TeraPCA: A fast and scalable method to study genetic variation in tera-scale genotypes

Aritra Bose[1], Vassilis Kalantzis[2], Eugenia Kontopoulou[1], Mai Elkady[1], Peristera Paschou[3], Petros Drineas[1]

1-Department of Computer Science, Purdue University, West Lafayette, IN, USA
2-IBM Research, Thomas J. Watson Research Center, Yorktown Heights, NY, USA
3-Department of Biological Sciences, Purdue University, West Lafayette, IN, USA

Problem statement

- Principal Component Analysis (PCA) is a key tool in the study of population structure in human genetics.
- As modern datasets become increasingly larger in size, traditional approaches based on loading the entire dataset in the system memory (RAM) become impractical and out-of-core implementations are the only viable alternative.

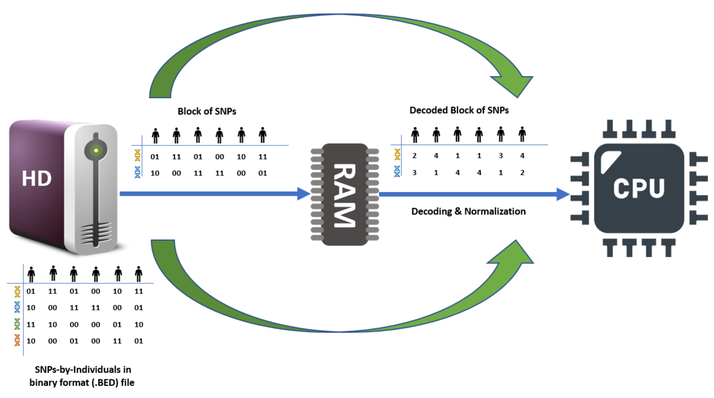
The TeraPCA library

Our contribution is summarized in TeraPCA, a C++ library to perform out-of-core PCA of large genetics datasets:

- TeraPCA computes the sought Principal Components by partially solving a symmetric eigenvalue problem.
- This eigenvalue problem is solved by Randomized Subspace Iteration.

Why it works: typical applications in genetics only require a very small number of PCs (e.g., 10) within a small accuracy (e.g., two-three digits). Most importantly, Randomized Subspace Iteration features block iteration thus allowing higher granularity in out-of-core settings.

TeraPCA in a Nutshell



Algorithm Randomized Subspace Iteration

Input: $A^\top \in \mathbb{R}^{n\times m}$, initial guess matrix $X_0 \in \mathbb{R}^{m\times s}$ 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 A.

$C = A(A^\top X_0)$
repeat
  $Q = \mathtt{orth}(C)$
  $C = AA^\top Q$
  $M = Q^\top C$
  Compute the eigenvalue decomposition $M = XDX^\top$
  $C = QX$
until convergence
return first k columns of Q

Algorithm Out-of-core MMV $C = A(A^\top X)$

Input: $\zeta > 0$, $m \times s$ matrix X.
Output: $m \times s$ matrix C.

$C = 0$
for $i = 1 : \zeta$ **do**
  Fetch the i-th row-block of $A^\top$
  $C = C + A_i(A_i^\top X)$
end for

Datasets & Experimental Setup

- Our goal is to approximate the **ten** leading Principal Components (PCs). For TeraPCA we set the dimension (s) of the initial approximation subspace equal to **twenty**.
- All our experiments ran at Purdue's Brown cluster on a dedicated node which features an Intel Xeon Gold 6126 @ 2.6 GHz processor, 96 GB RAM and 64-bit CentOS Linux 7 operating system.

Dataset	Size (.PED file)	Size (.BED file)	# Samples	# SNPs
S_1 (simulated)	19 GB	120 MB	5,000	1,000,000
S_2 (simulated)	38 GB	239 MB	10,000	1,000,000
S_3 (simulated)	373 GB	24 GB	100,000	1,000,000
S_4 (simulated)	1.9 TB	117 GB	500,000	1,000,000
S_5 (simulated)	3.7 TB	233 GB	1,000,000	1,000,000
S_6 (simulated)	38 GB	2.4 GB	100,000	100,000
S_7 (simulated)	150 GB	9.4 GB	2,000	20,000,000
HGDP	615 MB	39 MB	1,043	154,417
1000 Genomes	8.4 GB	483 MB	2,504	808,704
PRK	2 GB	126 MB	4,706	111,831
T2D	1.8 GB	111 MB	6,370	72,457

- TeraPCA GitHub Repository: https://github.com/aritra90/TeraPCA
- Data Simulator GitHub Repository: https://github.com/eugeniamaria/DataSimulator

Time comparisons

Comparison with FlashPCA2 (only 2GB RAM was allowed)

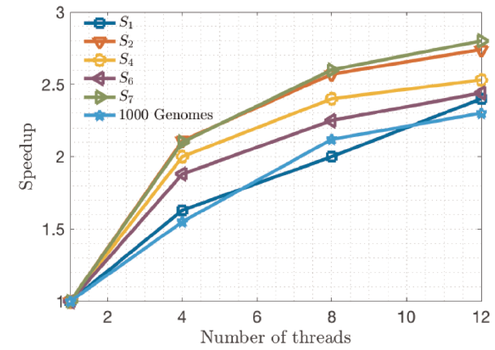
Table: TeraPCA vs FlashPCA2 (* indicates no convergence after 50 hrs).

Dataset	TeraPCA	FlashPCA2	Speed-up
S_1	26.2 mins	33.3 mins	1.27
S_2	39.3 mins	87.5 mins	2.22
S_3	7.9 hrs	35.6 hrs	4.50
S_4	7.3 hrs	n/a*	∞
S_5	13.2 hrs	n/a*	∞
S_6	39.5 mins	141.1 mins	3.57
S_7	37.3 mins	106.5 mins	2.86
HGDP	6.5 secs	7.7 secs	1.22
1000 Genomes	4.3 mins	3.5 mins	0.81
T2D	96 secs	119 secs	1.24
PRK	76 secs	73 secs	0.96

TeraPCA has an advantage over FlashPCA2 (which is based on Implicit Restarted Arnoldi) due to its block nature which allows to:

- search for multiple PCs simultaneously
- perform more computations per epoch
- take advantage of state-of-the-art dense linear algebra kernels (e.g., BLAS, LAPACK)

Speedup using Multithreading



Accuracy Results

Accuracy of leading PCs

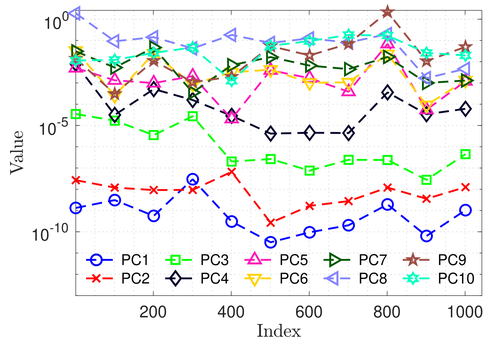


Figure: Element-wise relative error of the 10 leading PCs computed by TeraPCA versus those computed by LAPACK for the HGDP dataset.

Accuracy of leading eigenvalues

Table: Accuracy of the 10 leading eigenvalues computed for TeraPCA and FlashPCA2.

eigenvalue index	relative error		eigenvalue index	relative error	
	TeraPCA	FlashPCA2		TeraPCA	FlashPCA2
1	9.91E-15	1.74E-03	6	3.01E-06	7.63E-04
2	1.02E-13	1.30E-03	7	3.36E-06	1.47E-03
3	5.65E-11	1.49E-03	8	1.04E-05	6.81E-04
4	2.18E-08	1.31E-03	9	7.11E-05	1.28E-03
5	2.65E-06	1.10E-03	10	1.74E-04	7.44E-04

Acknowledgements

This work was supported by NSF BigData 1661760, NSF IIS1319280 and NSF IIS 1302231.