Low rank approximation and write avoiding algorithms

Laura Grigori

Alpines

Inria Paris - LJLL, UPMC

with A. Ayala, S. Cayrols, J. Demmel

informatics mathematics

Motivation - the communication wall

Time to move data >> time per flop

- Gap steadily and exponentially growing over time

Annual improvements

- Time / flop **59%** (1995-2004) **34%** (2006-2016)
- Interprocessor bandwidth **26%**
- Interprocessor latency **15%**
- DRAM latency **5.5%**

DRAM latency:

- DDR2 (2007) ~ 120 ns 1x
- DDR4 (2014) ~ 45 ns 2.6x in 7 years
- Stacked memory ~ similar to DDR4

Time/flop

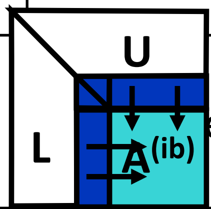
- 2006 Intel Yonah ~ 2GHz x 2 cores (32 GFlops/chip) 1x
- 2015 Intel Haswell ~2.3GHz x 16 cores (588 GFlops/chip) 18x in 9 years

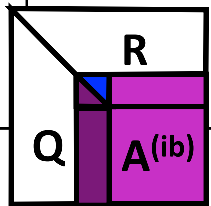
Source: J. Shalf, LBNL

2D Parallel algorithms and communication bounds

- Memory per processor = n^2 / P, the lower bounds on communication are
  #words_moved $\geq \Omega (n^2 / P^{1/2})$, #messages $\geq \Omega (P^{1/2})$

Algorithm	Minimizing #words (not #messages)	Minimizing #words and #messages
Cholesky	ScaLAPACK	ScaLAPACK
LU	ScaLAPACK es partial pivoting	[LG, Demmel, Xiang, 08] [Khabou, Demmel, LG, Gu, 12] uses tournament pivoting
QR	ScaLAPACK	[Demmel, LG, Hoemmen, Langou, 08] uses different representation of Q
RRQR	ScaLAPACK	[Demmel, LG, Gu, Xiang 13] uses tournament pivoting, 3x flops





- Only several references shown, block algorithms (ScaLAPACK) and communication avoiding algorithms
- CA algorithms exist also for SVD and eigenvalue computation

Parallel write avoiding algorithms

Need to avoid writing suggested by emerging memory technologies, as NVMs:

- Writes more expensive (in time and energy) than reads
- Writes are less reliable than reads

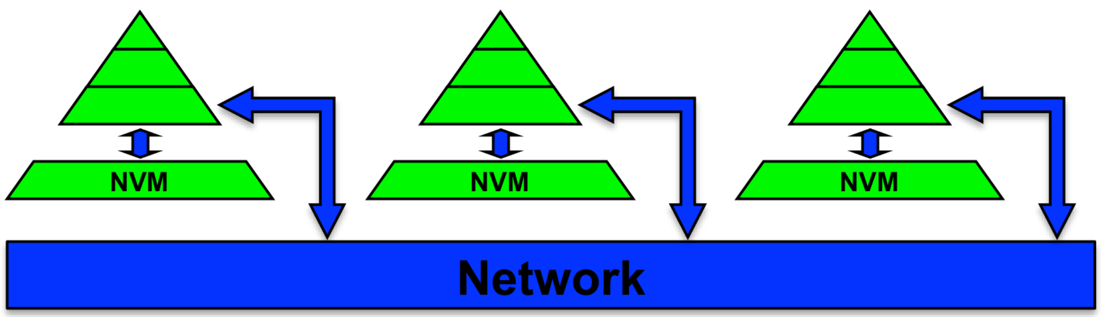
Some examples:

- Phase Change Memory: Reads 25 us latency

  Writes: 15x slower than reads (latency and bandwidth)

  consume 10x more energy
- Conductive Bridging RAM - CBRAM

  Writes: use more energy (1pJ) than reads (50 fJ)
- Gap improving by new technologies such as XPoint and other FLASH alternatives, but not eliminated



Parallel write-avoiding algorithms

- Matrix A does not fit in DRAM (of size M), need to use NVM (of size n^2 / P)

- Two lower bounds on volume of communication
  - Interprocessor communication: $\Omega(n^2 / P^{1/2})$
  - Writes to NVM: n^2 / P

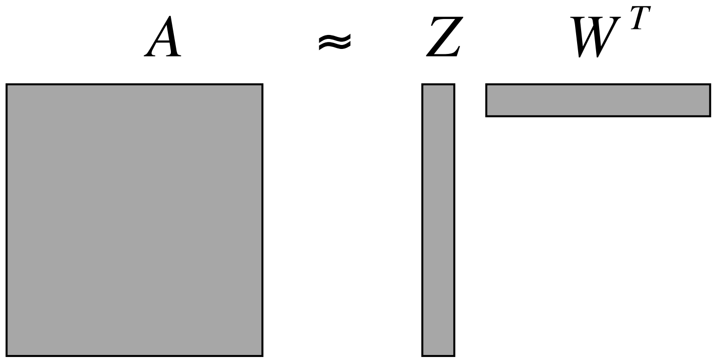
- Result: any three-nested loop algorithm (matrix multiplication, LU,..), must asymptotically exceed at least one of these lower bounds
  - If $\Omega(n^2 / P^{1/2})$ words are transferred over the network, then $\Omega(n^2 / P^{2/3})$ words must be written to NVM !

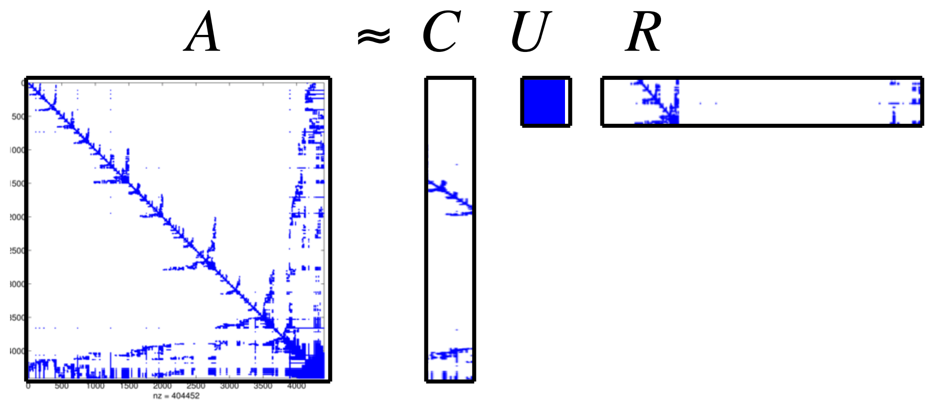
- Parallel LU: choice of best algorithm depends on hardware parameters

	#words interprocessor comm.	#writes NVM
Left-looking	$O((n^3 \log^2 P) / (P M^{1/2}))$	$O(n^2 / P)$
Right-looking	$O((n^2 \log P) / P^{1/2})$	$O((n^2 \log^2 P) / P^{1/2})$

Low rank matrix approximation

- Problem: given m x n matrix A, compute rank-k approximation ZW^T, where Z is m x k and W^T is k x n.





- Problem with diverse applications
  - from scientific computing: fast solvers for integral equations, H-matrices
  - to data analytics: principal component analysis, image processing, …

- Used in iterative process by multiplication with a set of vectors

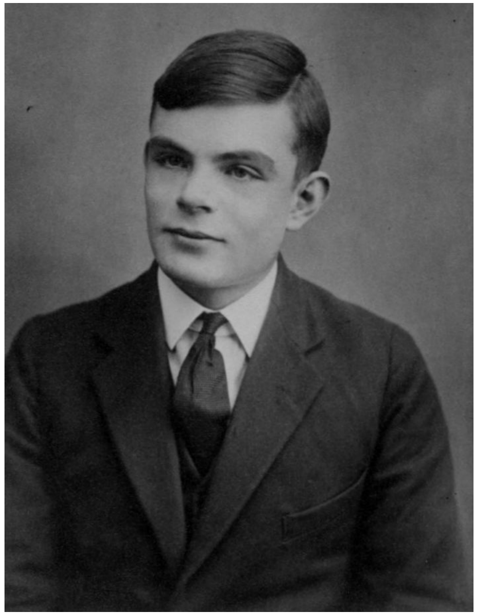
$$
\begin{array}{lccc}
 & Ax & \rightarrow & ZW^T x \\
\text{Flops}: & 2mn & \rightarrow & 2(m+n)k
\end{array}
$$

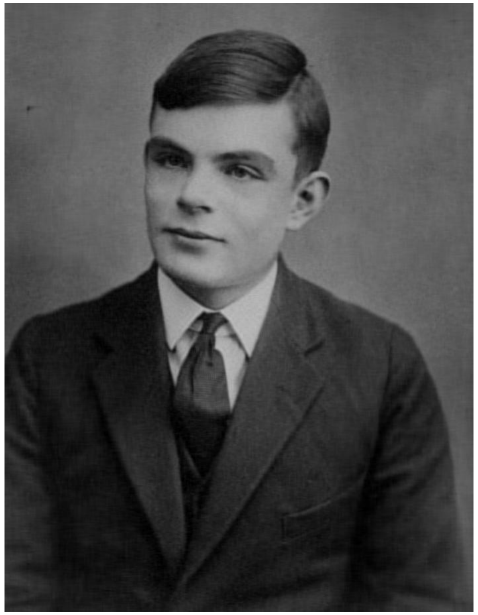
Low rank matrix approximation

- Problem: given m x n matrix A, compute rank-k approximation ZW^T, where Z is m x k and W^T is k x n.

- Best rank-k approximation $A_k = U_k \Sigma_k V_k^T$ is the rank-k truncated SVD of A

$$\min_{\mathrm{rank}(\tilde{A}_k) \le k} \|A - \tilde{A}_k\|_2 = \|A - A_k\|_2 = \sigma_{k+1}(A)$$


Original image, 707x256


Rank-75 approximation, SVD

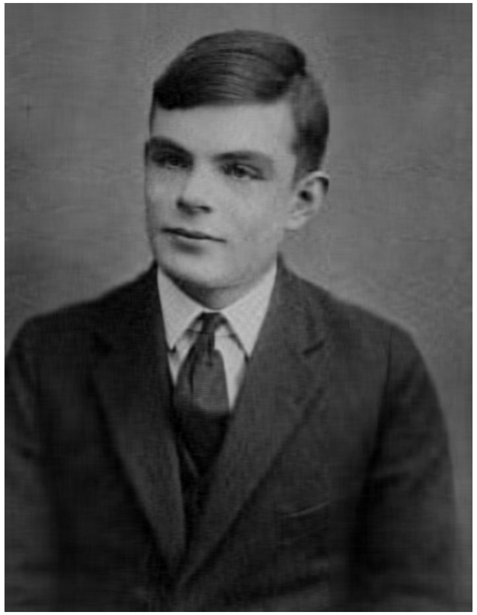
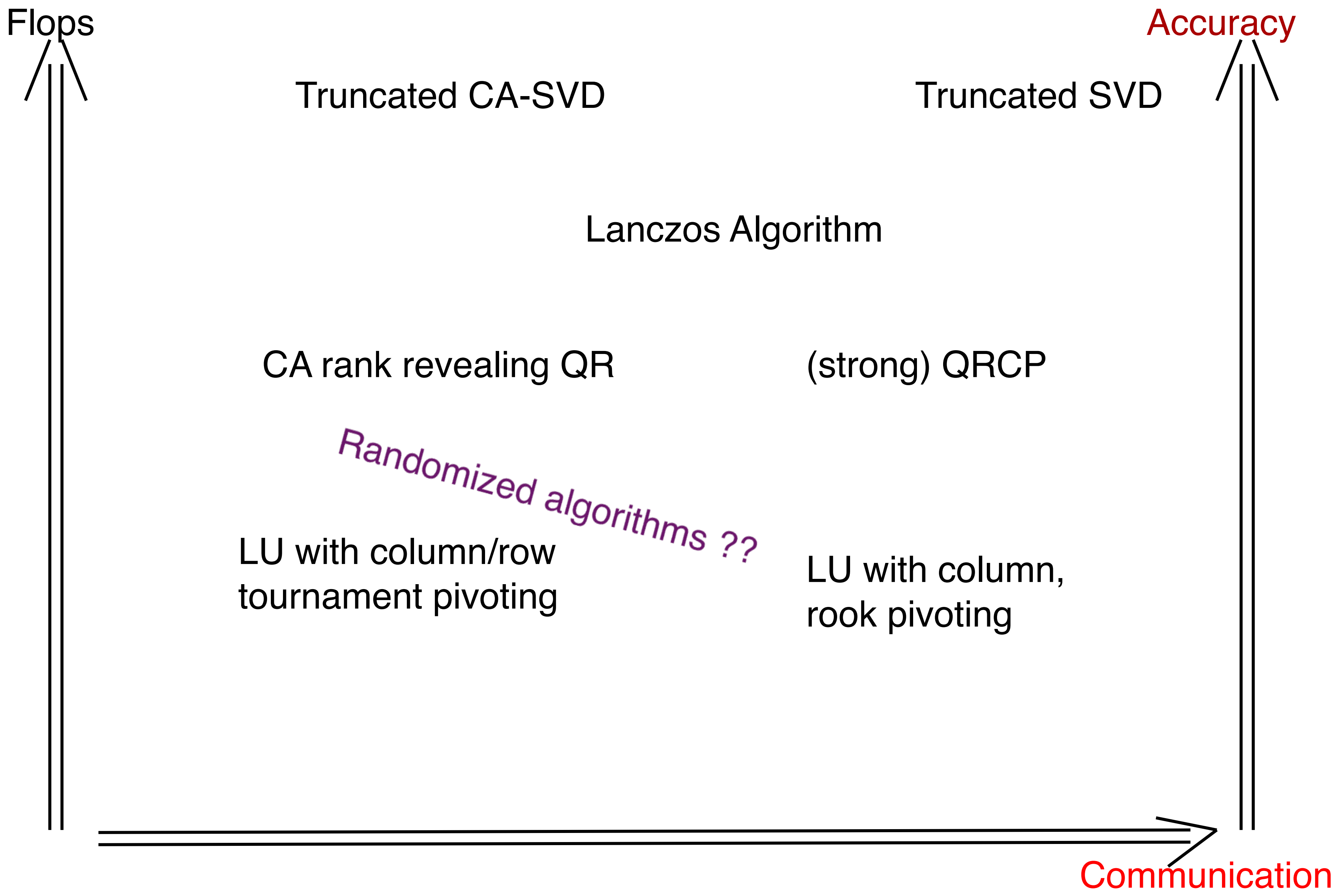

Rank-38 approximation, SVD

Image source: https://upload.wikimedia.org/wikipedia/commons/a/a1/Alan_Turing_Aged_16.jpg

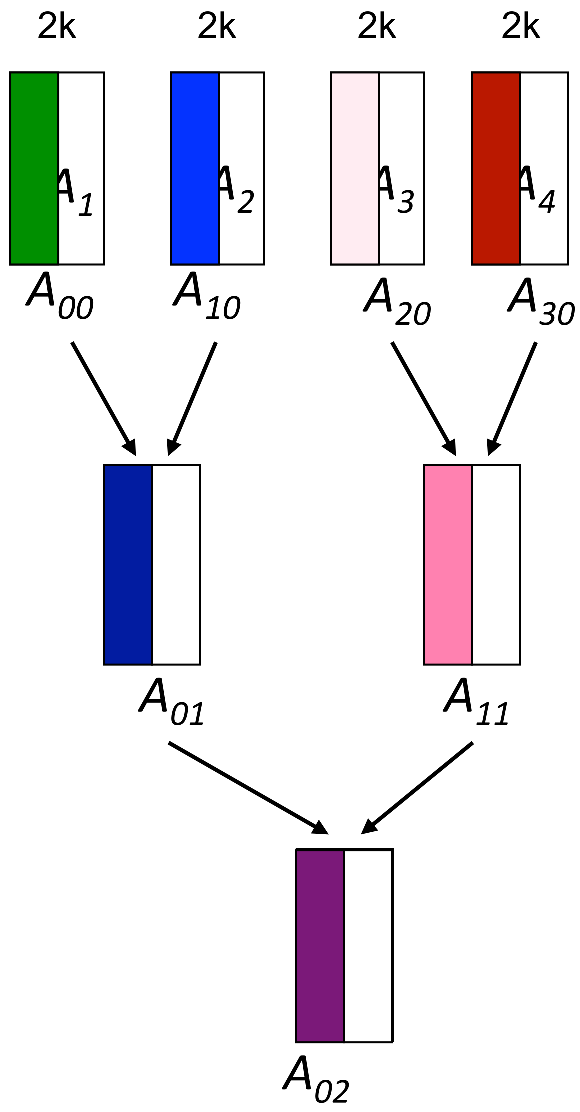
Low rank matrix approximation: trade-offs



Select k cols using tournament pivoting

Partition $A=(A_1, A_2, A_3, A_4)$.
Select k cols from each column block,
by using QR with column pivoting

At each level i of the tree
At each node j do in parallel
Let $A_{v,i-1}$, $A_{w,i-1}$ be the cols selected by the children of node j
Select b cols from $(A_{v,i-1}, A_{w,i-1})$, by using QR with column pivoting
Return columns in A_{ji}



LU_CRTP: LU with column/row tournament pivoting

- Given A of size m x n, compute a factorization

$$P_r A P_c = \begin{pmatrix} \bar{A}_{11} & \bar{A}_{12} \\ \bar{A}_{21} & \bar{A}_{22} \end{pmatrix} = \begin{pmatrix} I & \\ \bar{A}_{21}\bar{A}_{11}^{-1} & I \end{pmatrix} \begin{pmatrix} \bar{A}_{11} & \bar{A}_{12} \\ & S(\bar{A}_{11}) \end{pmatrix},$$

$$S(\bar{A}_{11}) = \bar{A}_{22} - \bar{A}_{21}\bar{A}_{11}^{-1}\bar{A}_{12},$$

where $\bar{A}_{11}$ is k x k, P_r and P_c are chosen by using tournament pivoting

- LU_CRTP factorization satisfies

$$1 \le \frac{\sigma_i(A)}{\sigma_i(\bar{A}_{11})}, \frac{\sigma_j(S(\bar{A}_{11}))}{\sigma_{k+j}(A)} \le \sqrt{\left(1 + \mathrm{F}^2(n-k)\right)\left(1 + F^2(m-k)\right)},$$

$$\left\| S(\bar{A}_{11}) \right\|_{\max} \le \min\left(\left(1 + F\sqrt{k}\right) \|A\|_{\max}, F\sqrt{1 + F^2(m-k)}\,\sigma_k(A) \right)$$

for any $1 \le i \le k$ and $1 \le j \le \min(m,n) - k$, $F \le \frac{1}{\sqrt{2k}}(n/k)^{\log_2(2\sqrt{2}k)}$

LU_CRTP

- Given LU_CRTP factorization

$$P_r A P_c = \begin{pmatrix} \bar{A}_{11} & \bar{A}_{12} \\ \bar{A}_{21} & \bar{A}_{22} \end{pmatrix} = \begin{pmatrix} I & \\ \bar{A}_{21}\bar{A}_{11}^{-1} & I \end{pmatrix} \begin{pmatrix} \bar{A}_{11} & \bar{A}_{12} \\ & S(\bar{A}_{11}) \end{pmatrix},$$

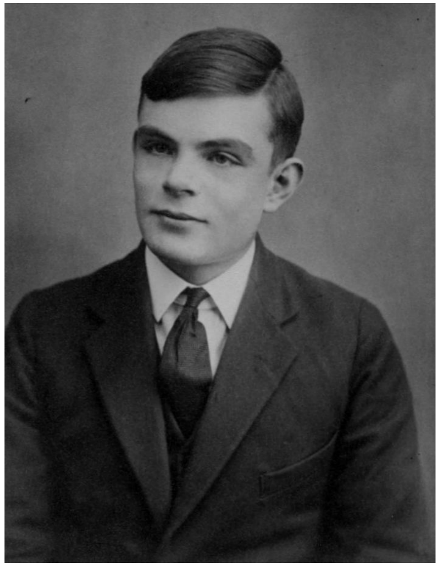
  the rank - k CUR approximation is

$$\tilde{A}_k = \begin{pmatrix} I \\ \bar{A}_{21}\bar{A}_{11}^{-1} \end{pmatrix} \begin{pmatrix} \bar{A}_{11} & \bar{A}_{12} \end{pmatrix} = \begin{pmatrix} \bar{A}_{11} \\ \bar{A}_{21} \end{pmatrix} \bar{A}_{11}^{-1} \begin{pmatrix} \bar{A}_{11} & \bar{A}_{12} \end{pmatrix}$$

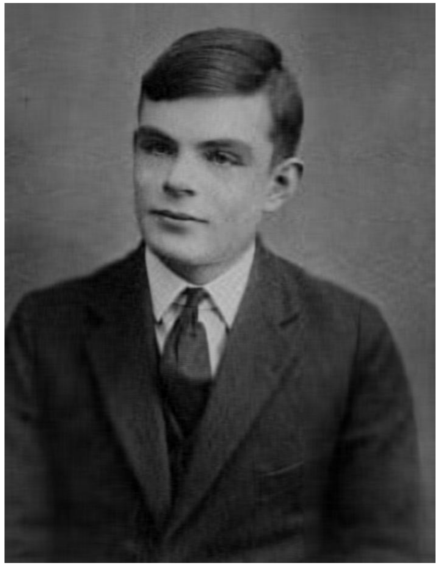
- $\bar{A}_{11}^{-1}$ is never formed, its factorization is used when $\tilde{A}_k$ is applied to a vector
- In randomized algorithms, U = C^+ A R^+, where C^+, R^+ are Moore-Penrose generalized inverses

Results for image of size 256x707

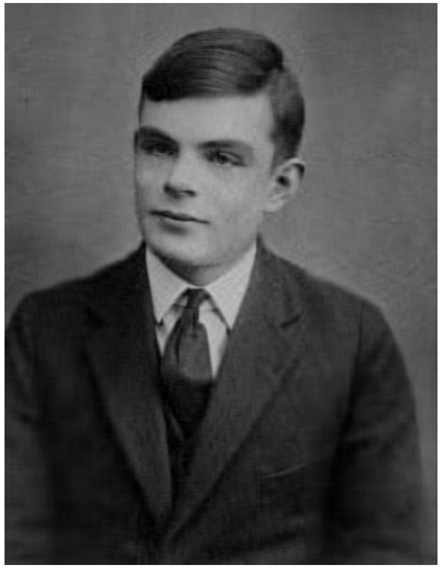
Original image, 707x256



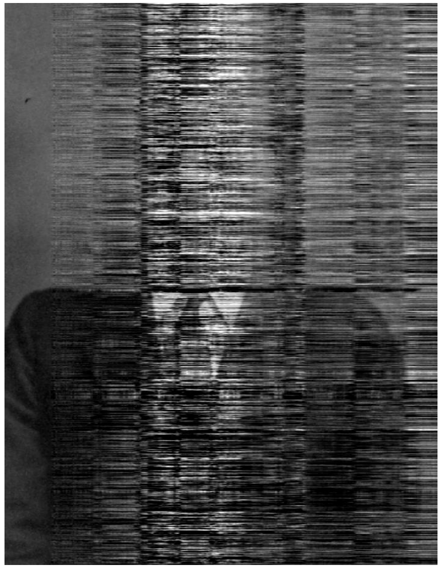
SVD: Rank-38 approximation



SVD: Rank-75 approximation

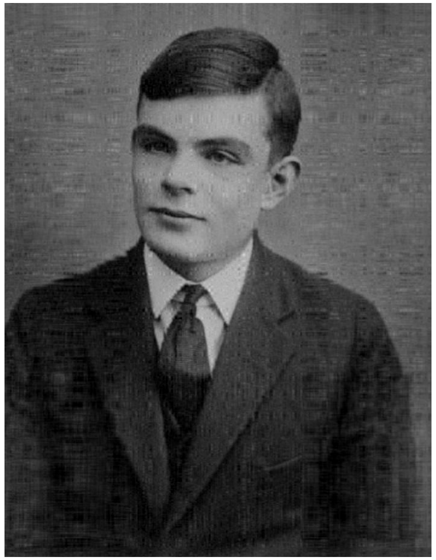


LUPP: Rank-75 approximation



LU_CRTP: Rank-38 approx.

LU_CRTP: Rank-75 approx.



Tournament pivoting for sparse matrices

A has arbitrary sparsity structure

$$flops(TP_{FT}) \le 2nnz(A)k^2$$

$$flops(TP_{BT}) \le 8\frac{nnz(A)}{P}k^2 \log\frac{n}{k}$$

$G(A^T A)$ is an $n^{1/2}$ - separable graph

$$flops(TP_{FT}) \le O\left(nnz(A)k^{3/2}\right)$$

$$flops(TP_{BT}) \le O\left(\frac{nnz(A)}{P}k^{3/2}\log\frac{n}{k}\right)$$

- Randomized algorithm by Clarkson and Woodruff, STOC'13

Given n x n matrix A, it computes LDW^T, where D is k x k, such that

$\left\|A - LDW^T\right\|_F \le (1+\varepsilon)\left\|A - A_k\right\|_F$, $\quad A_k$ is the best rank - k approximation.

flops $\le O(nnz(A)) + n\varepsilon^{-4}\log^{O(1)}(n\varepsilon^{-4})$

- Tournament pivoting is faster if $\varepsilon \le \dfrac{1}{\left(nnz(A)/n\right)^{1/4}}$

  or if $\varepsilon = 0.1$ and $nnz(A)/n \le 10^4$

Performance results

Comparison of number of nonzeros in the factors L/U, Q/R.

Name/size	Nnz A(:,1:K)	Rank K	Nnz QRCP/ LU_CRTP	Nnz LU_CRTP/ LUPP
Rfdevice 74104	633	128	10.0	1.1
	2255	512	82.6	0.9
	4681	1024	207.2	0.0
Parab_fem 525825	896	128	-	0.5
	3584	512	-	0.3
	7168	1024	-	0.2

Performance results

Selection of 256 columns by tournament pivoting

Edison, Cray XC30 (NERSC) – 2x12-core Intel Ivy Bridge (2.4 GHz)

Tournament pivoting uses SPQR (T. Davis) + dGEQP3 (Lapack), time in secs

Matrices:	n x n	n x n/32
• Parab_fem:	528825 x 528825	528825 x 16432
• Mac_econ:	206500 x 206500	206500 x 6453

	Time n x 2k	**Time n x n/32 SPQR+GEQP3**	**Number of MPI processes 16**	**32**	**64**	**128**	**256**	**512**	**1024**
Parab_fem	0.26	0.26+1129	46.7	24.5	13.7	8.4	5.9	4.8	4.4
Mac_econ	0.46	25.4+510	132.7	86.3	111.4	59.6	27.2	-	-

Conclusions

- Deterministic low rank approximation algorithm
  - Accuracy close to rank revealing QR factorization
  - Complexity close to randomized algorithms
- Future work
  - Design algorithms that do not need explicitly the matrix
  - Do a thorough comparison with randomized algorithms

Thanks to: EC H2020 NLAFET

Further information:

http://www-rocq.inria.fr/who/Laura.Grigori/