batches” of leaf blocks
  - flattening trees in the case of hierarchical methods

# Low-rank approximations for compressible tiles

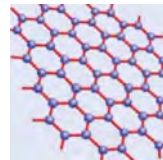
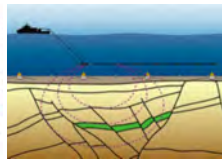
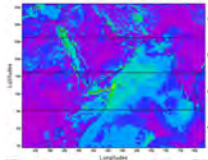
Options for forming data sparse representations of the amenable off-diagonal blocks

- standard SVD:  $O(n^3)$ , too expensive, especially for repeated compressions after additive tile manipulations
- randomized SVD (Halko *et al.*, 2011):  $O(n^2 \log k)$  for rank  $k$ , requires only a small number of passes over the data, saving over the SVD in memory accesses as well as operations
- adaptive cross approximation (ACA) (Bebendorf, 2000):  $O(k^2 n \log n)$ , motivated by integral equation kernels
- matrix skeletonization (representing a matrix by a representative collection of row and columns), such as CUR, sketching, or interpolatory decompositions based on proxies

# Application opportunities

With such new algorithms, today's HPC can extend many applications that possess

- memory capacity constraints (e.g., geospatial statistics, PDE-constrained optimization)
- power constraints (e.g., remote telescopes)
- real-time constraints (e.g., wireless communication)
- running time constraints (e.g., chemistry, materials, genome-wide associations)



# Example: covariance matrices from spatial statistics

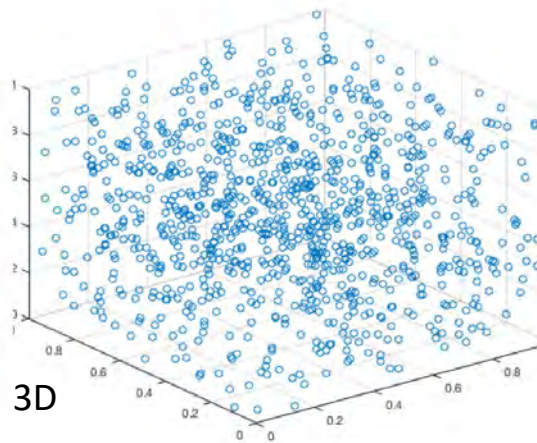
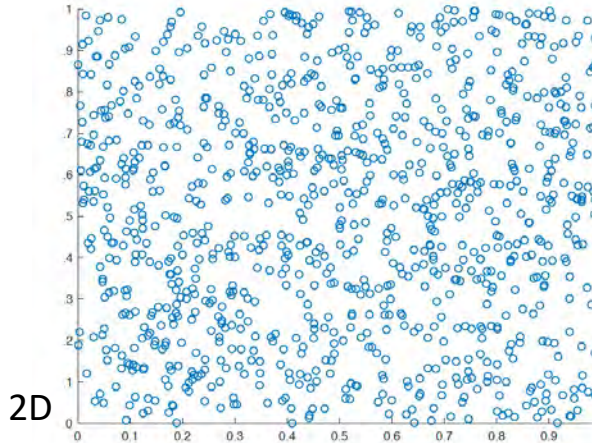
- Climate and weather applications have many measurements located regularly or irregularly in a region; prediction is needed at other locations
- Modeled as realization of Gaussian or Matérn spatial random field, with parameters to be fit
- Leads to evaluating, inside an optimization loop, the log-likelihood function involving a large dense (but data sparse) covariance matrix  $\Sigma$

$$\ell(\boldsymbol{\theta}) = -\frac{1}{2}\mathbf{Z}^T \Sigma^{-1}(\boldsymbol{\theta})\mathbf{Z} - \frac{1}{2}\log|\Sigma(\boldsymbol{\theta})|$$

- Apply inverse  $\Sigma^{-1}$  and determinant  $|\Sigma|$  with Cholesky

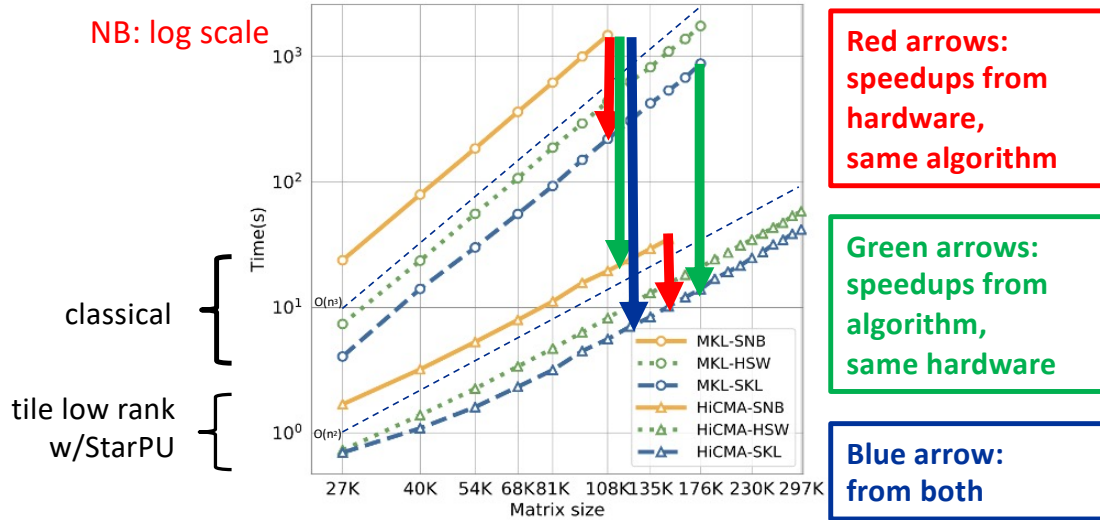
# Synthetic scaling test

Random coordinate generation within the unit square or unit cube with Matérn kernel decay, each pair of points connected by square exponential decay,  $a_{ij} \sim \exp(-c|x_i - x_j|^2)$



# HiCMA TLR vs. Intel MKL on shared memory

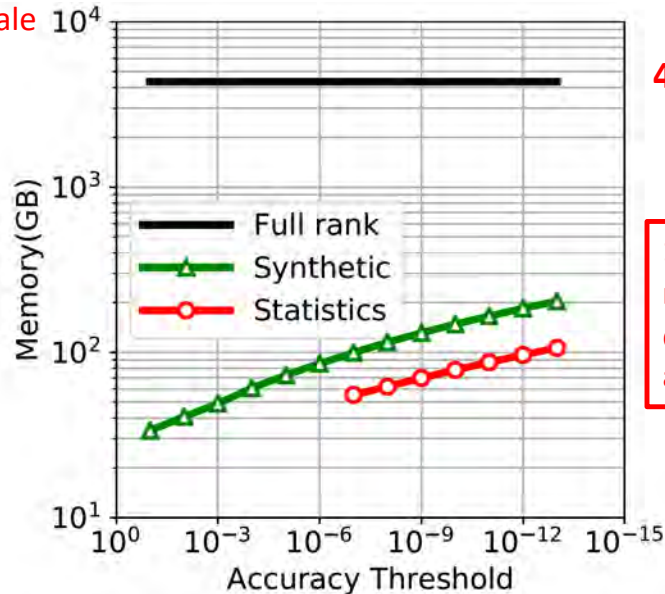
- Gaussian kernel to accuracy  $1.0e-8$  in each tile
- Three generations of Intel manycore (Sandy Bridge, Haswell, Skylake)
- Two generations of linear algebra (classical dense and tile low rank)





# Memory footprint for TLR fully DP matrix of size 1M

NB: log scale

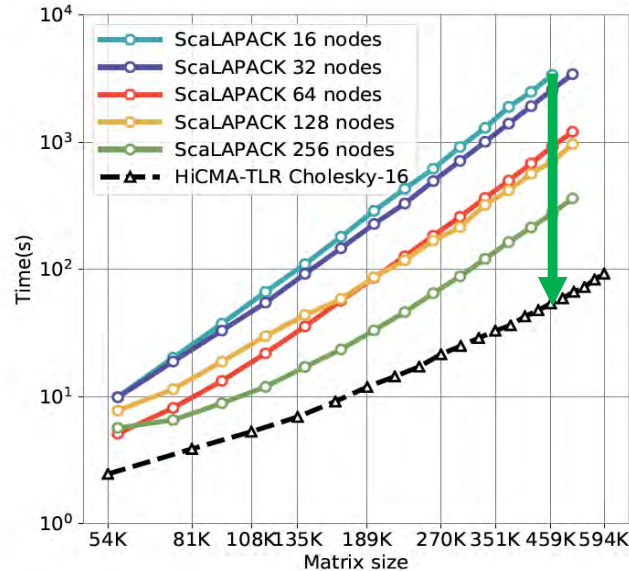


4 TB

1 to 2 orders of magnitude less, depending upon accuracy (x-axis)

# HiCMA TLR vs. ScaLAPACK on distributed memory

NB: log scale



Green arrow:  
speedup from  
algorithm,  
same 16 nodes

Shaheen II at KAUST: a Cray XC40 system with 6,174 compute nodes, each of which has two 16-core Intel Haswell CPUs running at 2.30 GHz and 128 GB of DDR4 main memory

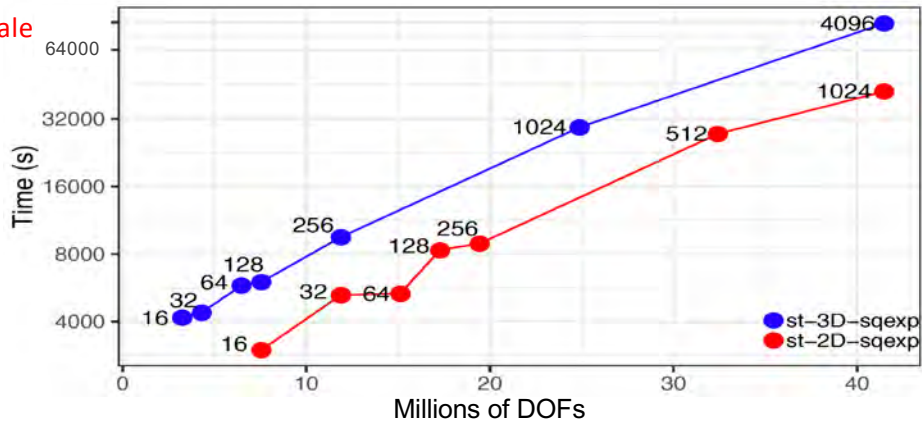
Akbudak, Ltaief, Mikhalev, Charara & K., *Exploiting Data Sparsity for Large-scale Matrix Computations*, Euro-Par 2018

# Extreme Tile Low Rank

Cholesky factorization of a TLR matrix derived from Gaussian covariance of random distributions, up to 42M DOFs, on up to 4096 nodes (131,072 cores) of a Cray XC40

- would require 7.05 PetaBytes in dense DP (using symmetry)
- would require 77 days by ScaLAPACK (at the TLR rate of 3.7 Pflop/s)

NB: log scale



Fully dense computation would have cost about \$1.03M in electricity and generated about 2500 metric tons of CO<sub>2</sub>e

Cao, Pei, Akbudak, Mikhalev, Bosilca, Ltaief, K. & Dongarra, *Extreme-Scale Task-Based Cholesky Factorization Toward Climate and Weather Prediction Applications*. PASC'20 (ACM)

# Two motivations for mixed precision

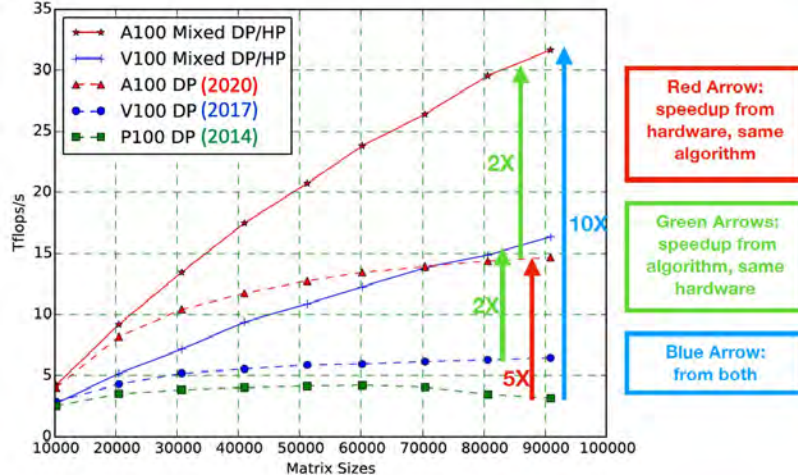
- Mathematical: (much) better than “no precision”
  - Statisticians often approximate remote diagonals as *zero* after performing a diagonally clustered space-filling curve ordering, so their coefficients must be orders of magnitude down from the diagonals
  - not just *smoothly decaying* in the low-rank sense, but actually *small*
- Computational: faster time to solution
  - hence lower energy consumption and higher performance, especially by exploiting heterogeneity

Peak Performance in TF/s	V100 NVLink	A100 NVLink	H100 SXM
FP64	7.5	9.7	34
FP32		19.5	67
FP64 Tensor Core	15	19.5	67
FP32 Tensor Core	8x	156	495
FP16 Tensor Core	120	312	989
	rel. 2017	rel. 2020	rel. 2023

The table shows performance improvements over time. Red curved arrows point from the 2017 column to the 2020 column, and from the 2020 column to the 2023 column. Red text labels '8x' and '16x' are placed between the columns to indicate the performance gain. For FP32 Tensor Core, the gain from 2017 to 2020 is 8x (156 / 19.5). For FP16 Tensor Core, the gain from 2020 to 2023 is 16x (989 / 61.8, where 61.8 is the sum of FP32 and FP64 Tensor Core performance in 2020).

# Mixed precision geospatial statistics on GPUs

- Gaussian kernel to accuracy  $1.0e-9$  in each tile
- Three generations of NVIDIA GPU (Pascal, Volta, Ampere)
- Two generations of linear algebra (double precision and mixed DP/HP)



Ltaief, Genton, Grataour, K. & Ravasi, 2022, *Responsibly Reckless Matrix Algorithms for HPC Scientific Applications*, Computing in Science and Engineering

# 2022 Gordon Bell Finalist justification

## Reshaping Geostatistical Modeling and Prediction for Extreme-Scale Environmental Applications

### I. JUSTIFICATION FOR THE GORDON BELL PRIZE

Synergistic combination of mixed-precision computations and low-rank matrix approximations. Dynamic task-based runtime system and data movement. Scalability on 16K Fugaku nodes (786,432 cores) for maximum log-likelihood estimation (MLE). Performance speedup up to 12X over FP64 execution while attaining application-worthy accuracy. Incorporation into path-finding software framework for geostatistical applications.

# 2022 Gordon Bell Finalist attributes

## II. PERFORMANCE ATTRIBUTES

Performance Attributes	Our submission
Problem Size	Nine million geospatial locations <sup>1</sup>
Category of achievement	Time-to-solution and scalability
Type of method used	Maximum Likelihood Estimation (MLE)
Results reported on basis of	Whole application
Precision reported	Double, single, and half precision
System scale	16K Fujitsu A64FX nodes of Fugaku <sup>1</sup>
Measurement mechanism	Timers; FLOPS; Performance modeling

# GB'22 collaborators

**KAUST Supercomputing Core Lab, HLRS-Stuttgart, Oak Ridge LCF, RIKEN, and:**



Qinglei Cao



Yu Pei



George Boslica



Jack Dongarra



Rabab Alomairy



Pratik Nag



Sameh Abdulah



Hatem Ltaief



Ying Sun

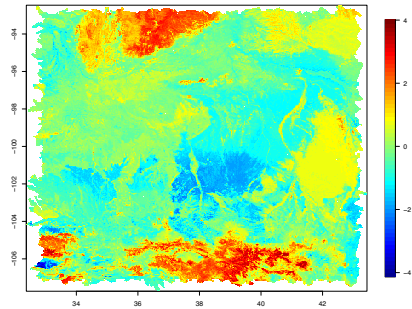


Marc Genton

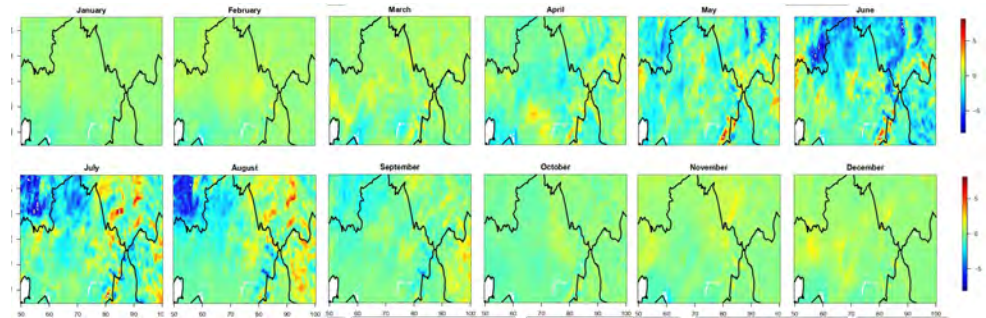


# App: spatial & spatio-temporal environmental statistics

Space and space-time modeling using Maximum Likelihood Estimation (MLE) on two environmental datasets



2D soil moisture data  
at the top layer of the  
Mississippi River basin



2021 monthly evapotranspiration (ET)  
over Central Asia

[means are subtracted out in these plots]

# Statistical “emulation” (complementary to simulation)

- Predicts quantities directly from data (*e.g.*, weather, climate)
  - assumes a correlation model
  - data may be from observations or from first-principles simulations
  - statistical alternative to large-ensemble simulation averages
- Relied upon for economic and policy decisions
  - predicting demands, engineering safety margins, mitigating hazards, siting renewable resources, etc.
  - such applications are among principal supercomputing workloads
- Whereas simulations based on PDEs are usually memory bandwidth-bound, emulations based on covariance matrices are usually compute-bound (achieve a high % of bandwidth peak)

# The computational challenge

- Contemporary observational datasets can be huge
  - Collect  $p$  observations at each of  $n$  locations  $Z_p(x_n, y_n, z_n, t_n)$
  - Find optimal fit of the observations  $Z$  to a plausible function
  - Infer values at missing locations of interest
- Maximum Likelihood Estimate (MLE)
  - model for estimating parameters required to perform inference
- Complexity:
  - Arithmetic cost: solve systems with and calculate determinant of  $n$ -by- $n$  covariance matrix
  - $O((pn)^3)$  floating-point operations and  $O((pn)^2)$  memory
  - Memory footprint:  $10^6$  locations require 4 TB memory (double precision, invoking symmetry, for  $p=1$ )

# The computational challenge opportunity

- Contemporary observational datasets can be huge
  - Collect  $p$  observations at each of  $n$  locations  $Z_p(x_n, y_n, z_n, t_n)$
  - Find optimal fit of the observations  $Z$  to a plausible function
  - Infer values at missing locations of interest
- Maximum Likelihood Estimate (MLE)
  - model for estimating parameters required to perform inference
- Complexity:
  - Arithmetic cost: solve systems with and calculate determinant of  $n$ -by- $n$  covariance matrix
  - $O((pn)^3)$  floating-point operations and  $O((pn)^2)$  memory
  - Memory footprint:  $10^6$  locations require 4 TB memory (double precision, invoking symmetry, for  $p=1$ )

# Motivation: High Performance Computational Statistics (HPCS)

“Increasing amounts of data are being produced (e.g., by remote sensing instruments and numerical models), while techniques to handle millions of observations have historically lagged behind... Computational implementations that work with irregularly-spaced observations are still rare.” - Dorit Hammerling, NCAR, July 2019



1M × 1M dense sym DP matrix requires 4 TB,  $N^3 \sim 10^{18}$  Flops

Traditional approaches:

Global low rank

Zero outer diagonals

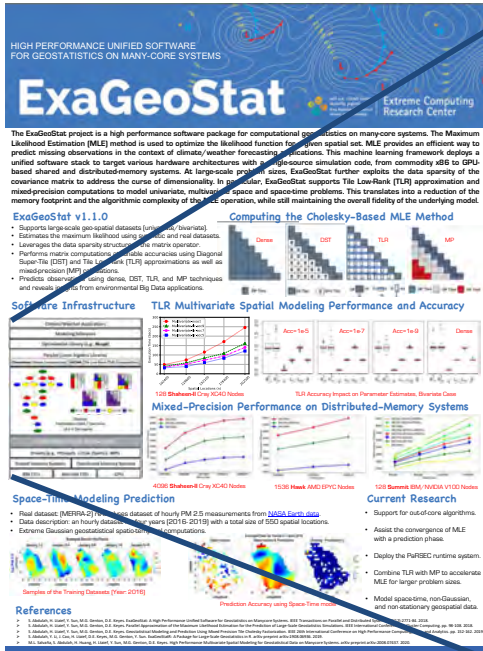
Better approaches:

Hierarchical low rank

Reduced precision outer  
diagonals



<https://github.com/ecrc/exageostat>



**HIGH PERFORMANCE UNIFIED SOFTWARE FOR GEOSTATISTICS ON MANY-CORE SYSTEMS**

# ExaGeoStat

Extreme Computing Research Center

The ExaGeoStat project is a high performance software package for computational geostatistics on many-core systems. The Maximum Likelihood Estimation (MLE) method is used to optimize the likelihood function for given spatial set. MLE provides an efficient way to predict missing observations in the context of climate/weather forecasting applications. This machine learning framework deploys a unified software stack to target various hardware architectures with a single-source simulation code, from commodity x86 to GPU-based shared and distributed-memory systems. At large-scale, ExaGeoStat further exploits the data sparsity of the covariance matrix to address the curse of dimensionality. In particular, ExaGeoStat supports Tile Low-Rank (TLR) approximation and mixed-precision computations to model variable, multi-scale, sparse and space-time problems. This translates into a reduction of the memory footprint and the algorithmic complexity of the MLE operation, while still maintaining the overall fidelity of the underlying model.

**ExaGeoStat v1.1.0**

- Supports large-scale geo-spatial datasets (e.g., US, Slovenia)
- Estimates the maximum likelihood using synthetic and real datasets
- Leverages the data sparsity structure for sparse matrix operation
- Performs matrix computations with high accuracies using Diagonal Super-Tile (DST) and Tile Low-Rank (TLR) approximations as well as mixed-precision (MP) computation
- Predicts observations using dense, DST, TLR, and MP techniques and results are from experimental Big Data applications

**Computing the Cholesky-Based MLE Method**

**Software Infrastructure**

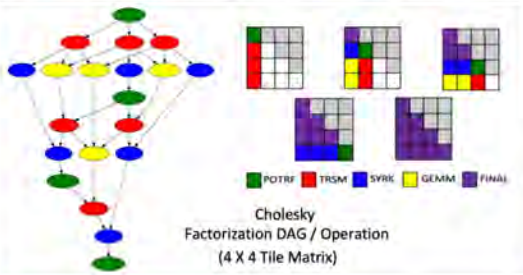
**TLR Multivariate Spatial Modeling Performance and Accuracy**

**Mixed-Precision Performance on Distributed-Memory Systems**

**Space-Time Modeling Prediction**

**References**

**Current Research**

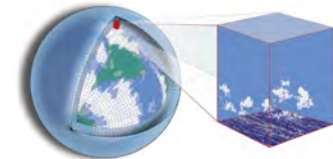
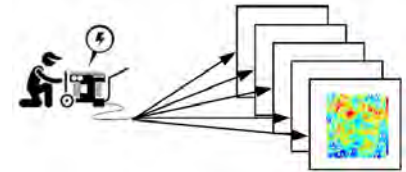
Climate/Weather Applications	
Modeling/Inference	
Optimization Library (e.g., NLOPT)	
Parallel Linear Algebra Libraries	
Chameleon (Dense Computation)	HICMA (Tile Low-Rank (TLR) Computation)
 <p>Cholesky Factorization DAG / Operation (4 X 4 Tile Matrix)</p> <p>Legend: POTRF (Green), TRSM (Red), SYRK (Blue), GEMM (Yellow), FINAL (Purple)</p>	
StarPU/ParSEC Dynamic Runtime System	
Drivers (e.g., Pthreads, CUDA, OpenCL, MPI)	
Shared Memory Systems	Distributed Memory Systems
X86 CPU	AArch64 CPU
	GPU



Sameh Abdulah,  
Research Scientist  
ECRC, KAUST

# ExaGeoStat's 3-fold framework

- Synthetic Dataset Generator
  - Generates large-scale geospatial datasets which can be used separately as benchmark datasets for other software packages
- Maximum Likelihood Estimator (MLE)
  - Evaluates the maximum likelihood function on large-scale geospatial datasets
  - Supports dense full machine precision, Tile Low-Rank (TLR) approximation, low-precision approximation accuracy, and now TLR-MP
- ExaGeoStat Predictor
  - Infers unknown measurements at new geospatial locations from the MLE model

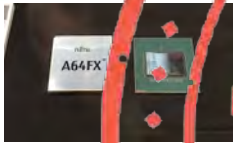


# The portable ExaGenStat software stack

Nov 2023  
Top500 data



AMD EPYC



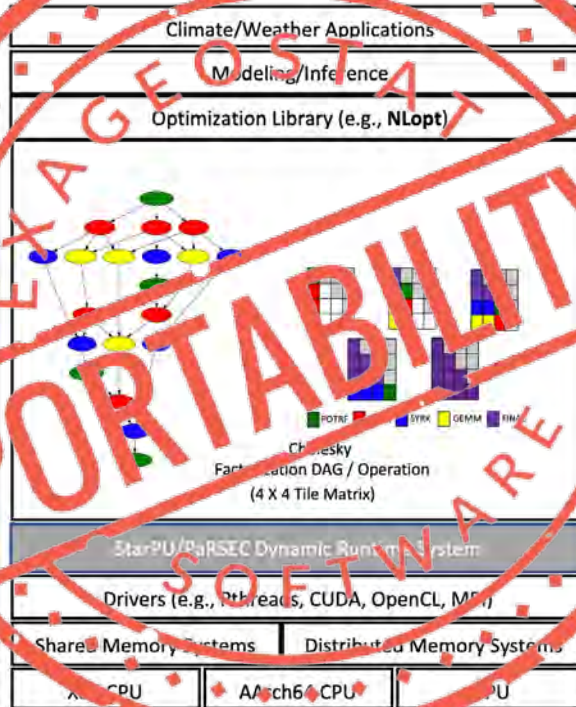
Fujitsu A64FX



NVIDIA V100



Intel X86



#1 Frontier



#4 Fugaku



#7 Summit



#141 Shaheen-2



# Maximum Likelihood Estimator (MLE)

- The log-likelihood function:  $\ell(\boldsymbol{\theta}) = -\frac{n}{2} \log(2\pi) - \frac{1}{2} \log |\boldsymbol{\Sigma}(\boldsymbol{\theta})| - \frac{1}{2} \mathbf{Z}^\top \boldsymbol{\Sigma}(\boldsymbol{\theta})^{-1} \mathbf{Z}$ .
- Optimization over  $\boldsymbol{\theta}$  to maximize the likelihood function estimation until convergence
  - generate the covariance matrix  $\boldsymbol{\Sigma}(\boldsymbol{\theta})$  using a specified kernel
  - evaluate the log determinant and the inverse operations, which require a Cholesky factorization of the given covariance matrix
  - update  $\boldsymbol{\theta}$
- NLOPT\* is typically used to maximize the likelihood
- Parallel PSwarm optimization algorithm runs several likelihood estimation steps at the same time (an embarrassingly parallel outer loop)

\*open-source library by Prof. Steve Johnson of MIT

# Covariance functions supported in ExaGeoStat

Univariate Matern Kernel

$$C(r; \theta) = \frac{\theta_1}{2^{\theta_3-1} \Gamma(\theta_3)} \left( \frac{r}{\theta_2} \right)^{\theta_3} \mathcal{K}_{\theta_3} \left( \frac{r}{\theta_2} \right)$$

(3 parameters to fit: variance, range, smoothness)

Space/Time Nonseparable Kernel

$$C(\mathbf{h}, u) = \frac{\sigma^2}{a_t |u|^{2\alpha} + 1} \mathcal{M}_\nu \left\{ \frac{\|\mathbf{h}\|/a_s}{(a_t |u|^{2\alpha} + 1)^{\beta/2}} \right\}$$

(6 parameters to fit, add: time-range, time-smoothness, and separability)

Multivariate Parsimonious Kernel

$$C_{ij}(\|\mathbf{h}\|; \theta) = \frac{\rho_{ij} \sigma_{ii} \sigma_{jj}}{2^{\nu_{ij}-1} \Gamma(\nu_{ij})} \left( \frac{\|\mathbf{h}\|}{a} \right)^{\nu_{ij}} \mathcal{K}_{\nu_{ij}} \left( \frac{\|\mathbf{h}\|}{a} \right)$$

Tukey g-and-h Non-Gaussian Field with Kernel

$$\rho_Z(h) = \frac{1}{\Gamma(\nu) 2^{\nu-1}} \left( 4\sqrt{2\nu} \frac{h}{\phi} \right)^\nu \mathcal{K}_\nu \left( 4\sqrt{2\nu} \frac{h}{\phi} \right)$$

Multivariate Flexible Kernel

$$C(\mathbf{h}; u) = \frac{\sigma^2}{2^{\nu-1} \Gamma(\nu)} (a|u|^{2\alpha} + 1)^{\delta+\beta d/2} \left( \frac{c\|\mathbf{h}\|}{(a|u|^{2\alpha} + 1)^{\beta/2}} \right)^\nu \\ \times \mathcal{K}_\nu \left( \frac{c\|\mathbf{h}\|}{(a|u|^{2\alpha} + 1)^{\beta/2}} \right), \quad (\mathbf{h}; u) \in \mathbb{R}^d \times \mathbb{R},$$

Powered Exponential Kernel

$$C(r; \theta) = \theta_0 \exp \left( \frac{-r^{\theta_2}}{\theta_1} \right)$$

# How to choose the rank?

- Tiles are compressed to low rank based on user-supplied tolerance parameter, based on the first neglected singular value-vector pair.
- A tile-centric, structure-aware heuristic decides at runtime whether the tile should remain in low rank form or converted back to dense, based on estimates of the overheads of maintaining and operating with the compressed form.
- The structure-aware runtime decision is based only the estimated number of flops and time to solution, while the precision-aware runtime decision (next slide) is based only on the accuracy requirements of representing the matrix in the Frobenius norm.

# How to choose the precision?

- Consider 2-precision case, with machine epsilons (unit roundoffs)  $u_{high}$  and  $u_{low}$ , resp.
- Let  $\|A\|_F$  be the Frobenius norm of the global matrix square matrix  $A$ , which is computable by streaming  $A$  through just once
- Let  $n_T$  be the number of tiles in each dimension of  $A$
- Then any tile  $A_{ij}$  such that

$$\|A_{ij}\|_F / (\|A\|_F / n_T) < u_{high} / u_{low}$$

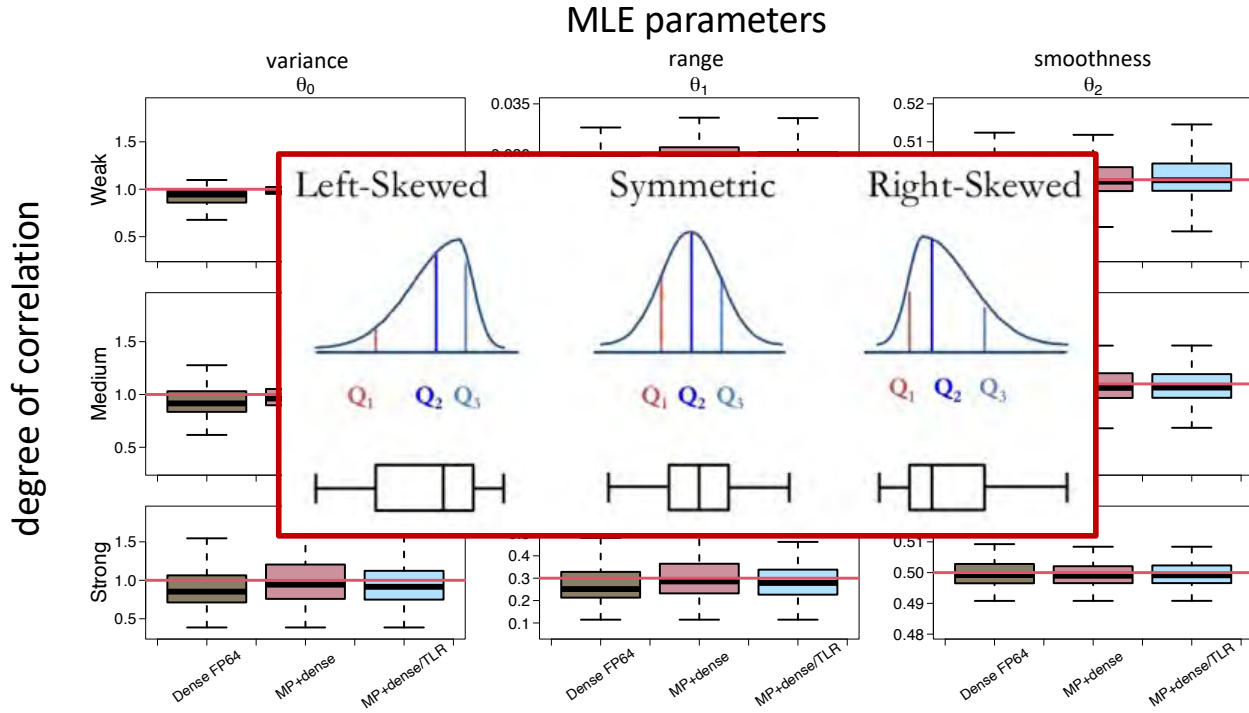
is stored in low precision; otherwise kept in high

- The mixed precision tiled matrix  $\mathcal{A}$  thus formed satisfies

$$\|\mathcal{A} - A\|_F < u_{high} \|A\|_F$$

- Generalizes to multiple precisions
- Tiles can be converted dynamically at runtime

# Accuracy on synthetic 2D space dataset



# Accuracy on real 3D (2D space + time) dataset

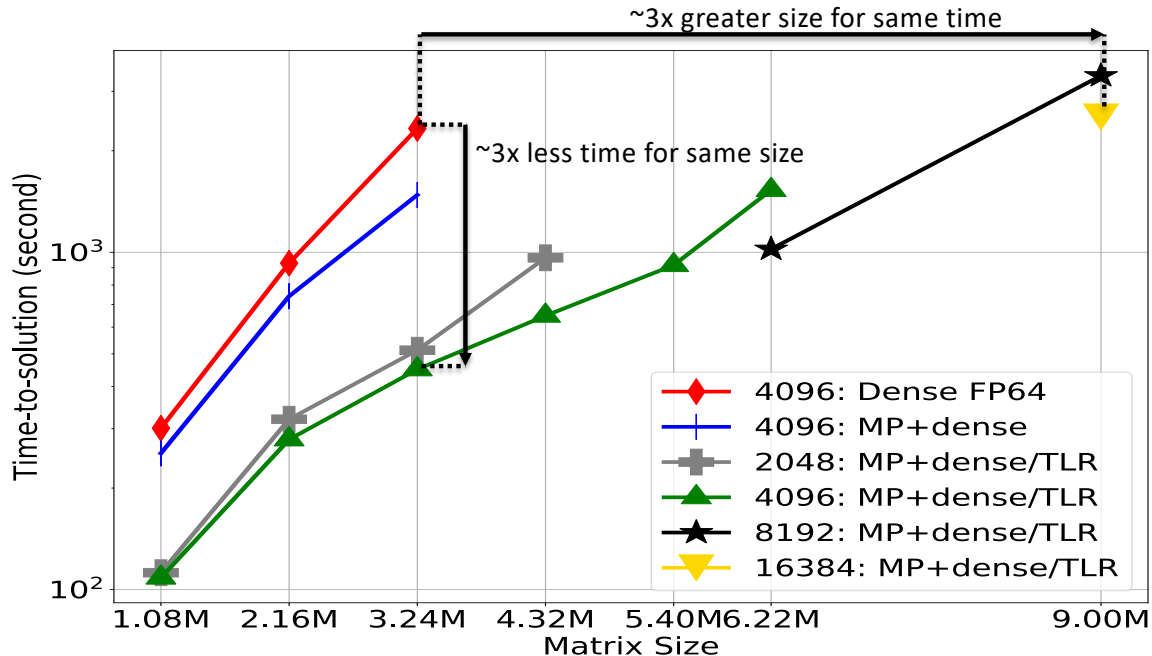
Variants	Variance ( $\theta_0$ )	Range ( $\theta_1$ )	Smoothness ( $\theta_2$ )
Dense FP64	1.0087	3.7904	0.3164
MP+dense	0.9428	3.8795	0.3072
MP+dense/TLR	0.9247	3.7756	0.3068

Variants	Range-time ( $\theta_3$ )	Smoothness-time ( $\theta_4$ )	Nonsep-param ( $\theta_5$ )
Dense FP64	0.0101	3.4890	0.1844
MP+dense	0.0102	3.4941	0.1860
MP+dense/TLR	0.0102	3.5858	0.1857

Variants	Log-Likelihood (llh)	MSPE
Dense FP64	-136675.1	0.9345
MP+dense	-136529.0	0.9348
MP+dense/TLR	-136541.8	0.9428

mean-square  
prediction error

# Performance on up to 16K nodes of Fugaku



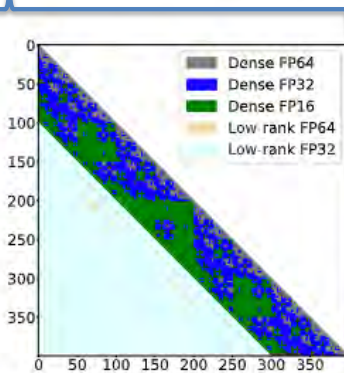
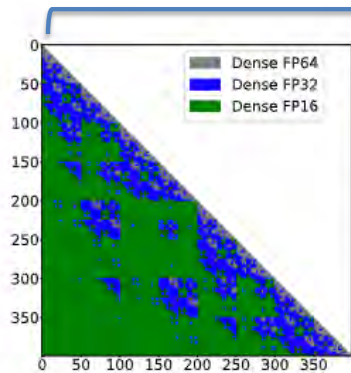
**To be improved:**  
Still tuning runtime system PaRSEC on Fugaku's 32GB/node

# Tile map for 2D space kernel with $\sim 1\text{M}$ points

370 tiles of size 2700 in each dimension

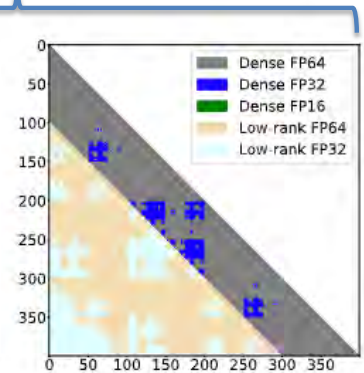
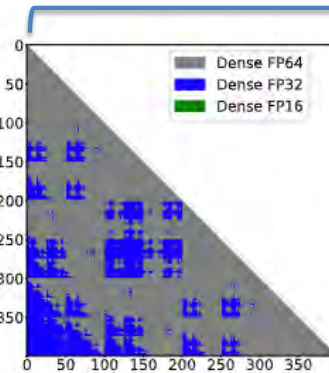
weak correlation

strong correlation



memory footprint  
1.6 TB

memory footprint  
0.9 TB



memory footprint  
3.8 TB

memory footprint  
1.8 TB

default dense double is  $\sim 4$  TB



# Impact for spatial statistics



*The potential for this combination in spatial statistics generally is high... The authors have demonstrated **controllable and high accuracy typical of universal double precision, while exploiting mostly half precision, and keeping relatively few tiles clustered around the diagonal in their original fully dense format.** The result is reduction in time to solution of an order of magnitude or more, with the ratio of improvement growing with problem size, but already transformative.*

-- Professor Sudipto Banerjee, UCLA

# Impact for spatial statistics



*The innovations described in numerical linear algebra and in dynamic runtime task scheduling deliver an order of magnitude or more of reduction in execution time for a sufficiently large spatial or spatial-temporal data set using the Maximum Likelihood Estimation (MLE) and kriging paradigm. Perhaps more importantly, **by reducing the memory footprint of such models, they allow much larger datasets to be accommodated within given computational resources.** The advance this creates for spatial statisticians – geophysical and otherwise – is **potentially immense**, given that this result is now available through ExaGeoStat.*

-- Professor Doug Nychka, Colorado School of Mines

# Impact for spatial statistics



*An especially attractive aspect of the submission is the innovation that it required in the a64fx ARM architecture of Fugaku, namely the accumulation in 32 bits of the 16-bit floating point multiply. I regard this aspect of the KAUST-UT-RIKEN collaboration of abiding benefit beyond the particular application of this submission.*

*As you know, my mottos for data science are that “Statistics is the ‘Physics’ of Data” and “Statistics is to Machine Learning as Physics is to Engineering.” **Your Gordon Bell campaign is accelerating the use of spatial statistics to allow it to keep up with exascale hardware.***

-- Dr. George Ostrouchov, ORNL

# 2023 Gordon Bell Finalist justification

## Scaling the “Memory Wall” for Multi-Dimensional Seismic Processing with Algebraic Compression on Cerebras CS-2 Systems

### I. JUSTIFICATION FOR THE GORDON BELL PRIZE

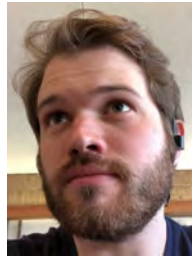
High-performance matrix-vector multiplication using low-rank approximation. Memory layout optimizations and batched executions on massively parallel Cerebras CS-2 systems. Leveraging AI-customized hardware capabilities for seismic applications for a low-carbon future. Application-worthy accuracy (FP32) with a sustained bandwidth of 92.58PB/s (for 48 CS-2s) would constitute the third-highest throughput from June'23 Top500.

# 2023 Gordon Bell Finalist attributes

Performance Attributes	Our submission
Problem Size	Broadband 3D seismic dataset ( $\sim 20k$ sources and receivers and frequencies up to $50Hz$ )
Category of achievement	Sustained bandwidth Scalability
Type of method used	Algebraic compression
Results reported on basis of	Whole application (for GPU cluster) Main kernel (for Cerebras cluster)
Precision reported	Single precision complex
System scale	Up to 48 Cerebras CS-2 systems, i.e., 35,784,000 processing elements <sup>1</sup>
Measurement mechanism	Timers; Memory accesses; Performance modeling

# GB'23 collaborators

**Group42 (Abu Dhabi), KAUST Supercomputing Core Lab and:**



Leighton Wilson



Mathias Jacquelin



Yuxi Hong



Hatem Ltaief



Matteo Ravasi

# Cerebras CS-2 Wafer-Scale Engine (WSE)



# 2023 Gordon Bell submission

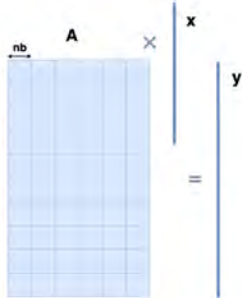


Fig. 2: Original dense MVM.

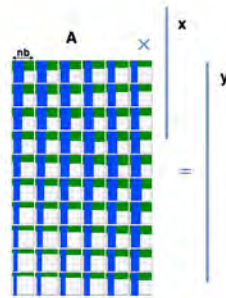


Fig. 3: Rank-compressed operator.

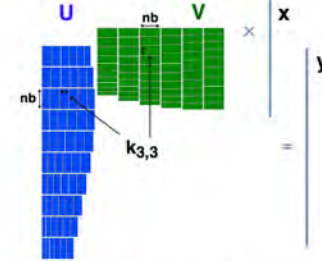


Fig. 4: Stacked bases  $U$  and  $V$ .

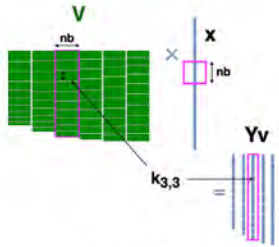


Fig. 5:  $V$ -batch stage of MVM.

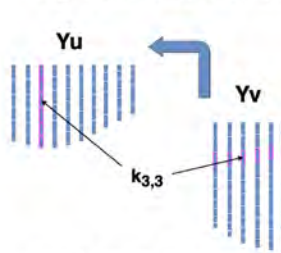


Fig. 6: Shuffle from  $V$  to  $U$  bases.

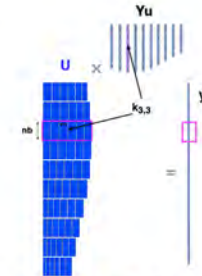
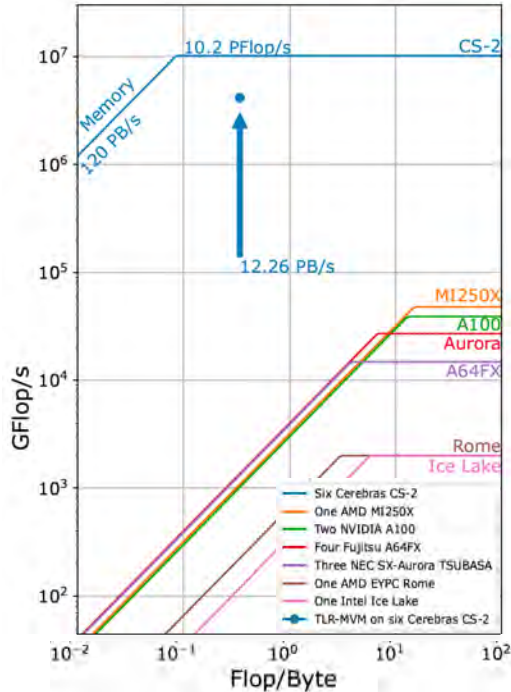


Fig. 7:  $U$ -batch of MVM.

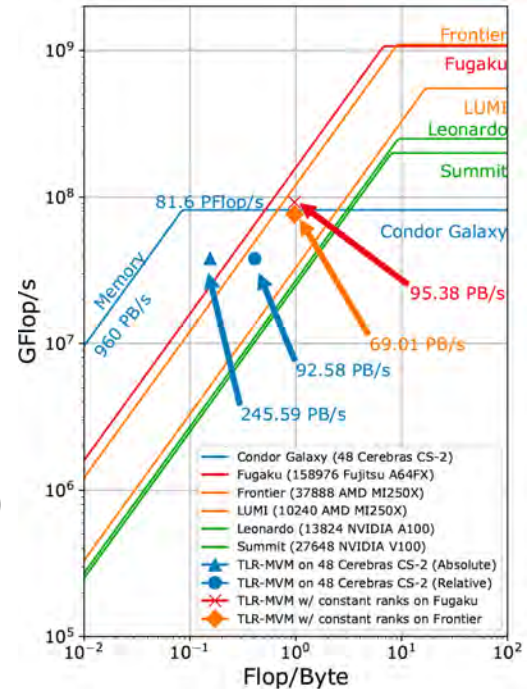


# 2023 Gordon Bell submission

Roofline models of 6 CS-2's compared with other solutions



Roofline models of 48 CS-2's compared with top 5 (Nov 2023)



# Impact for geophysical imaging



*For the past 3 decades, we have needed large-scale convolutions for multiple applications to tackle subsurface challenges – which are now greater than ever for the energy transition, such as rapid, wide-scale monitoring of subsurface hydrogen storage – but have never achieved it due to the unsurmountable bottleneck imposed by the size of datasets (starting at TBs).*

***This project, with its balanced focus on accuracy and practical performance, is likely to finally break through a decades-old barrier in geophysical imaging.***

– Dr. Ivan Vasconcelos, Shearwater Geoservices

# Impact for geophysical imaging



*The impact that the efficient implementation of multi-dimensional convolution with low-rank tiles that Ltaief and co-authors have developed is better understood if we bear in mind that multidimensional convolution and deconvolution are ubiquitous operations in seismic processing.*

***This new implementation may lead to a drastic reduction of the turnaround time of seismic data processing projects. The consequence is that the decision-makers, regardless of whether they use seismic images for conventional hydrocarbon exploration or for other applications, will receive valuable information in a timely manner.***

– Dr. Claudio Bagaini, SLB (Schlumberger)

# Impact for geophysical imaging



*Conventional algorithms for MDD would not have mapped onto the Cerebras CS-2 engines because their  $N^3$  arithmetic complexity is prohibitive. Only the algebraically compressed form of the problem fits. All parts of this interdisciplinary project are thus necessary for its success.*

***As the title indicates, this team is ‘scaling the memory wall’ that has loomed over computational science & engineering at the high end for, by now, three decades. Their algorithms and CS-2 implementation have enormous implications for our community, since their application is representative of many important CS&E problems.***

– Professor Omar Ghattas, U Texas

# 2024 Gordon Bell Climate Finalist justification

## Boosting Earth System Model Outputs And Saving PetaBytes in their Storage Using Exascale Climate Emulators

### I. JUSTIFICATION FOR THE GORDON BELL PRIZE

Exascale climate emulator developed using 318 billion hourly and 31 billion daily observations for generating climate emulations at ultra-high spatial resolution ( $0.034^\circ \sim 3.5$  km). Modeling climate data using spherical harmonics. Mixed-precision computations. PaRSEC dynamic runtime system. Running on 9,025 nodes on Frontier, 1,936 nodes on Alps, 1,024 nodes on Leonardo, and 3,072 nodes on Summit, with the hybrid Flop/s rates 0.976 EFlop/s, 0.739 EFlop/s, 0.243 EFlop/s, and 0.375 EFlop/s, respectively.

# 2024 Gordon Bell Climate Finalist attributes

Problem size	54,486,360 spatial locations across the globe at a spatial resolution of 0.034° (~3.5 km)
Category of achievement Type of method used	Scalability and peak performance Spherical Harmonic Transform (SHT) and Cholesky factorization
Results reported on basis of Precision reported System scale	Cholesky factorization Double and mixed-precision - 0.976 EFlop/s on 9,025 nodes of Frontier (36,100 AMD MI250X multi-chip module (MCM) GPUs) equivalent to 72,200 AMD Graphics Compute Dies (GCDs) - 0.739 EFlop/s on 1,936 nodes of Alps (7,744 NVIDIA Grace-Hopper Superchips (GH200)) - 0.243 EFlop/s on 1,024 nodes of Leonardo (4,096 NVIDIA A100 GPUs) - 0.375 EFlop/s on 3,072 nodes of Summit (18,432 NVIDIA V100 GPUs)
Measurement mechanism	Timers, Flops

# GB'24 climate prize collaborators

KAUST Supercomputing Core Lab, Oak Ridge LCF, CSCS Alps, CINECA Leonardo, and:



Allison Baker



George Boslica



Qinglei Cao



Stefano Castruccio



Gera Stenchikov



Sameh Abdulah



Marc Genton



Zubair Khalid



Hatem Ltaief



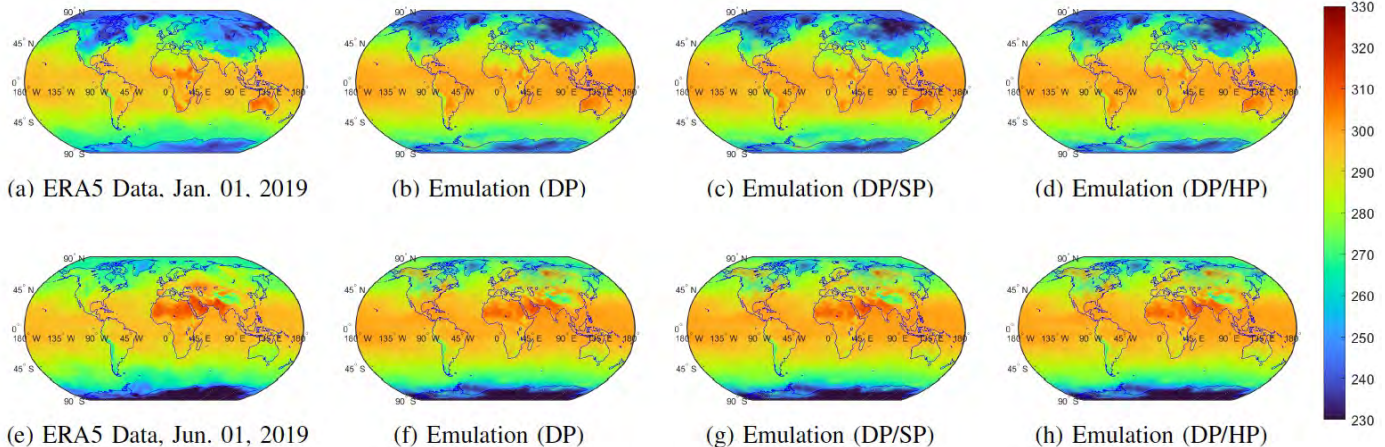
Yan Song



Ying Sun

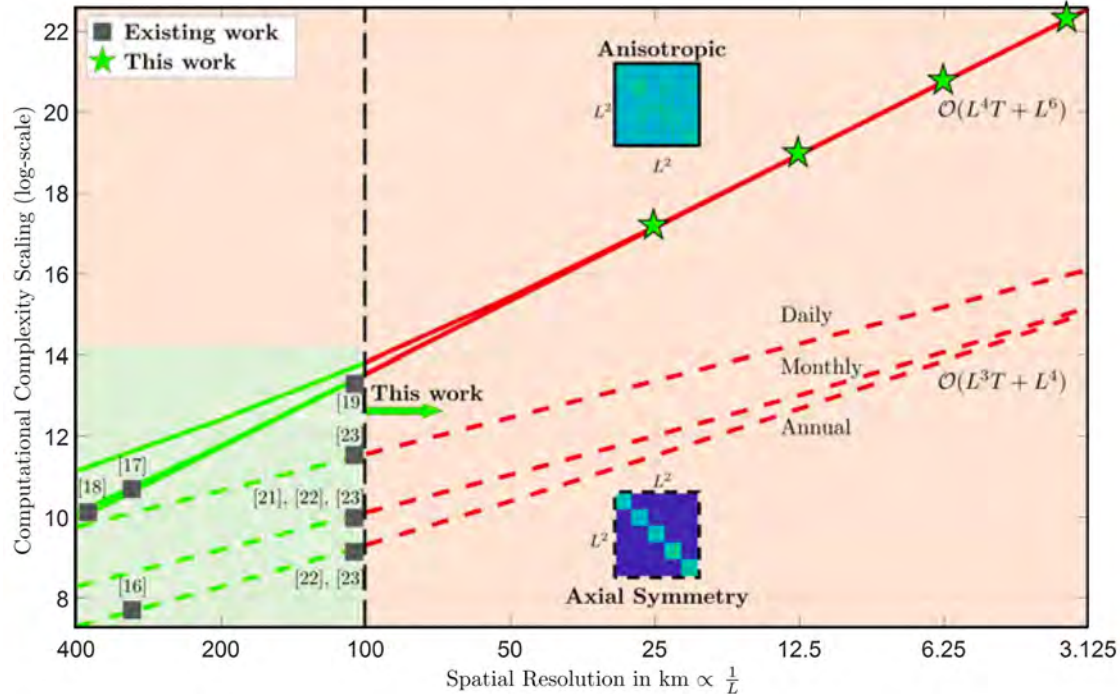
# Motivation for climate emulation

- 45-institution Coupled Model Intercomparison Project (CMIP)
- CMIP6 campaign recently generated 28 PB of data, at a cost of \$45/TB/year
- emulation trained on simulation generates realizations with same statistics
- efficient basis for compact storage, e.g., spherical harmonics





# Expanding emulation capabilities



# Performance on four Top10 systems (eff. Pflops/s)

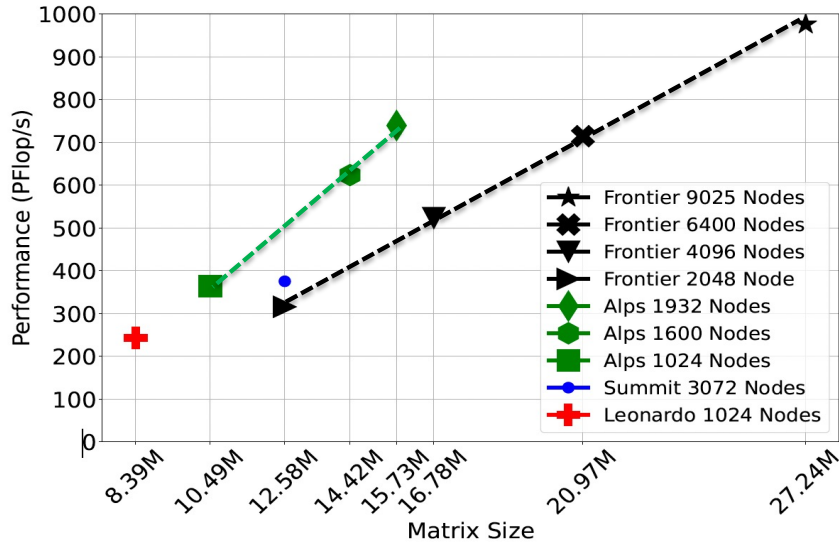


Fig. 8: Performance of largest runs on Summit, Leonardo, Alps, and Frontier; with additional run-up points on Alps and Frontier, all using the DP/HP precision variant.

# Per node performance on a 1024-node subsystem

TABLE I: DP/HP Performance on 1,024 nodes of the four systems, i.e., Frontier, Alps, Leonardo, and Summit.

System	Frontier	Alps	Leonardo	Summit
Vendor	AMD	NVIDIA		
Chip	MI250X	GH200	A100	V100
# GPUs	4,096	4,096	4,096	6,144
Matrix Size	8.39M	10.49M	8.39M	6.29M
Performance (PFflop/s)	223.7	384.2	243.1	153.6
TFlop/s/GPU	54.6	93.8	57.2	25

# 2024 Gordon Bell Prize Finalist justification

## Toward Capturing Genetic Epistasis From Multivariate Genome-Wide Association Studies Using Mixed-Precision Kernel Ridge Regression

### I. JUSTIFICATION FOR THE GORDON BELL PRIZE

High-performance tile-centric matrix computations for kernel ridge regression. End-to-end GWAS software supporting the largest-ever multivariate study of 305K patients from UK BioBank. Application-worthy FP64 accuracy using four precisions. Sustained throughput exceeding 11X over FP 64 on A100s. Near-perfect weak-scaling on one-third of Leonardo, projecting to 2 MP Eflops/s on full system.

# 2024 Gordon Bell Prize Finalist attribution

## II. PERFORMANCE ATTRIBUTES

Performance Attributes	Value
Problem Size	305K patients [real data, UK BioBank] 300K patients [synthetic data, msprime] 13M patients [synthetic data, random fill]
Category of achievement	Scalability, performance, time to solution
Type of method used	Kernel Ridge Regression
Results reported on basis of	Whole GWAS application: - mixed precision distance computation - mixed precision Cholesky factorization - mixed precision triangular solve
Precision reported	FP64, FP32, FP16, FP8, INT8
System scale	2/3 of Summit (18,432 V100 GPUs) 1/3 of Leonardo (4,096 A100 GPUs) full Frontier (36,100 MI250X GPUs) 4/5 of Alps (8,100 GH200 Superchips) Sustained 1.805 mixed precision ExaOp/s
Measurement mechanism	Timers, Flops

# GB'24 prize collaborators

KAUST Supercomputing Core Lab, Oak Ridge LCF, CSCS Alps, CINECA Leonardo, and:



Rached Abdelkhalek



Rabab Alomairy



Qinglei Cao



Benedikt Dorschner



Thorsten Kurth



Lotfi Slim



Salim Bougaffa



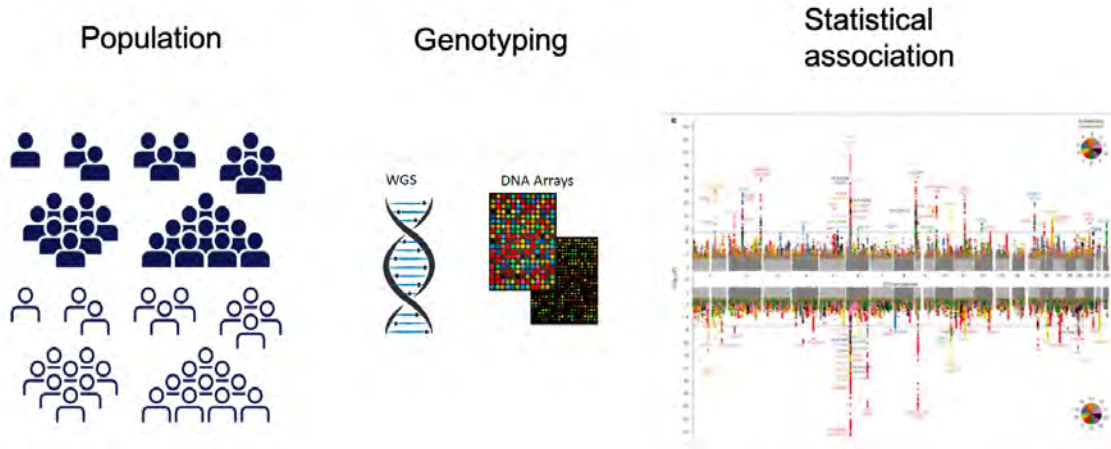
Hatem Ltaief



Jie Ren

# Motivation for epistatic genome association studies

- Train statistical model on genotype/environmental-to-phenotype data
- Use to predict disease and other genetic/environmental characteristics
- Ridge regression is a linear association that considers individual SNPs
- Kernel ridge regression is correlates instances of multiple SNPs



# Submitted to the 2024 Gordon Bell Prize

TABLE I: Comparing RR vs. KRR Pearson correlations.

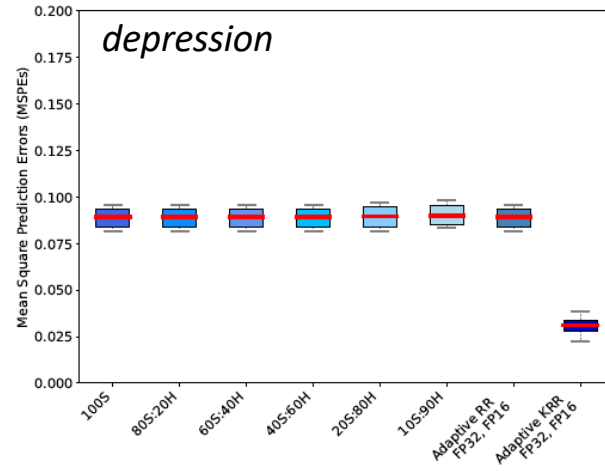
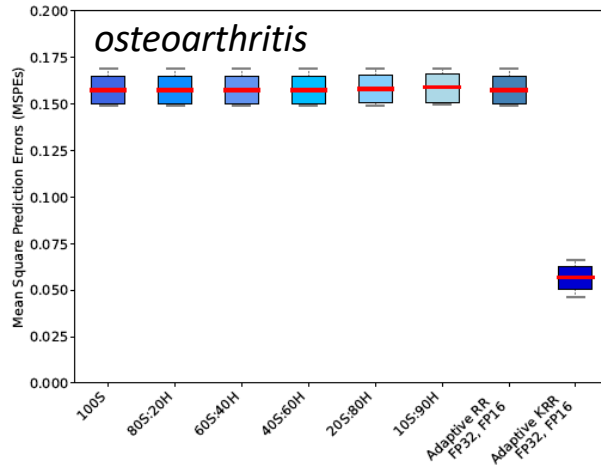
<b>Phenotypes</b>	<b>RR</b>	<b>KRR</b>
Hypertension	0.2983	0.8071
Asthma	0.2517	0.8205
Allergic Rhinitis	0.2008	0.8652
Osteoarthritis	0.3189	0.8386
Depression	0.2041	0.8454

epistatic



# Motivation for epistatic genome association studies

- Train statistical model on genotype/environmental to phenotype data
- Use to predict disease and other genetic/environmental characteristics
- Ridge regression is a linear association that considers individual SNPs
- Kernel ridge regression is correlates instances of multiple SNPs



# Kernel Ridge Regression is just linear algebra

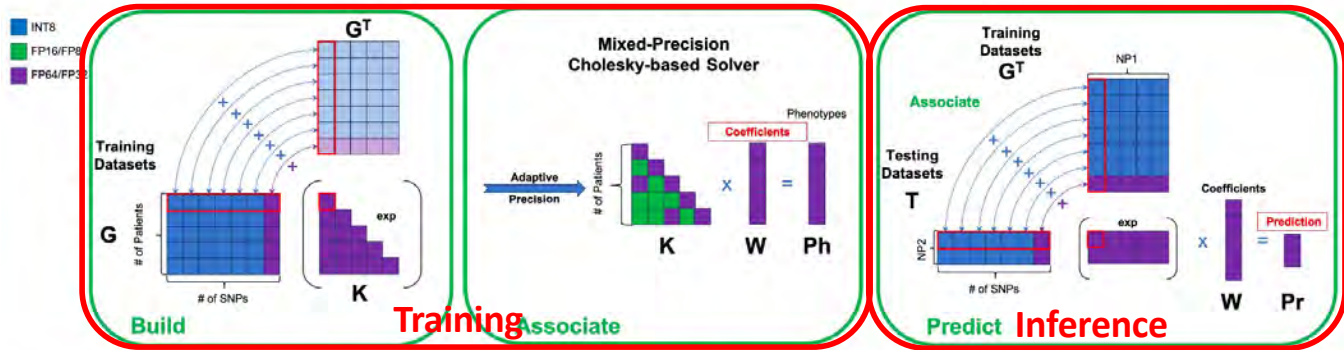


Fig. 3: Leveraging the INT8 / FP8 / FP16 / FP32 / FP64 multivariate KRR-based GWAS for genetic epistasis.

**Algorithm 2:** Build the KRR matrix.

- 1: **Procedure** BUILD( $\gamma, G_1, G_2, K$ )
- 2:  $N_{P1} \leftarrow \text{rowsize}(G_1)$
- 3:  $N_{P2} \leftarrow \text{rowsize}(G_2)$
- 4:  $K \leftarrow \text{zeros}(N_{P1}, N_{P2})$
- 5: **for**  $i$  in range(1,  $N_{P1}$ ) **do**
- 6:   **for**  $j$  in range(1,  $N_{P2}$ ) **do**
- 7:      $K[i, j] \leftarrow \text{KERNELMATRIX}(\text{type}, \gamma, G_1[i, :], G_2[j, :])$
- 8:   **end for**
- 9: **end for**

**Algorithm 5:** Kernel Matrix Definitions.

- 1: **Function** KERNELMATRIX( $\text{type}, \gamma, p_1, p_2$ )
- 2:  $N_S \leftarrow \text{size}(p_1)$
- 3: **if**  $\text{type} == \text{'Gaussian'}$  **then**
- 4:   **return**  $e^{-\gamma \cdot \|p_1 - p_2\|^2}$
- 5: **else if**  $\text{type} == \text{'IBS'}$  **then**
- 6:   **return**  $\frac{p_1 \sim p_2}{N_S}$
- 7: **end if**

# Tensor-core intensive “Build” phase in more detail

Let  $G$  contain three patients,  $a$ ,  $b$ , and  $c$ , with two markers

$$G = \begin{pmatrix} a_1 & b_1 & c_1 \\ a_2 & b_2 & c_2 \end{pmatrix}$$

We want the squared distance matrix

$$\tilde{D} = \begin{pmatrix} 0 & \|b - a\|^2 & \|c - a\|^2 \\ \|a - b\|^2 & 0 & \|c - b\|^2 \\ \|a - c\|^2 & \|b - c\|^2 & 0 \end{pmatrix}$$

where  $a^2 = a_1^2 + a_2^2$ , etc., to exponentiate to get the kernel matrix,  $K$

Initialize  $D$  to the square of the length of each patient vector down the columns ( $O(N)$  data)

$$D = \begin{pmatrix} a_1^2 + a_2^2 & b_1^2 + b_2^2 & c_1^2 + c_2^2 \\ a_1^2 + a_2^2 & b_1^2 + b_2^2 & c_1^2 + c_2^2 \\ a_1^2 + a_2^2 & b_1^2 + b_2^2 & c_1^2 + c_2^2 \end{pmatrix}$$

Form the symmetric

$$\tilde{D} \leftarrow D + D^T$$

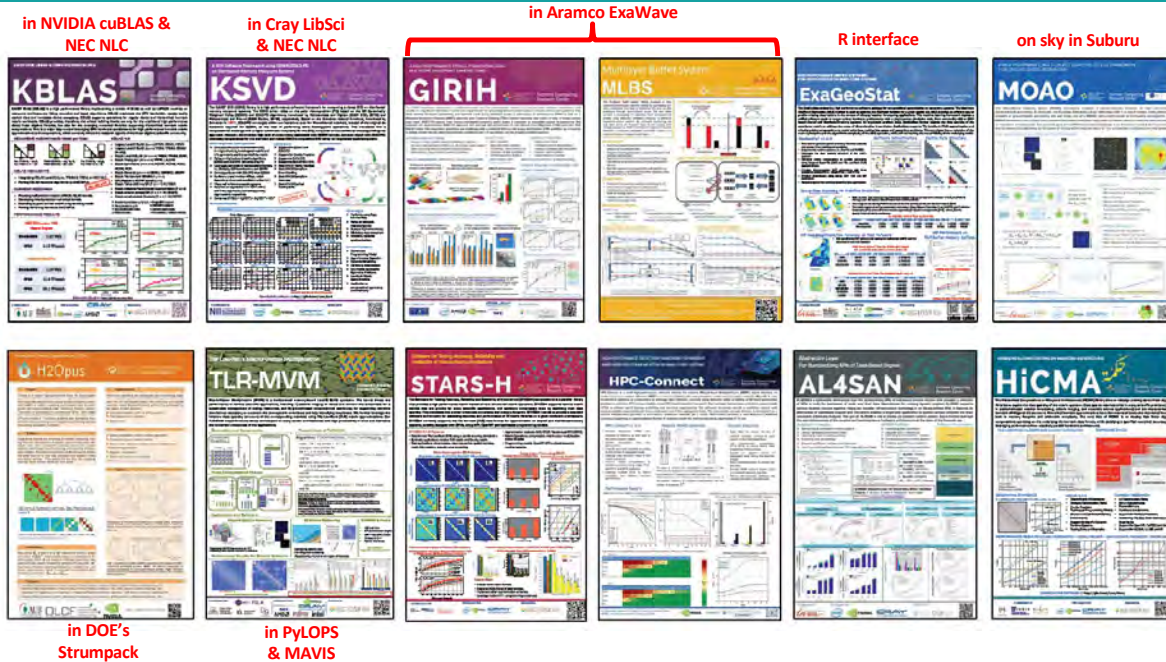
Then, apply BLAS3 in INT8 to get

$$\tilde{D} \leftarrow \tilde{D} - 2 * GG^T$$

using

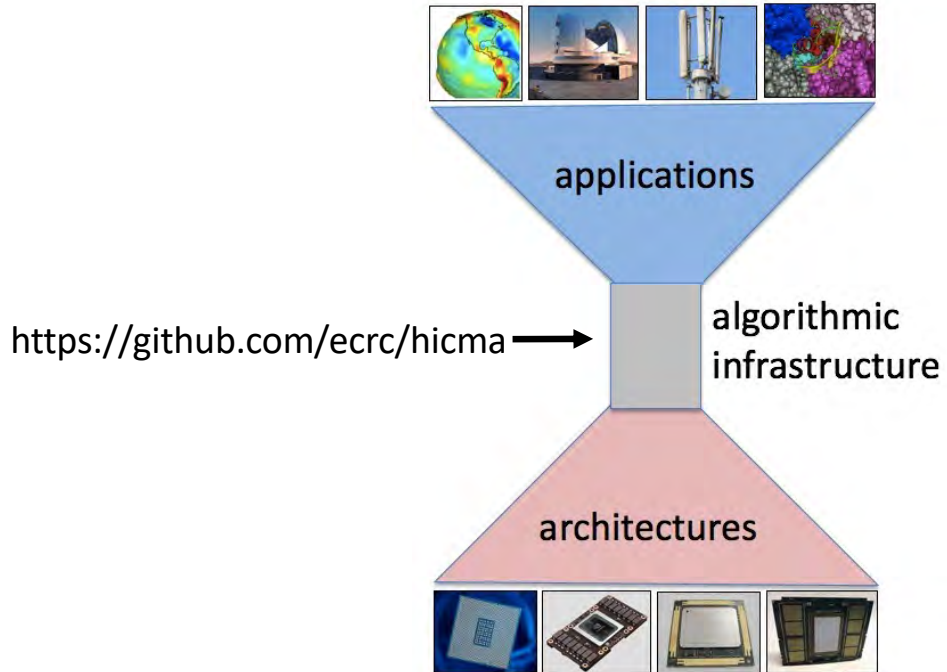
$$a^2 - 2ab + b^2 = \|a - b\|^2$$

# Some home-grown software



Updated annually for SC'xy , at <https://github.com/ecrc>

# Hourglass model of software



# Conclusions, recapped

As computational infrastructure demands a growing sector of research budgets and global energy expenditure, we must address the need for greater efficiency

As a community, we have excelled at this historically in three aspects:

- architectures
- applications (redefining *actual outputs of interest*)
- algorithms

There are *new algorithmic* opportunities in:

- reduced rank representations
- reduced precision representations

# Sustainable computing – two meanings

12 RESPONSIBLE CONSUMPTION AND PRODUCTION



## Computing sustainably

- or at least efficiently – not computing more than necessary for a given scientific target

7 AFFORDABLE AND CLEAN ENERGY



## Computing to *support* sustainability

- renewable energy
- affordable energy



# For follow-up

- 1) *Parallel Approximation of the Maximum Likelihood Estimation for the Prediction of Large-Scale Geostatistics Simulations*, S. Abdulah, H. Ltaief, Y. Sun, M. G. Genton & D. Keyes, 2018, IEEE International Conference on Cluster Computing (CLUSTER), 2018, pp. 98-108, doi: 10.1109/CLUSTER.2018.00089.
- 2) *Hierarchical Algorithms on Hierarchical Architectures*, D. Keyes, H. Ltaief & G. Turkiyyah, 2020, Philosophical Transactions of the Royal Society, Series A 378:20190055, doi 10.1098/rsta.2019.0055
- 3) *Responsibly Reckless Matrix Algorithms for HPC Scientific Applications*, H. Ltaief, M. G. Genton, D. Gratadour, D. Keyes & M. Ravasi, 2022, Computing in Science and Engineering, doi 10.1109/MCSE.2022.3215477.
- 4) *Reshaping Geostatistical Modeling and Prediction for Extreme-Scale Environmental Applications*, Q. Cao, S. Abdulah, R. Alomairy, Y. Pei, P. Nag, G. Bosilca, J. Dongarra, M. G. Genton, D. E. Keyes, H. Ltaief & Y. Sun, 2022, in proceedings of the International Conference for High Performance Computing, Networking, Storage, and Analysis (SC'22), IEEE Computer Society (ACM Gordon Bell Finalist), doi 10.1109/SC41404.2022.00007.
- 5) *Mixed Precision Algorithms in Numerical Linear Algebra*, 2022, N. J. Higham & T. Mary, Acta Numerica, pp. 347–414, doi:10.1017/S0962492922000022.
- 6) *Scaling the “Memory Wall” for Multi-Dimensional Seismic Processing with Algebraic Compression on Cerebras CS-2 Systems*, H. Ltaief, Y. Hong, L. Wilson, M. Jacquelin, M. Ravasi, & David Keyes, 2023, , in proceedings of the International Conference for High Performance Computing, Networking, Storage, and Analysis (SC'22), IEEE Computer Society (ACM Gordon Bell Finalist), doi 10.1145/3581784.362704.



Thank you

شكرا

جامعة الملك عبد الله  
للعلوم والتقنية

King Abdullah University of  
Science and Technology

