DistPCA

Tera-Scale Genomic PCA via Out-of-Core Distributed Parallelism

Eugenia-Maria Kontopoulou

Department of Computer Engineering & Informatics
University of Patras

Linear Algebra Day
in honor of
Professor Emeritus Ioannis Maroulas

Athens, 16/06/2026

Thanks to my collaborators @ CEID

Georgios Mermigkis
PhD student

Argiris Sofotasios
PhD student

Efstratios Gallopoulos
Professor Emeritus

Panagiotis Hadjidoukas
Associate Professor

Special thanks to my PhD Advisor @ Purdue

Petros Drineas
Professor

Reprinted from *Communications in Pure and Applied Mathematics*, Vol. 13, No. I (February 1960). New York: John Wiley & Sons, Inc.

THE UNREASONABLE EFFECTIVENSS OF MATHEMATICS IN THE NATURAL SCIENCES

Eugene Wigner

Mathematics, rightly viewed, possesses not only truth, but supreme beauty cold and austere, like that of sculpture, without appeal to any part of our weaker nature, without the gorgeous trappings of painting or music, yet sublimely pure, and capable of a stern perfection such as only the greatest art can show. The true spirit of delight, the exaltation, the sense of being more than Man, which is the touchstone of the highest excellence, is to be found in mathematics as surely as in poetry.

- BERTRAND RUSSELL, Study of Mathematics

THE UNREASONABLE EFFECTIVENSS OF MATHEMATICS IN THE NATURAL SCIENCES

Eugene Wigner

Mathematics, rightly viewed, possesses not only truth, but supreme beauty cold and austere, like that of sculpture, without appeal to any part of our weaker nature, without the gorgeous trappings of painting or music, yet sublimely pure, and capable of a stern perfection such as only the greatest art can show. The true spirit of delight, the exaltation, the sense of being more than Man, which is the touchstone of the highest excellence, is to be found in mathematics as surely as in poetry.

- BERTRAND RUSSELL, Study of Mathematics

THE MAJOR PROBLEMS IN

MATRIX COMPUTATIONS

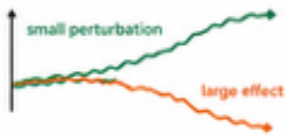
Challenges at the heart of theory and practice

1. HIGH COMPUTATIONAL COST

Matrix operations scale cubically ($O(n^3)$) or worse. Cost grows rapidly with size.

$\text{Cost} \sim O(n^3)$

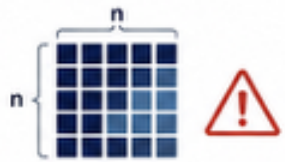
2. NUMERICAL INSTABILITY



Rounding errors and ill-conditioning can lead to large inaccuracies.

$\text{cond}(A) \gg 1$

3. MEMORY LIMITATIONS



Storing large dense matrices requires $O(n^2)$ memory. Becomes prohibitive quickly.

$\text{Memory} \sim O(n^2)$

4. SPARSITY EXPLOITATION

Ignoring sparsity wastes time and memory. Efficient use is nontrivial.

$\text{nnz} \ll n^2$

5. PARALLELIZATION AND SCALABILITY

Achieving good scalability on modern architectures is complex (communication, synchronization, load balance).

6. DATA MOVEMENT AND BANDWIDTH

Moving data is often more expensive than computing. Bandwidth is a major bottleneck.

$\text{Time}_{move} \gg \text{Time}_{flop}$

7. I/O AND OUT-OF-CORE COMPUTING

When matrices don't fit in RAM, I/O becomes the bottleneck. Out-of-core methods are hard.

I/O cost dominates

8. ALGORITHM SELECTION

Many algorithms exist. Choosing the right one depends on structure, accuracy, and resources.

No one-size-fits-all

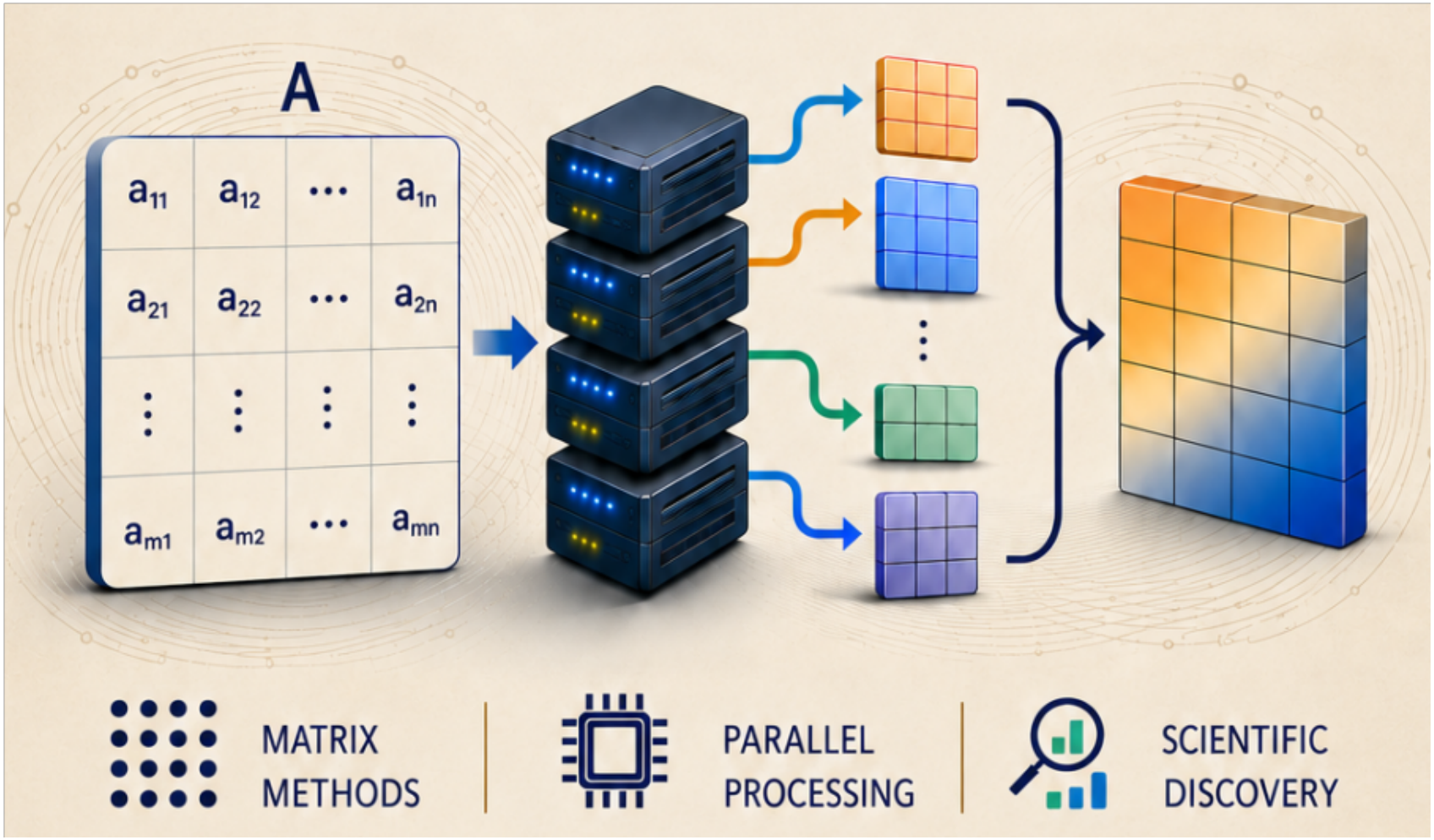
9. STRUCTURED AND SPECIAL MATRICES

Exploiting structure (symmetric, sparse, low-rank, Toeplitz, etc.) is difficult but essential for performance.

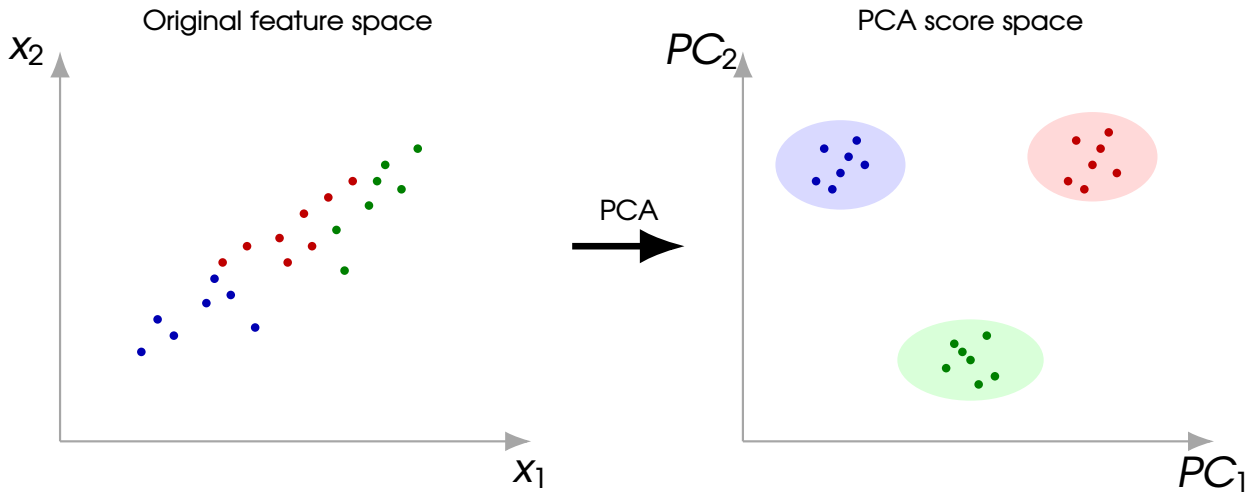
10. ACCURACY VS. PERFORMANCE TRADE-OFF

Higher accuracy often costs more time and memory. Finding the right balance is problem-dependent.

Matrix computations are at the core of science and engineering.
Overcoming these challenges requires mathematical insight, algorithmic innovation, and hardware-aware implementation.

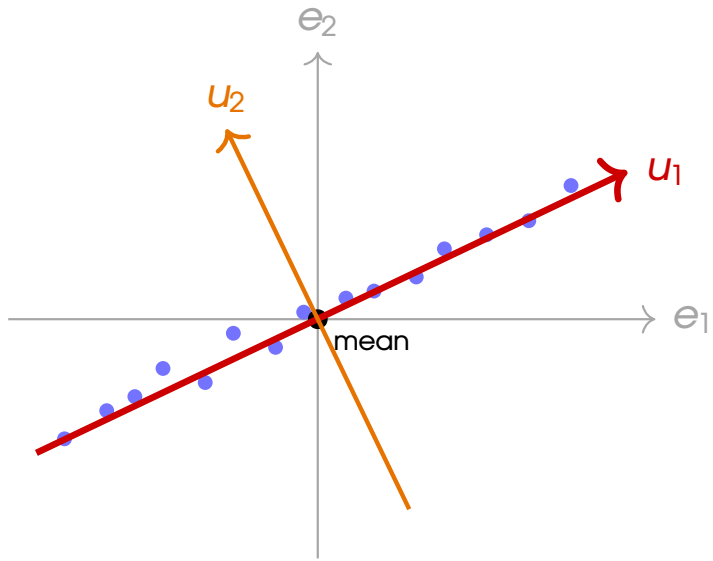

A
a_{11} a_{12} ⋯ a_{1n}
a_{21} a_{22} ⋯ a_{2n}
⋮ ⋮ ⋮ ⋮
a_{m1} a_{m2} ⋯ a_{mn}
MATRIX METHODS
PARALLEL PROCESSING
SCIENTIFIC DISCOVERY

Principal Component Analysis (PCA) is an *unsupervised learning method* used to reveal low-dimensional structure in high-dimensional data.



Key idea

Replace correlated features by new uncorrelated ones, the **principal components**, ordered by explained variance.



Geometrically, PCA constructs a *new orthonormal coordinate system*, through *rotation*, such that:

- the **first principal component** aligns with the direction of maximal variance;
- the **second principal component** aligns with the direction of maximal remaining variance, subject to being orthogonal to the first;
- each **subsequent principal component** aligns with the next largest remaining variance, subject to being orthogonal to all previous components.

Let $\mathbf{X} \in \mathbb{R}^{n \times m}$ be a matrix of n variables and m samples. We define the sample covariance matrix:

$$\mathbf{C} := \frac{1}{n-1} \mathbf{X}_c^\top \mathbf{X}_c,$$

where:

$$\mathbf{X}_c = \mathbf{X} \left(\mathbf{I}_m - \frac{1}{m} \mathbf{1}_m \mathbf{1}_m^\top \right)$$

is the matrix of mean-centered samples.

For the SPSD $\mathbf{C}$ it holds that for any unit vector $\mathbf{w} \in \mathbb{R}^m$, the Rayleigh quotient:

$$\mathbf{w}^\top \mathbf{C} \mathbf{w} = \frac{1}{n-1} \mathbf{w}^\top \mathbf{X}_c^\top \mathbf{X}_c \mathbf{w} = \frac{1}{n-1} \|\mathbf{X}_c \mathbf{w}\|_2^2$$

is equivalent to the *sample variance* of the projected, on $\mathrm{span}(\mathbf{w})$, mean-centered samples.

Since the first principal component maximizes the sample variance, it is the solution to the problem:

$$\arg \max_{\|\mathbf{w}\|_2=1} \mathbf{w}^\top \mathbf{C}\mathbf{w}. \tag{1}$$

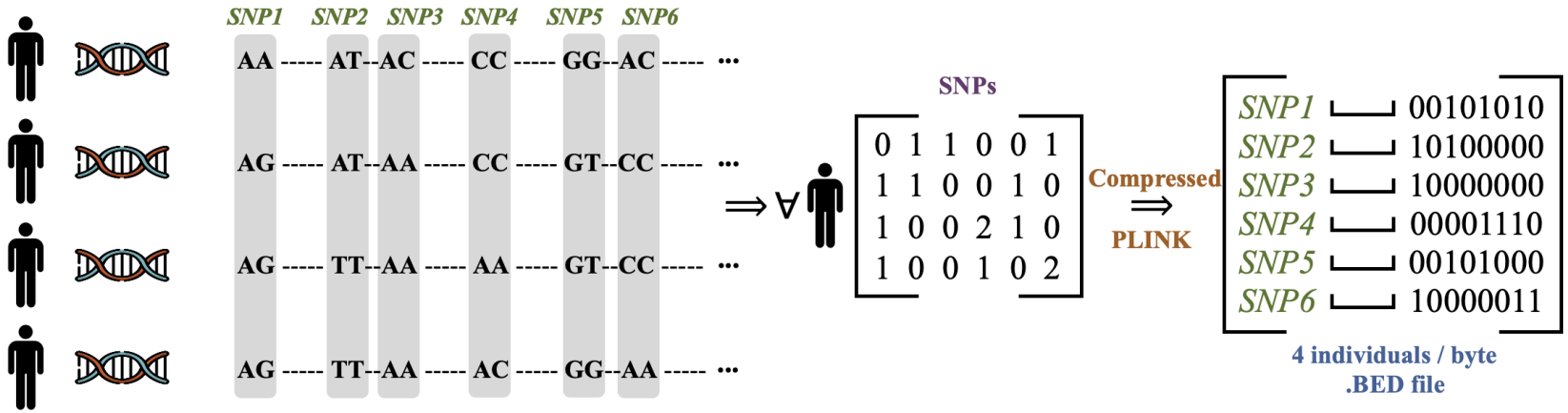
By the Rayleigh-Ritz theorem the solution to Problem (1) is $\mathbf{v}_1$, the eigenvector of $\mathbf{C}$ that corresponds to its dominant eigenvalue. Equivalently, $\mathbf{v}_1$ is the first right singular vector of $\mathbf{X}_c$.

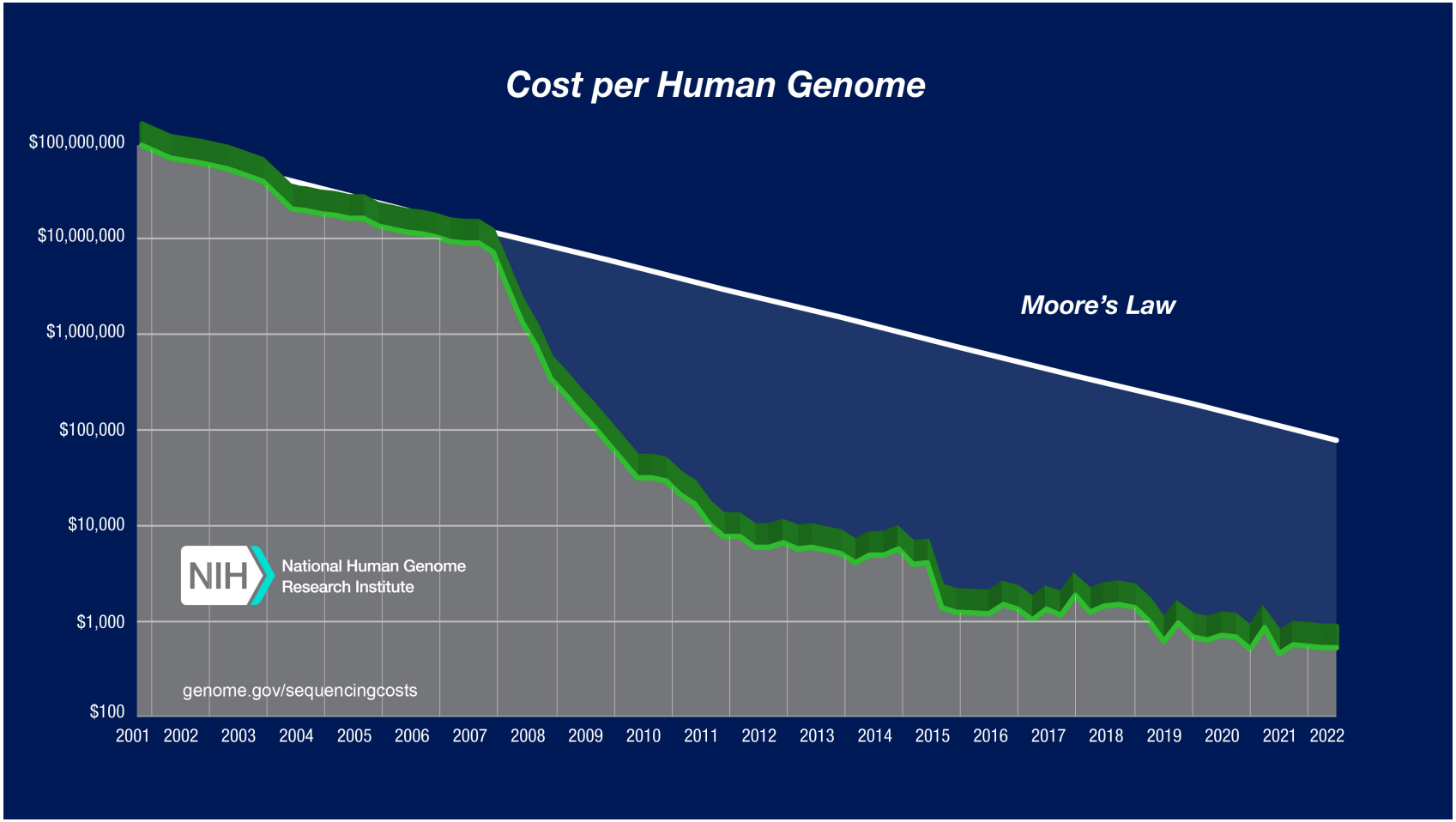
- **The first principal component** is $\mathbf{v}_1$.
- **Subsequent principal components** are obtained through deflation and correspond to the remaining eigenvectors of $\mathbf{C}$.

Human Genome

1. approximately 3.2B base pairs
2. we sequence specific variants across the genome, the Single Nucleotide Polymorphisms (SNPs)
3. approximately 4-5M SNPs per person
4. over 1.2B SNPs have been cataloged, but only a fraction are common in population.

Dataset Construction





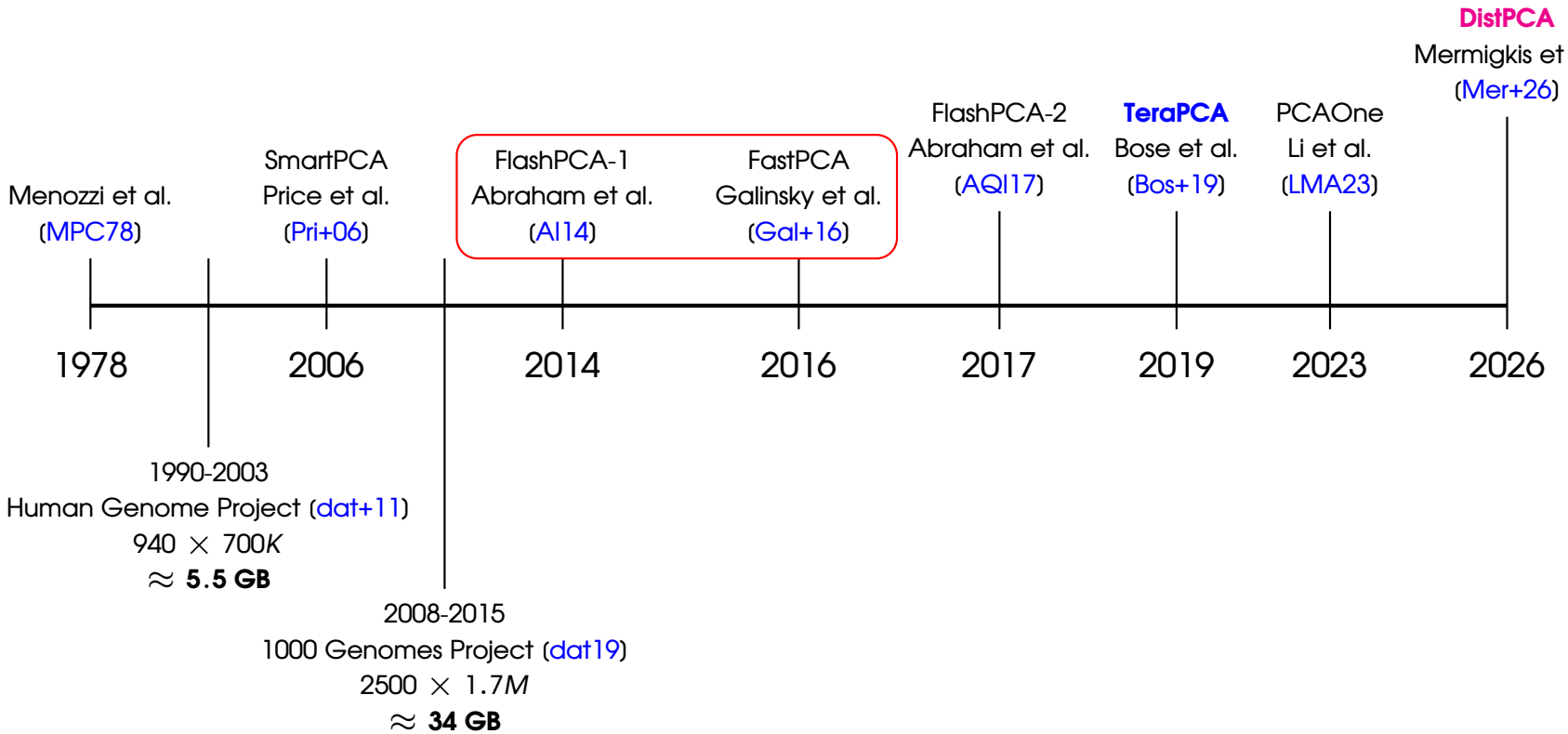
The cheaper it gets to sequence the genome the bigger the datasets become.

Practical Considerations

- PCA is used for population stratification, visualization, quality control, and data compression.
- In practice, only a small number of principal components (typically 10-20) are needed.
- High precision is unnecessary; 3-5 digits of accuracy are usually sufficient.
- **Disclaimer:** We do not include PCA methods that are based on probabilistic models like ProPCA (Agr+20) or EMU (Mei+21).

PCA in Human Genetics

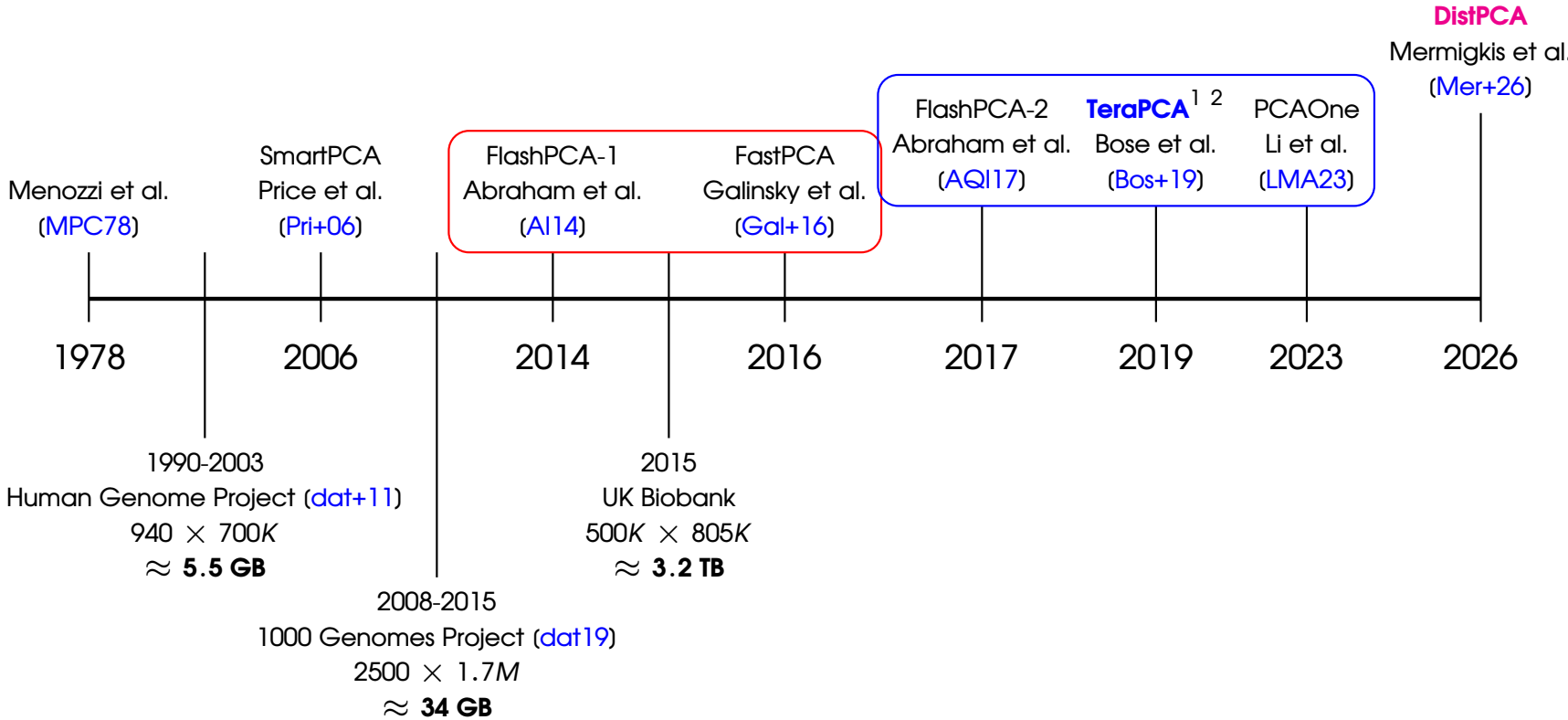
a historical overview



- **in-core methods**: datasets are considered to be in memory

PCA in Human Genetics

a historical overview



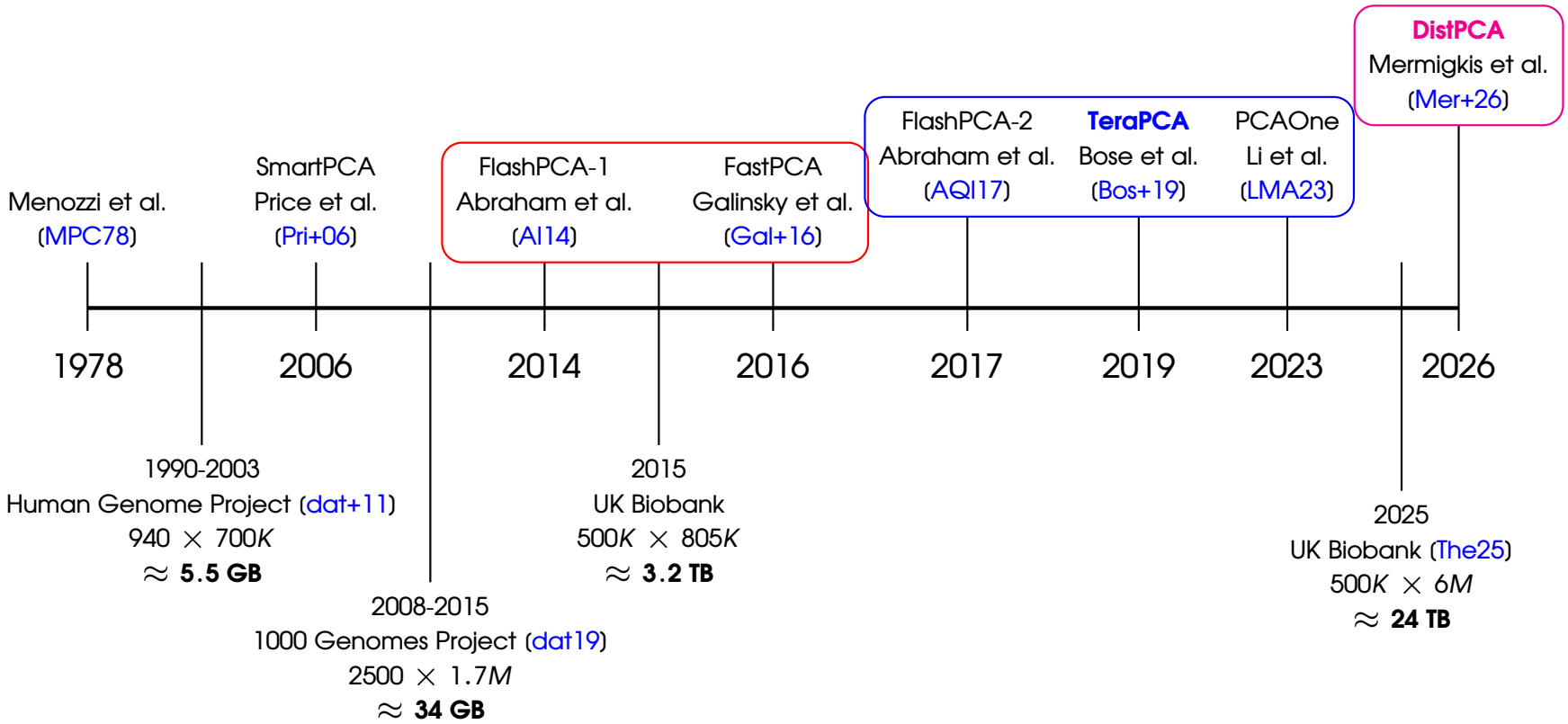
- **in-core methods**: datasets are considered to be in memory
- **single node out-of-core methods**: datasets are in storage, methods operate in blocks of data, increased overhead from I/O.

[1] A. Bose, V. Kalantzis, **E.K.**, M. Elkady, P. Paschou, P. Drineas (2019), *TeraPCA: a fast and scalable software package to study genetic variation in tera-scale genotypes* in Bioinformatics

[2] P. Drineas, I. Ipsen, **E.K**, M. Magdon-Ismail (2018), *Structural Convergence Results for Approximation of Dominant Subspaces from Block Krylov Spaces*, in SIMAX

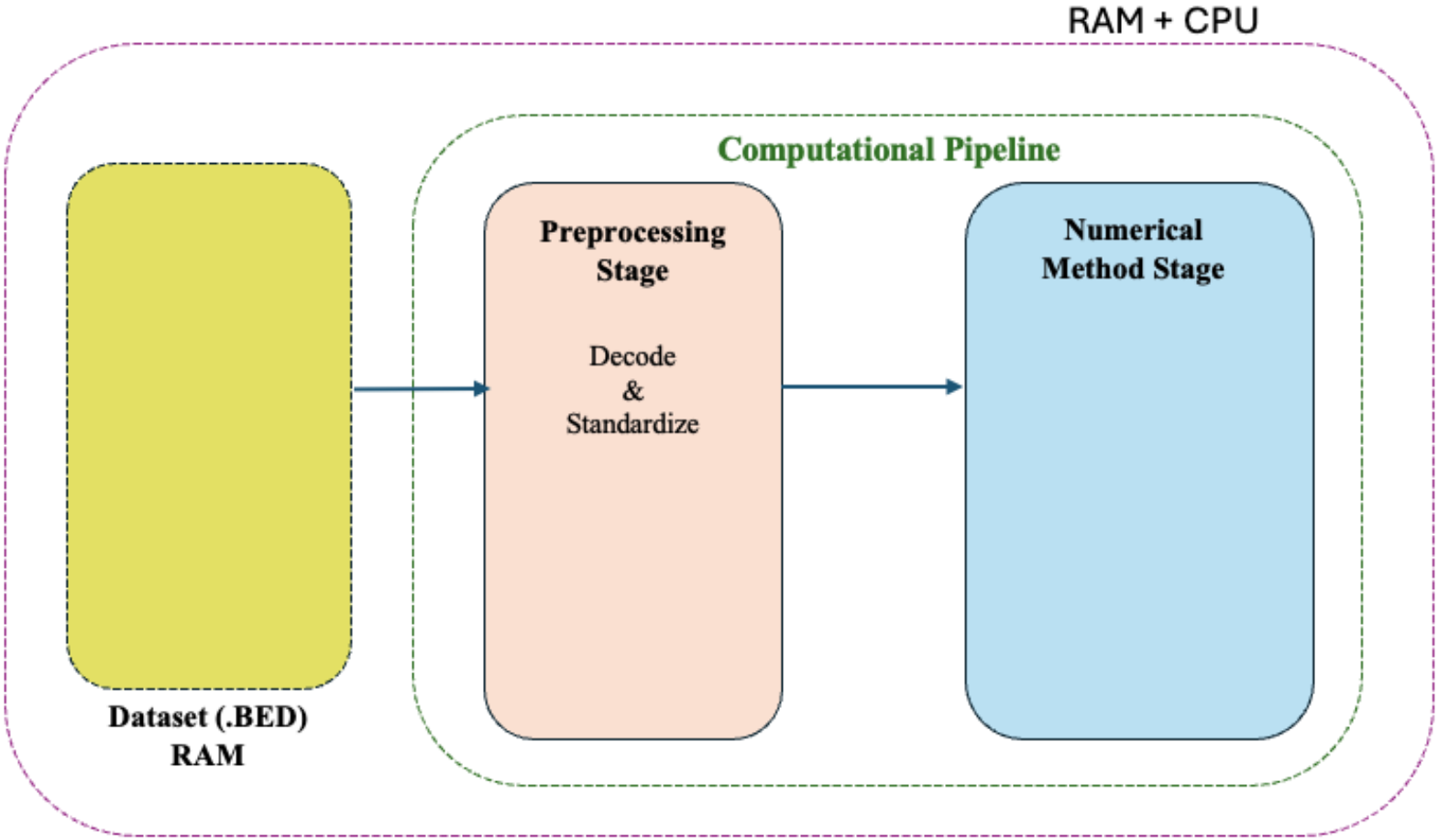
PCA in Human Genetics

a historical overview



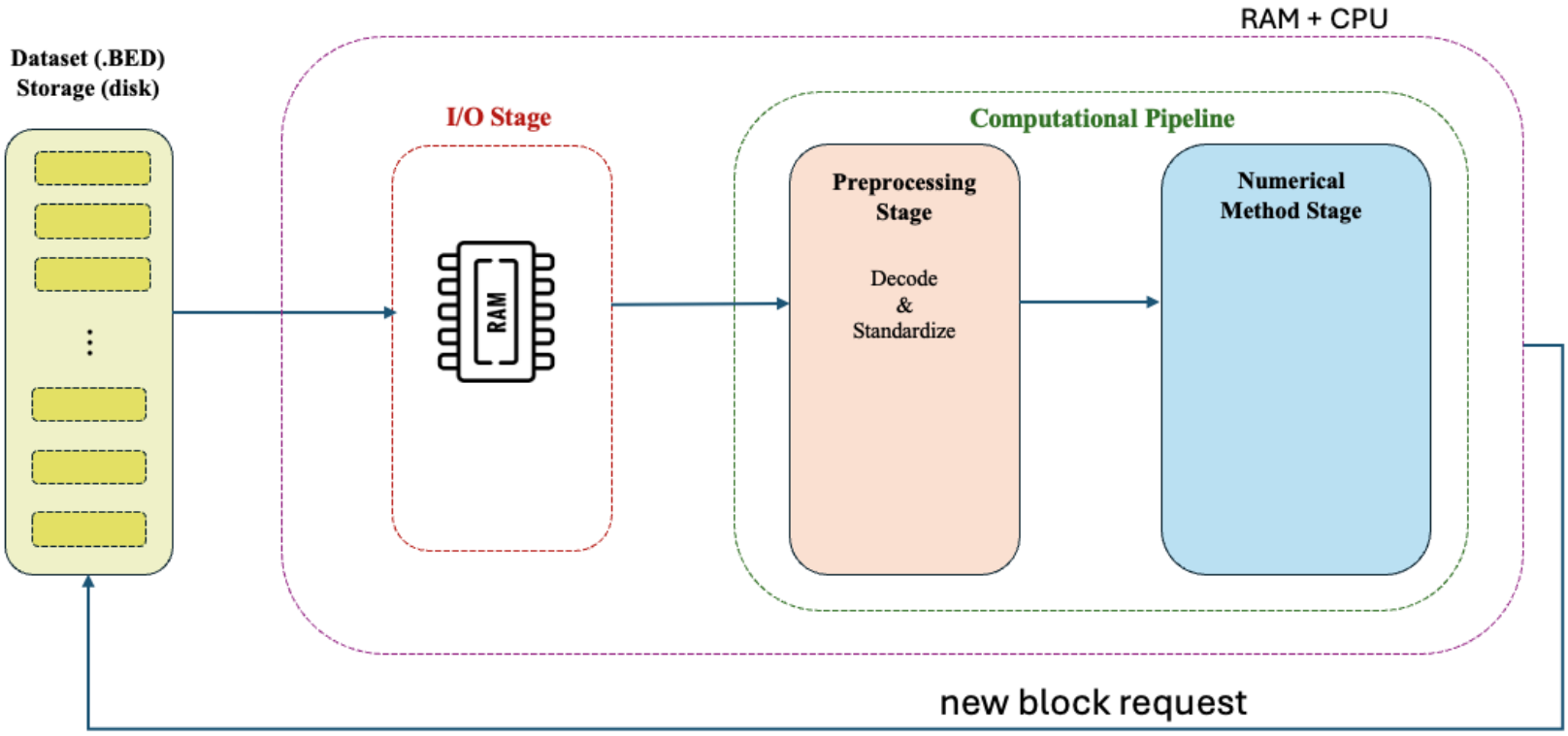
- **in-core methods**: datasets are considered to be in memory
- **single node out-of-core methods**: datasets are in storage, operation in blocks, increased overhead from I/O.
- **multi-node out-of-core method**: datasets are in storage, operation in blocks, I/O and computations are done through a distributed parallelism scheme.

in-core pipeline

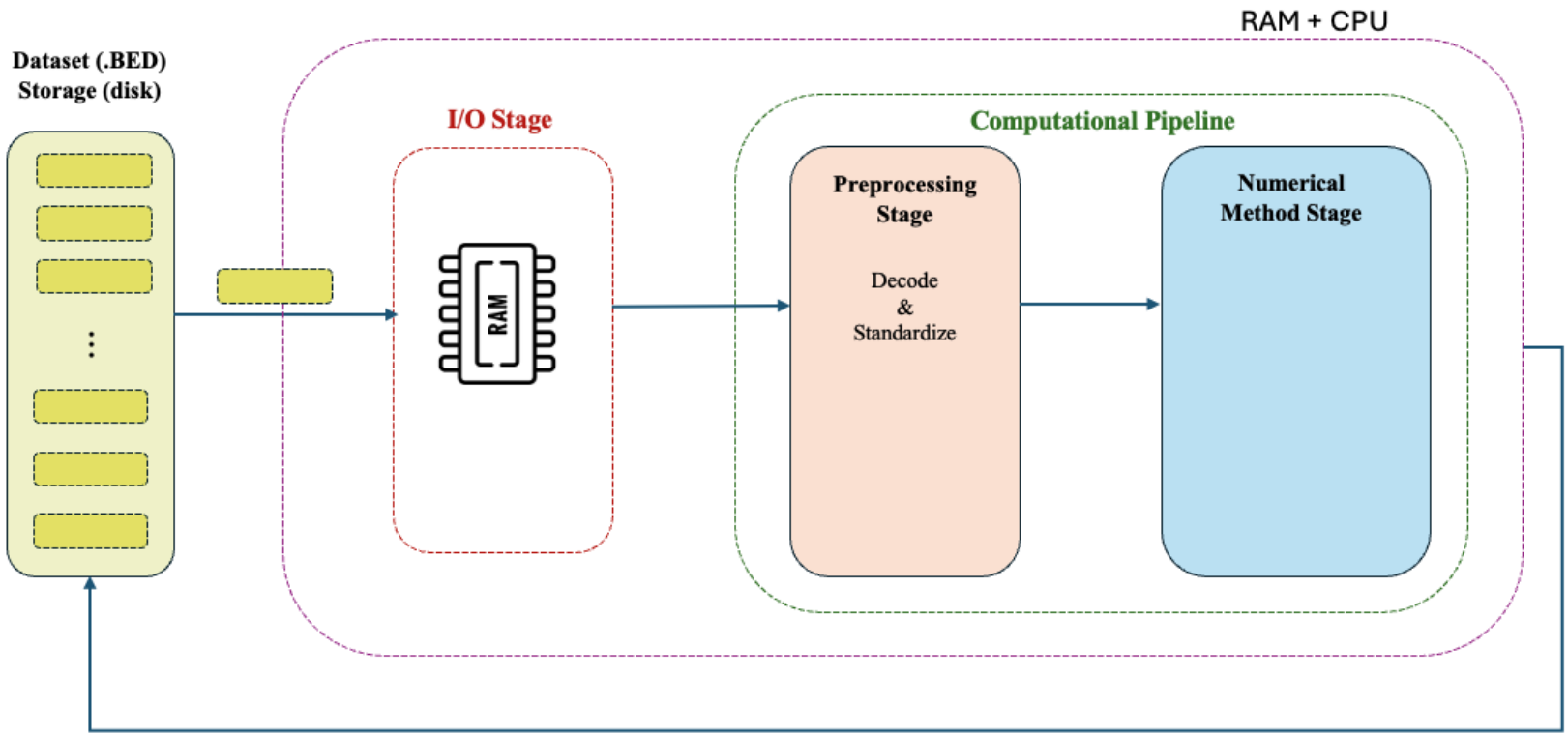


Main bottleneck: the computational pipeline, i.e. number of FLOPS

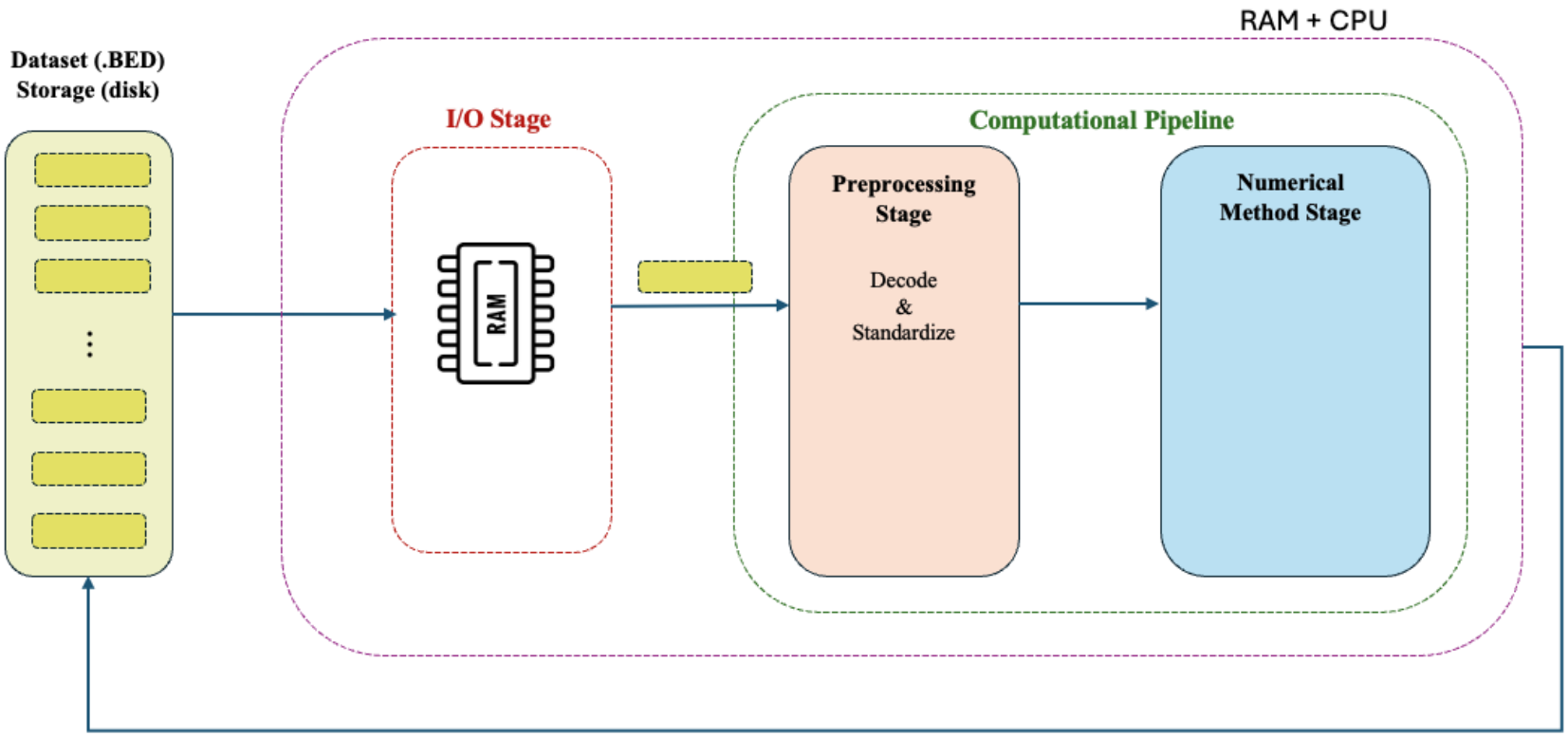
out-of-core pipeline



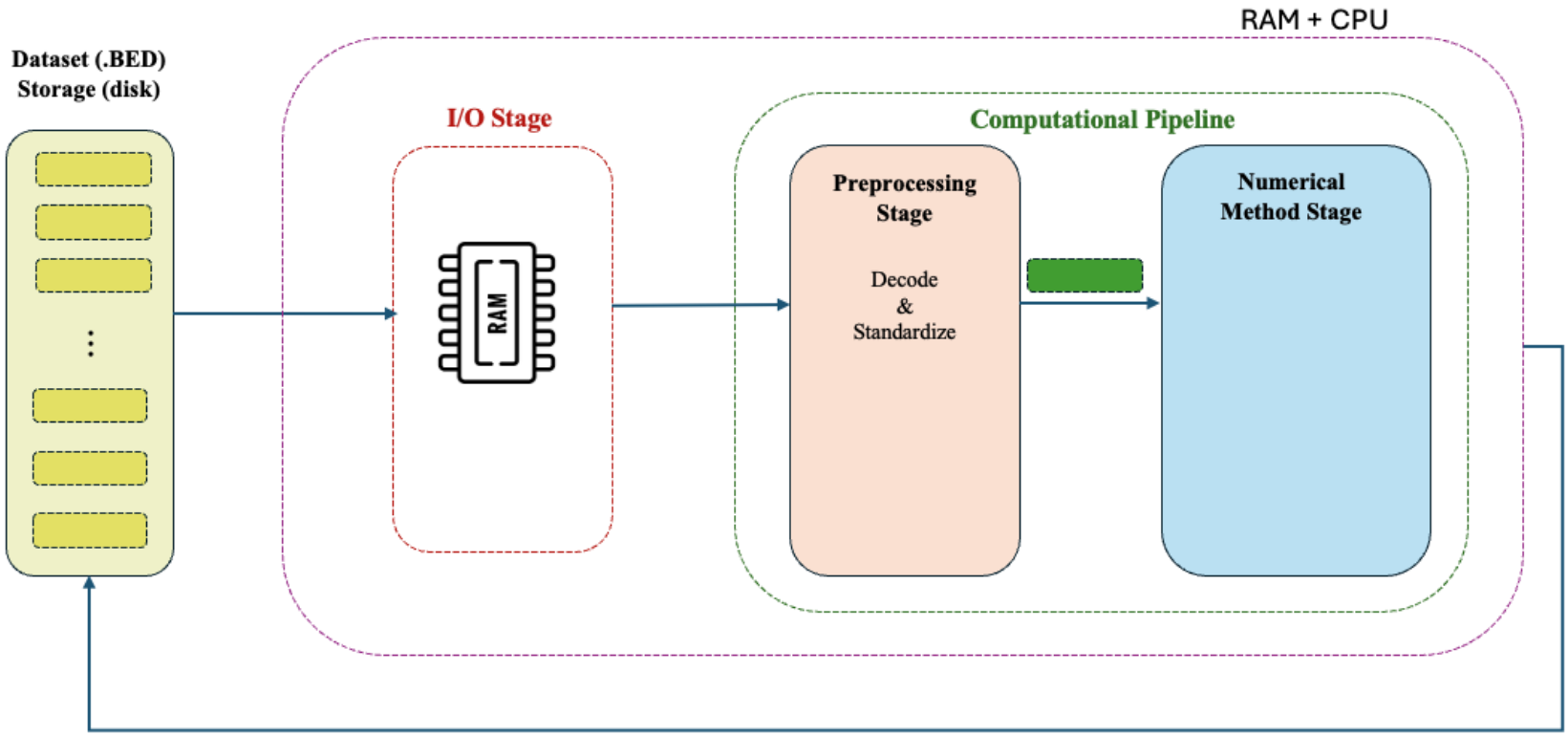
out-of-core pipeline



out-of-core pipeline



out-of-core pipeline



Main bottlenecks:

1. the computational pipeline, i.e. number of FLOPS
2. the communication between storage and RAM for the block fetches, i.e. number of passes over the data

As data moves farther from the CPU, communication becomes increasingly expensive.

Observed problems

Previous genomic PCA implementations:

1. focused on optimizing the numerical method stage
2. remained in single-node infrastructure, upper-bounding their scalability

As datasets grow, I/O dominates execution time resulting in increased CPU idle time.

Numerical method stage

The numerical method **remained fast**, mostly because of highly optimized libraries like LAPACK/BLAS, Eigen, Spectra, ...

Scalability of PCA pipeline

The ability of the pipeline to handle the growing size of data **deteriorated**.

DistPCA accelerates the entire pipeline.

All mentioned out-of-core genomic PCA methods employ a **Rayleigh-Ritz procedure**.

Given an orthonormal basis $\mathbf{Q} \in \mathbb{R}^{m \times s}$ with $k \leq s \ll m$, the approximation of the dominant eigenspace is computed as follows:

1. Project $\mathbf{C}$ into the s-dimensional subspace with orthonormal basis $\mathbf{Q}$: $\mathbf{H} = \mathbf{Q}^\top \mathbf{C}\mathbf{Q}$.
2. Solve the (way smaller) eigenproblem for $\mathbf{H}$ to find its eigenpairs, $[(\lambda_i, \mathbf{v}_i)]_{i=1}^{s}$.
3. Project the eigenpairs back to the original space to find the Ritz pairs of $\mathbf{C}$: $[(\lambda_i, \mathbf{Q}\mathbf{v}_i)]_{i=1}^{s}$

Reduced subspace

Given random guesses $\mathbf{x}_0 \in \mathbb{R}^m$ and $\mathbf{X}_0 \in \mathbb{R}^{m \times s}$ the reduced subspace can be:

1. Krylov subspace: $\mathcal{K}_s(\mathbf{C}, \mathbf{x}_0)$, where $s = k$ e.g. FlashPCA2, PCAone
2. $\mathcal{S}_s(\mathbf{C}, \mathbf{X}_0) = \mathrm{range}(\mathbf{C}^\rho \mathbf{X}_0)$, where ρ depends on the desired accuracy e.g. TeraPCA, PCAone, DistPCA

Initial guess

All methods building subspace of the form (2) start from a random initial guess, typically generated from a normal distribution, that helps provide provable guarantees for the approximation through Random Matrix Theory. For this reason, they are often referred to as **randomized SVD.**

Input: $\mathbf{X}_c \in \mathbb{R}^{n \times m}$, initial guess matrix $\mathbf{X}_0 \in \mathbb{R}^{m \times s}$ with elements drawn i.i.d. from the normal distribution $\mathcal{N}(0, 1)$, $k \geq 1$, and $s \geq k$.

Output: The k leading approximate left singular vectors of $\mathbf{X}_c$.

$\mathbf{V} \leftarrow \mathbf{X}_0$
repeat
  $\mathbf{V} = \mathbf{X}_c^\top \mathbf{X}_c \mathbf{V}$
  for $i \leftarrow 1$ **to** $\rho - 1$ **do**
    $\mathbf{Q} = \text{orth}\,(\mathbf{V})$
    $\mathbf{V} = \mathbf{X}_c^\top \mathbf{X}_c \mathbf{Q}$
  end for
  $\mathbf{Q} = \text{orth}\,(\mathbf{V})$
  $\mathbf{V} = \mathbf{X}_c^\top \mathbf{X}_c \mathbf{Q}$
  $\mathbf{M} = \mathbf{Q}^\top \mathbf{V}$
  Compute the eigenvalue decomposition $\mathbf{M} = \hat{\mathbf{V}} \mathbf{D} \hat{\mathbf{V}}^\top$
  $\mathbf{V} = \mathbf{Q} \hat{\mathbf{V}}$
until convergence
return first k columns of $\mathbf{V}$

Input: $\mathbf{X}_c \in \mathbb{R}^{n \times m}$, initial guess matrix $\mathbf{X}_0 \in \mathbb{R}^{m \times s}$ with elements drawn i.i.d. from the normal distribution $\mathcal{N}(0, 1)$, $k \geq 1$, and $s \geq k$.

Output: The k leading approximate left singular vectors of $\mathbf{X}_c$.

$\mathbf{V} \leftarrow \mathbf{X}_0$
repeat
  $\mathbf{V} = \mathbf{X}_c^\top \mathbf{X}_c \mathbf{V}$
  $\mathbf{Q} = \text{orth}\,(\mathbf{V})$
  $\mathbf{V} = \mathbf{X}_c^\top \mathbf{X}_c \mathbf{Q}$
  $\mathbf{M} = \mathbf{Q}^\top \mathbf{V}$
  Compute the eigenvalue decomposition $\mathbf{M} = \hat{\mathbf{V}} \mathbf{D} \hat{\mathbf{V}}^\top$
  $\mathbf{V} = \mathbf{Q} \hat{\mathbf{V}}$
until convergence
return first k columns of $\mathbf{V}$

Computations in Steps 3 and 5 can be performed blockwise using the MMV multiplication algorithm, which exploits Level-3 BLAS operations for improved efficiency.

Input: number of blocks, $\zeta > 0$, $\mathbf{X} \in \mathbb{R}^{m \times s}$.
Output: $\mathbf{A} \in \mathbb{R}^{m \times s}$.

1: $\mathbf{A} \leftarrow \mathbf{0}_{m \times s}$
2: **for** $i \leftarrow 1 : \zeta$ **do**
3: Fetch the i-th row-block of $\mathbf{X}$
4: $\mathbf{A} = \mathbf{A} + \mathbf{X}_{(i)}{}^{\top}(\mathbf{X}_{(i)}\mathbf{X})$
5: **end for**
6: **return** $\mathbf{A}$

Recall in storage the genomic dataset is stored in a:

1. compressed and,
2. un-standardized, SNP-major format

1. Decode

- independent decoding
- Single Instruction Multiple Data (SIMD) vectorization

2. Standardize

- SNP-major data layout
- multithreading via OpenMP
- compute statistics (mean and std) the first time the block passes through the pipeline and store them

1. CPU idle time

To minimize CPU idle time, we employed a **double-buffering scheme** that overlaps I/O with computation:

- CPU processes current block
- a new block is fetched

2. I/O latency & computational throughput

To reduce the impact of I/O latency and increase computational throughput, we observed that we have to carefully **tune the block size**:

small: more I/O requests and processing overhead $\Rightarrow$ lower computational throughput

too large: exceeds the last-level cache $\Rightarrow$ more cache misses and data movement

Best choice $\Rightarrow$ a block size that fits in the last-level cache $\Rightarrow$ fewer cache misses and higher throughput

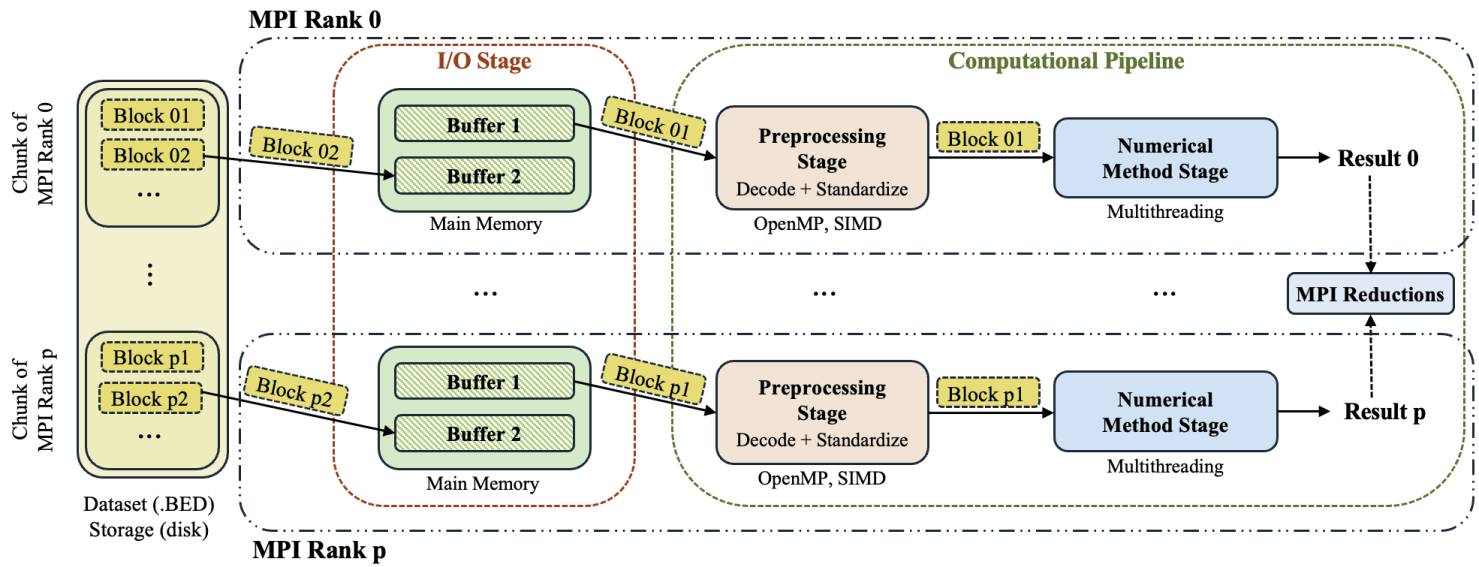
It is obvious this the block size parameter is system-dependent.

All the optimizations mentioned above would not have been sufficient without the key ingredient that brought them together and allowed the pipeline to reach its full potential:

parallelization of I/O using MPI

Distributed Parallelism with MPI

- multiple MPI ranks process *independent* sets of data blocks *concurrently*
- each rank produces a partial result, which is combined at the end using MPI collective operations



Datasets

Name	Individuals	SNPs	Type	Size(.BED file)
SGDP	345	694,659	Real	58 MB
HGDP	942	133,594	Real	31 MB
1000 Genomes	2,490	1,664,505	Real	989 MB
50K Genomes	50,000	6,000,000	Synthetic	75 GB
500K Genomes	500,000	3,000,000	Synthetic	350 GB
1M Genomes	1,000,000	1,000,000	Synthetic	233 GB

Synthetic Datasets were created with the Data Simulator.

Hardware

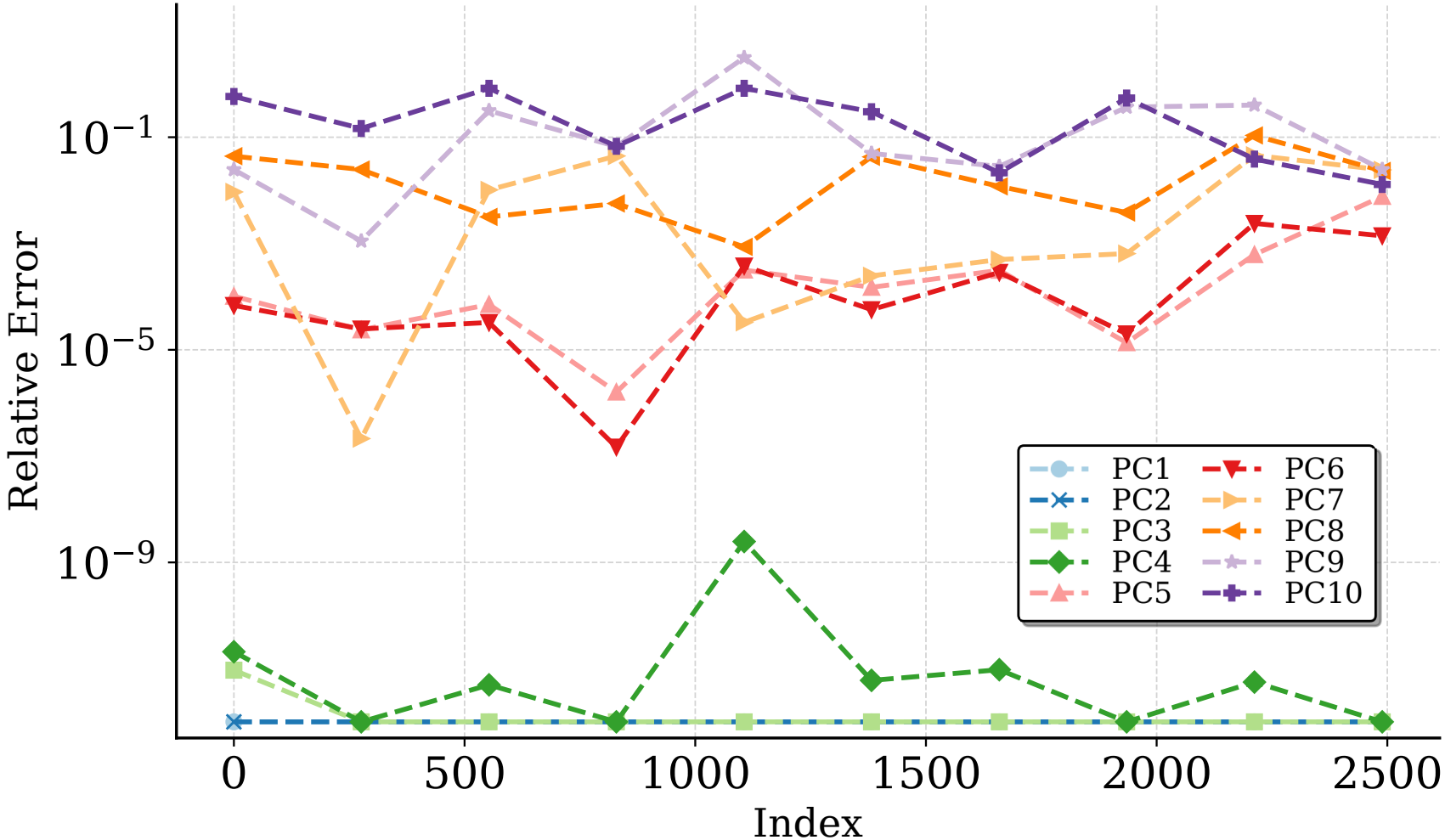
ARIS supercomputer - 4 compute nodes:

- 128 cores @2.25GHz (AMD EPYC 7742 CPU)
- 512GB RAM

General Parameters

- 64GB RAM
- $k = 20$ PCs (unless stated otherwise), $s = 2k$
- block size: 100 SNPs
- MEV: `1e-3`
- # threads: 8

We used the 1000 Genomes dataset, for which we were able to compute its exact SVD.

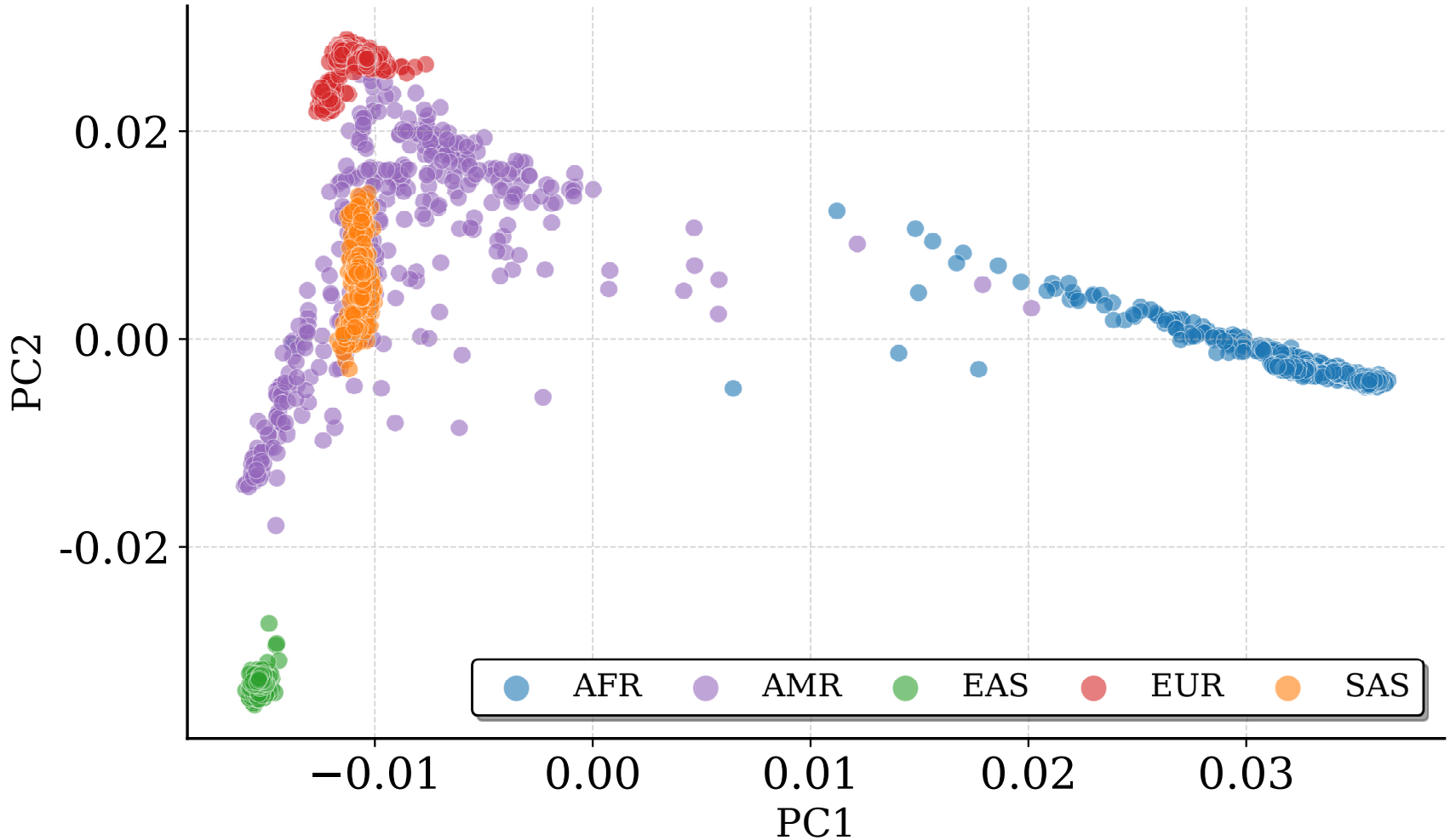


Entry wise relative error for the first 10 PCs with the subspace size, $s = 20$

Observations

- the first 4 PCs are approximated with high accuracy
- the first 4 eigenvalues are distinct:

λ_1	λ_2	λ_3	λ_4	λ_5
217.9280	98.0250	27.2434	20.7748	4.1516
λ_6	λ_7	λ_8	λ_9	λ_{10}
3.9312	3.7885	3.0153	1.9969	1.9024

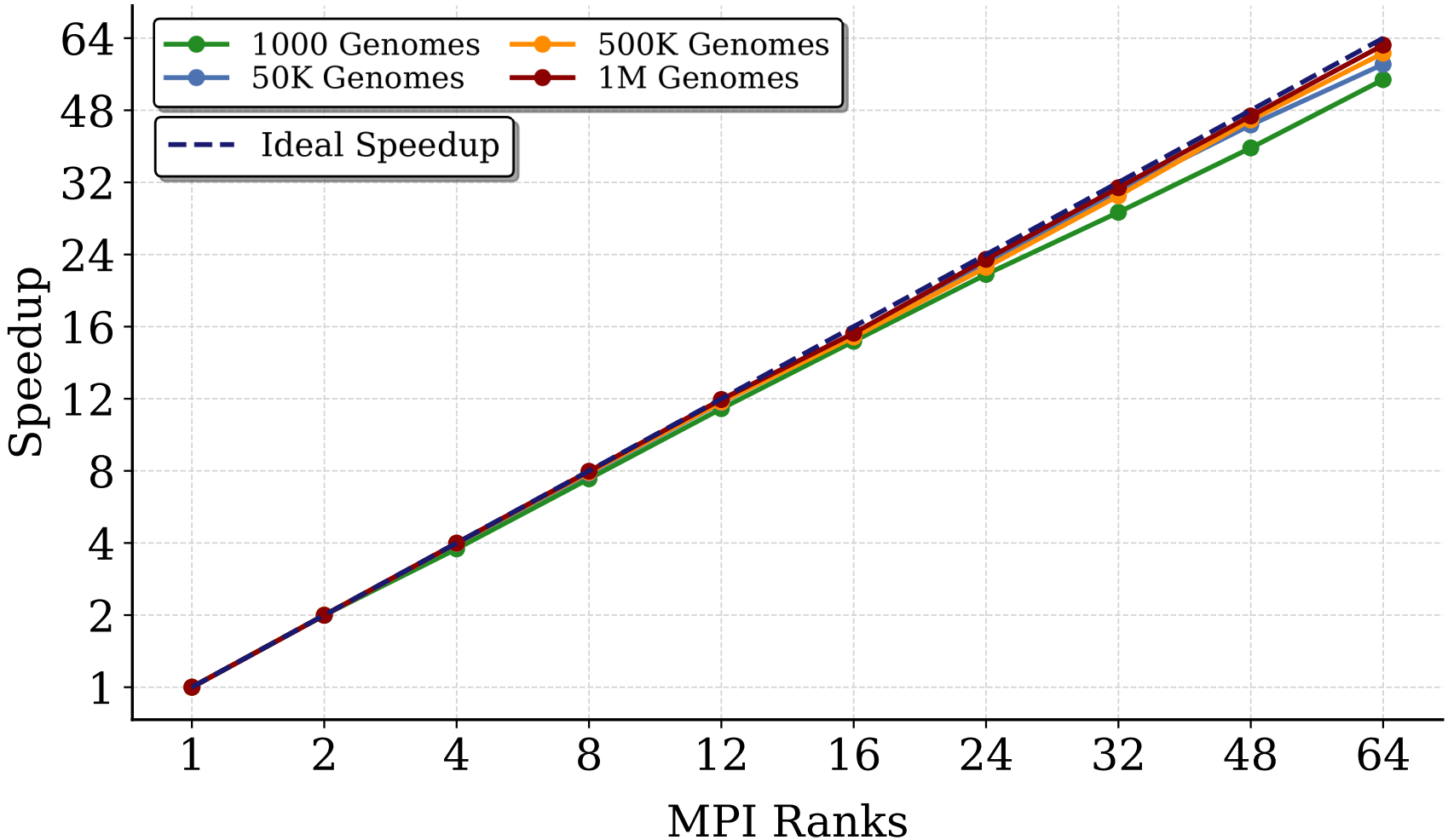


projection of individuals from 1000 Genomes dataset on PC1 and PC2.

Parameters

MPI ranks = 8 DistPCA OpenMP threads = 64

Dataset	PCAone	DistPCA	Speedup	Reduction %
1000 Genomes	173s	**47s**	3.68x	72.8%
50K Genomes	9.1h	**7.8h**	1.17x	14.1%
500K Genomes	12.1h	**2.3h**	5.26x	78.5%
1M Genomes	7.9h	**2.6h**	3.04x	67.9%



near linear speedup & strong scalability

We proposed a framework that:

- accelerates the entire genomics PCA pipeline through a 4-step optimization scheme (MPI I/O, OpenMP multithreading, SIMD, double buffering)
- extends the scalability of PCA in genomics
- can be incorporated with any numerical method that is based on associative operations

Future Work

- enable more numerical methods along with our framework
- provide a multi-GPU implementation of our framework
- explore whether we can create a multi-precision computational pipeline, enabling lower precision in some areas

What did we Learn?

- Numerical Linear Algebra plays a very important role in sciences, e.g. human genetics.
- Data movement matters in large-scale problems.
- High performance computing can unlock new potential for the application of Numerical Linear Algebra methods at large-scale problems .
- Relatively simple methods, like Randomized Subspace Iteration, can have unexpected applications in large-scale problems.

Questions?

(Agr+20) Aman Agrawal et al. ``Scalable probabilistic PCA for large-scale genetic variation data''. In: *PLOS Genetics* 16.5 (May 2020), pp. 1-19.

(AI14) Gad Abraham and Michael Inouye. ``Fast principal component analysis of large-scale genome-wide data''. In: *PLoS One* 9.4 (Apr. 2014), e93766.

(AQI17) Gad Abraham, Yixuan Qiu, and Michael Inouye. ``FlashPCA2: principal component analysis of Biobank-scale genotype datasets''. In: *Bioinf.* 33.17 (Sept. 2017), pp. 2776-8.

(Bos+19) Aritra Bose et al. ``TeraPCA: a fast and scalable software package to study genetic variation in tera-scale genotypes''. In: *Bioinf.* 35.19 (Oct. 2019), pp. 3679-83.

(dat+11) Yontao (dataset) Lu et al. ``Affymetrix Human Origins dataset (Supplement 10)''. In: *David Reich Lab* (2011). URL: https://reich.hms.harvard.edu/human-origins-dataset-supplement-10.

(dat19) Florian (dataset) Privé. ``1000 Genomes (Phase 3) files with SNPs in common with HapMap3 and UK Biobank''. In: *Figshare* (Aug. 2019). DOI: 10.6084/m9.figshare.9208979.v4.

(Gal+16) Kevin J. Galinsky et al. ``Fast Principal-Component Analysis Reveals Convergent Evolution of ADH1B in Europe and East Asia''. In: *Amer. J. Human Genetics* 98.3 (Mar. 2016), pp. 456-72.

(HMT11) N. Halko, P.-G. Martinsson, and J. A. Tropp. ``Finding Structure with Randomness: Probabilistic Algorithms for Constructing Approximate Matrix Decompositions''. In: *SIAM Rev.* 53.2 (2011), pp. 217-88.

(LMA23) Zilong Li, Jonas Meisner, and Anders Albrechtsen. ``Fast and accurate out-of-core PCA framework for large scale biobank data''. In: *Genome Res.* 33.9 (Sept. 2023), pp. 1599-608.

(Mei+21) Jonas Meisner et al. ``Large-scale inference of population structure in presence of missingness using PCA''. In: *Bioinformatics* 37.13 (July 2021), pp. 1868-1875.

(Mer+26) Georgios Mermigkis et al. ``DistPCA: Tera-Scale Genomic PCA via Out-of-Core Distributed Parallelism''. In: *bioRxiv* (2026). DOI: 10.64898/2026.05.15.725487. URL: https://www.biorxiv.org/content/10.64898/2026.05.15.725487v1.

(MPC78) P. Menozzi, A. Piazza, and L. Cavalli-Sforza. ``Synthetic maps of human gene frequencies in Europeans''. In: *Sci.* 201.4358 (Sept. 1978), pp. 786-92.

(Pri+06) A. L. Price et al. ``Principal components analysis corrects for stratification in genome-wide association studies''. In: *Nature Genetics* 38.8 (Aug. 2006), pp. 904-9.

(The25) The UK Biobank Whole-Genome Sequencing Consortium. ``Whole-genome sequencing of 490,640 UK Biobank participants''. In: *Nature* 645.8081 (Sept. 2025), pp. 692-701.