Column Subset Selection: TCS and NLA

Michael W. Mahoney

ICSI and Department of Statistics, UC Berkeley

Complexity of Matrix Computations Seminar

November 10, 2021

Before I start: A public service announcement:

Prospectus for a Randomized BLAS and LAPACK

A part of the BALLISTIC project

November 8, 2021

Abstract

This report presents a preliminary plan for an LAPACK-like library based on randomized numerical linear algebra (RNLA), including draft implementations in Python and Matlab.[1] Our goal is to gather feedback from user and developer communities before we develop highly optimized code. The most pressing issue is to determine the appropriateness of RandLAPACK's conceptual architecture and the overall project scope. Appropriateness of scope should be assessed both with regard to mathematical functionality and target computer architecture.

- We are putting randomness into LAPACK.
- Lots of practical and conceptual questions.
- We have a design document, and we want feedback.
- For details, contact me or Jim Demmel or Riley Murray (rjmurray@berkeley.edu, who is leading the effort)

What we'll cover

- Background/overview
- TCS and NLA on CSSP
- Recent developments
- (no rank-revealing)

What we'll cover

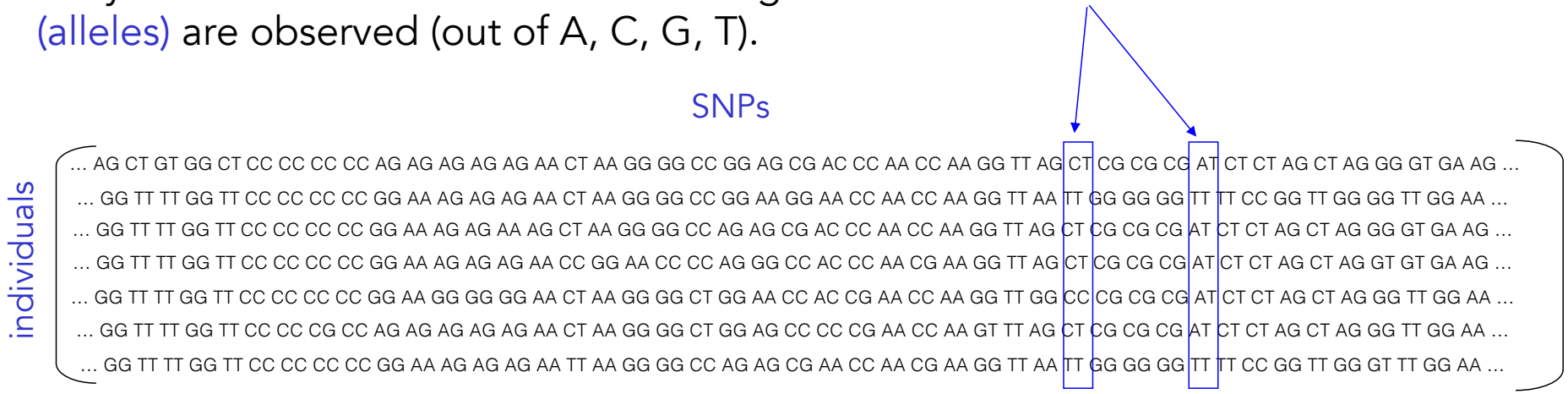
- Background/overview
- TCS and NLA on CSSP
- Recent developments
- (no rank-revealing)

A "data" application: choosing good columns as features

In Human Genetics,

Single Nucleotide Polymorphisms: the most common type of genetic variation in the genome across different individuals.

They are known locations at the human genome where two alternate nucleotide bases (alleles) are observed (out of A, C, G, T).



- Matrices with thousands of individuals and hundreds of thousands of SNPs are available.
- This is basically a "term-document" matrix.
- What counts as a "good" set of columns?

TCS versus NLA approaches (1 of 2)

Model of computation

- Streaming: 1 pass: $+\Delta A_{ij}$, $+/-\Delta A_{ij}$
- Pass efficient: 2 (or t) passes, additional space, and additional time
- RAM: space/time
- Communication-aware or not:
- Exact arithmetic or not:

(TLDR: in TCS: no pivoting issues; cache and conditioning okay; exact arithmetic)

Ways to choose columns

- i.i.d. sampling: uniform or not; column norms; approx. leverage scores: oversampling
- Adaptive sampling: multiple iterations
- Volume sampling: get exactly k columns

Quantities of interest

- Reconstruction error (spectral, Frobenius, unitarily invariant, etc.)
- C is well-spread-out (e.g., volume)
- Find columns fast; "X" matrix (in $A \approx CX$) fast to compute
- Good numerical properties: "X" matrix is well conditioned, largest entry ≤ 1, pivot rule decisions, etc.
- Good implicit statistical properties (ridge regularization in expectation)
- Preserve sparsity; construct "X" using only information in C; etc.

TCS versus NLA approaches (2 of 2)

Resources

- Traditional: Space, time, etc.
- Non-traditional: Oversampling ($c \gg k$), randomness (fail w.p. δ), etc.

Things to do

- Sample columns; "do SVD" on sample
- Random projection; "do SVD" on sample
- Sample columns; interested in columns

Pre-History (TCS*)

Additive error**: $\|A-A'\|_F \leq \|A-A_k\|_F + \epsilon \|A\|_F$

- Sample columns w.r.t. norms; do SVD on sample
- Random projection; do SVD on sketch (JL on cols)
- Sample columns w.r.t. norms; construct CX/CUR

Relative error: $\|A-A'\|_F \leq (1+\epsilon) \|A-A_k\|_F$

- Sample columns w.r.t. leverage scores***; do SVD on sample
- Random projection; do SVD on sketch (JL on cols defining left singular subspace)
- Sample columns w.r.t. leverage scores***; construct CX/CUR

* TLDR: in TCS: no pivoting issues; cache and conditioning okay; exact arithmetic

** Strong lower bounds, i.e., can't do better, in their streaming model of computation

*** *Approx leverage scores*: originally $O(mn^2)$; then $O(SVD)$; then $O(SVD_k)$; then *O(random proj time)*

What we'll cover

- Background/overview
- TCS and NLA on CSSP
- Recent developments
- (no rank-revealing)

A Column Subset Selection Problem (CSSP*)

Given an m-by-n matrix A and a rank parameter k, choose *exactly k columns* of A s.t. the m-by-k matrix C minimizes an error over all $O(n^k)$ choices for C, e.g.:

$$\begin{aligned} \min ||A - P_C A||_2 &= \min ||A - CC^+ A||_2, \\ &\quad \text{where } ||X||_2 = \max_{x \in \mathbb{R}^n : |x|=1} |Xx| \\ \min ||A - P_C A||_F &= \min ||A - CC^+ A||_F, \\ &\quad \text{where } ||X||_F^2 = \sum_{ij} X_{ij}^2 \end{aligned}$$

$P_C = CC^+$ is the projector matrix on the subspace spanned by the columns of C.

Complexity of the problem? $O(n^k mn)$ trivially works; NP-hard if *k* grows as a function of *n*. (NP-hardness in Civril & Magdon-Ismail '07)

* CSSP ≈ CX/CUR ≈ interpolative decomposition / two-sided id ≈ pseudoskeleton approximations ≈ Q from QR

Prior work (circa 2007)

NLA algorithms for the CSSP

1. Deterministic, typically greedy approaches.
2. Deep connection with the RRQR.
3. Strongest results so far (spectral norm): in $O(mn^2)$ time

Gu-Eisenstat 1996 (and a long line of others):

- Given an m-by-n matrix A, "there exists" an algorithm that picks exactly k columns of A such that

$$\|A - P_C A\|_2 \leq O(k^{1/2}(n-k)^{1/2}) \left\|A - P_{U_k} A\right\|_2$$

(more generally, some function p(k,n))

- Strongest results so far (Frobenius norm): in $O(n^k)$ time

$$\|A - P_C A\|_F \leq \sqrt{k\,(n-k)}\, \|A - P_{U_k} A\|_2$$

TCS algorithms for the CSSP

1. Randomized, some failure probability.
2. Pick c ≫ k columns, e.g., O(poly(k)).
3. Very strong bounds for the Frobenius (but not spectral) norm in low polynomial time.

Drineas and Mahoney 2006:

- Given an m-by-n matrix A, "there exists" an $O(mn^2)$ algorithm that picks at most $O(k \log k / \varepsilon^2)$ columns of A s.t. with probability at least $1\text{-}10^{-20}$

$$\|A - P_C A\|_F \leq (1+\epsilon) \left\|A - P_{U_k} A\right\|_F$$

Deshpande and Vempala 2006:

- $O(mnk^2)$ time, $O(k^2 \log k/\varepsilon^2)$ columns -> $(1\pm\varepsilon)$-approximation.
- They prove the existence of *k* columns of A s.t.

$$\|A - P_C A\|_F \leq \sqrt{k}\, \|A - P_{U_k} A\|_F$$

- Compare to prior best existence result:

$$\|A - P_C A\|_F \leq \sqrt{k}\sqrt{n-k}\, \|A - A_k\|_2$$

A hybrid two-stage algorithm

Boutsidis, Drineas, and Mahoney 2008

Given an m-by-n matrix A (assume m ¸ n for simplicity):

- (Randomized phase) Run a randomized algorithm to pick *c = O(k log(k))* columns.
- (Deterministic phase) Run a deterministic algorithm on the those columns* to pick exactly *k* columns of A and form an m-by-k matrix C.

* actually, not so simple … details matter: *the matrix consisting of the corresponding columns of the transpose of top-k right singular vectors*

Algorithm runs in $O(mn^2)$ and satisfies, with probability at least $1\text{-}10^{-20}$,

$$\|A - P_C A\|_2 \leq O\left(k^{3/4}\log^{1/2} k(n-k)^{1/4}\right)\|A - P_{U_k}A\|_2$$
$$\|A - P_C A\|_F \leq O\left(k\log^{1/2} k\right)\|A - P_{U_k}A\|_F$$

Compared to NLA:

1. Time is comparable with NLA algorithms.
2. Spectral norm bound grows as $(n\text{-}k)^{1/4}$ instead of $(n\text{-}k)^{1/2}$!
3. W.r.t. k, it is $k^{1/4}\log^{1/2}k$ worse.
4. First asymptotic improvement of the work of Gu & Eisenstat 1996.

Compared to TCS:

1. An efficient algorithmic result.
2. Frobenius norm bound at most $(k \log k)^{1/2}$ worse than the previous best *existential* result.

What about between k and $O((k/\varepsilon^2)\log(k/\varepsilon^2))$?

Q1: How many columns are needed to get relative-error (Frobenius norm) approximations ?

- **DMM06/DMM08 (SIMAX):** $O((k/\varepsilon^2)\log(k/\varepsilon^2))$ columns -> relative-error
- **Deshpande & Rademacher (FOCS '10):** with exactly k columns, we can get

$$\left\|A - CC^{\dagger}A\right\|_F \leq \sqrt{k}\,\|A - A_k\|_F$$

Q2: What about the range between k and O(k log k)?

- **Boutsidis, Drineas, & Magdon-Ismail, (FOCS 11):** Relative-error by selecting $c=2k/\varepsilon+o(1)$ columns!
  (ideas from Batson, Spielman, & Srivastava (STOC '09) on graph sparsifiers ---
  running time is $O((mnk+nk^3)\varepsilon^{-1})$, simplicity is gone.)
- **Deshpande & Vempala (RANDOM 2006):** Relative-error needs at least k/ε columns.
- **Guruswami & Sinop (SODA 2012):** Approach, based on volume sampling, guarantee
  $(c+1)/(c+1-k)$ relative error bounds.
  This bound is asymptotically optimal (up to lower order terms).
  Deterministic alg takes $O(cnm^3 \log m)$ time; randomized alg takes $O(cnm^2)$ time
- **Guruswami & Sinop (FOCS 2011):** Apply column-based reconstruction in Quadratic Integer Programming.

Adaptive sampling

Deshpande *et al.* (2006) ToC:

Adaptive sampling algorithm: (pick columns adaptively in multiple rounds`

- First, pick c columns of A using (Euclidean-norm based) probabilities: $p_i = \frac{\|A_{*i}\|_2^2}{\|A\|_F^2}$
- At the t-th round, compute the residual matrix $E = A - CC^{+}A$,
- Iteratively, pick c columns of A using (Euclidean-norm based) probabilities: $p_i = \frac{\|E_{*i}\|_2^2}{\|E\|_F^2}$

Theorem: After t rounds, where, in each round,

$$c = O\left(\frac{k \log(1/\delta)}{\epsilon^2}\right)$$

columns of A are sampled, with probability at least 1-t**δ**, the error is:

$$\|A - CC^{+}A\|_{\mathrm{F}}^2 \leqslant \frac{1}{1-\epsilon}\|A - A_k\|_{\mathrm{F}}^2 + \epsilon^t \|A\|_{\mathrm{F}}^2$$

Examples of extensions:

- Drineas and Mahoney (LAA 2007): simple inductive proof, via matrix-matrix multiplication
- Paul, Magdon-Ismail, and Drineas, NIPS 2015: extended to leverage-score sampling: error after t rounds depends on $A\text{-}A_{tk}$ instead of $A\text{-}A_k$!

Volume sampling

Deshpande *et al.* (2006) ToC:

- **Algorithm**: Sample a set of columns with probability proportional to their volume.
- **Theorem:** In expectation,

$$\mathbf{E}\left(\left\|A - CC^{+}A\right\|_F^2\right) \leq (k+1)\left\|A - A_k\right\|_F^2$$

Can combine this with O(log k) rounds of adaptive sampling to get $1 \mp \epsilon$ relative-error.

BUT: computing the sampling probabilities P_S is (very) hard (must be approximated)

BUT BUT: Adaptive sampling can be used to approximate the volume sampling probabilities!

Overall: Use adaptive sampling to simulate volume sampling; return a set S of k columns of A to form an m-by-k matrix C such that

$$\mathbf{E}\left(\left\|A - CC^{+}A\right\|_F^2\right) \leq (k+1)!\left\|A - A_k\right\|_F^2$$

Combining with O(k log k) rounds of adaptive sampling reduces the above error to relative error by sampling c= $O(k/\epsilon + k^2 \log(k))$ each round

Extensions:

- Lots of TCS theoretical follow-up … not really practical … until …

What we'll cover

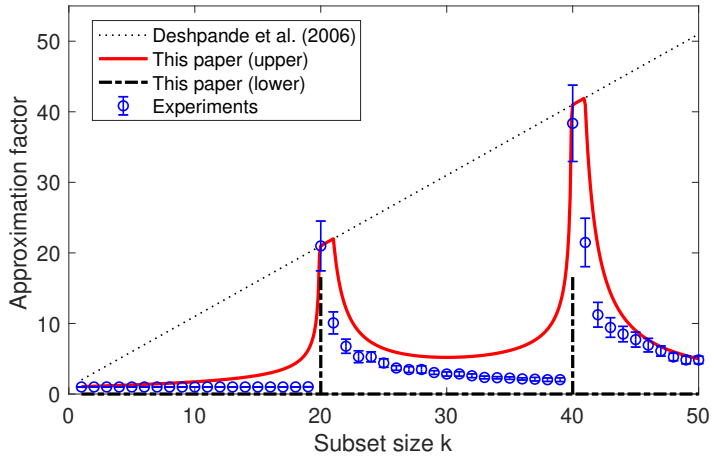
- Background/overview
- TCS and NLA on CSSP
- Recent developments
- (no rank-revealing)

Improved guarantees and a multiple-descent curve for Column Subset Selection and the Nyström method

Michał Dereziński, Rajiv Khanna and Michael Mahoney

University of California, Berkeley

NeurIPS 2020

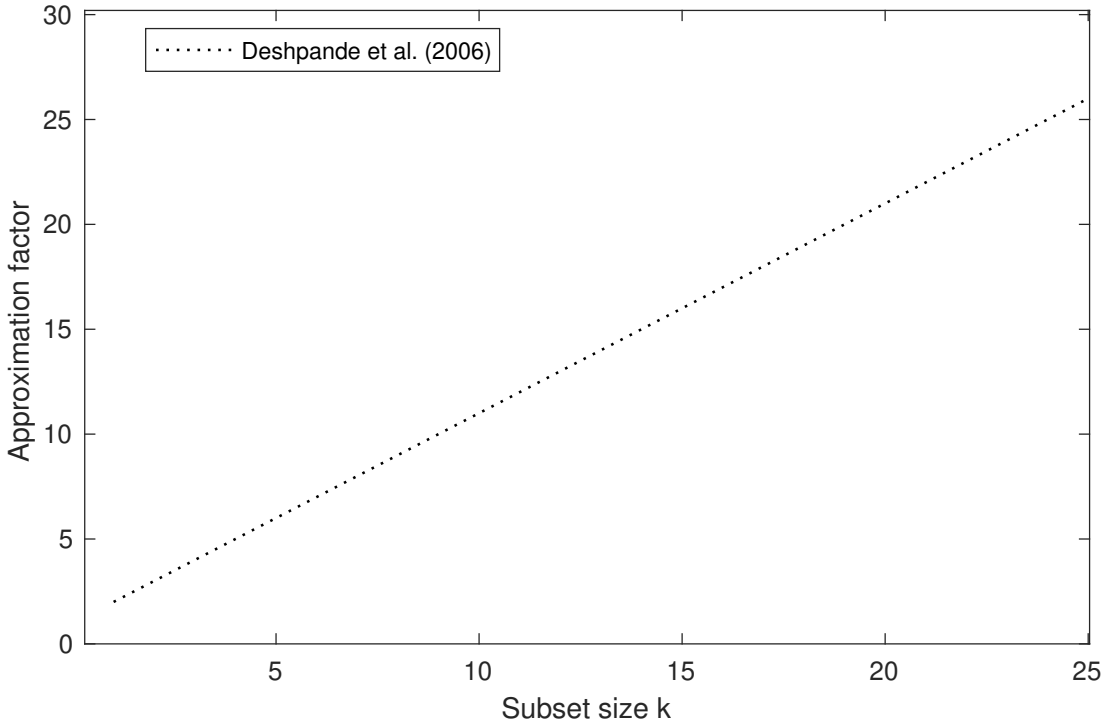


(Won the Best Paper Award at NeurIPS20 (the top ML venue): +1 for Linear Algebra!)

Prior worst-case guarantees

$$\underbrace{[\text{DRVW06}]}_{\text{CSSP}}, \underbrace{[\text{BW09}]}_{\text{Nyström}} : \quad \underbrace{\text{Approximation factor} \quad \leq \quad k+1}_{\text{worst-case optimal!}}$$

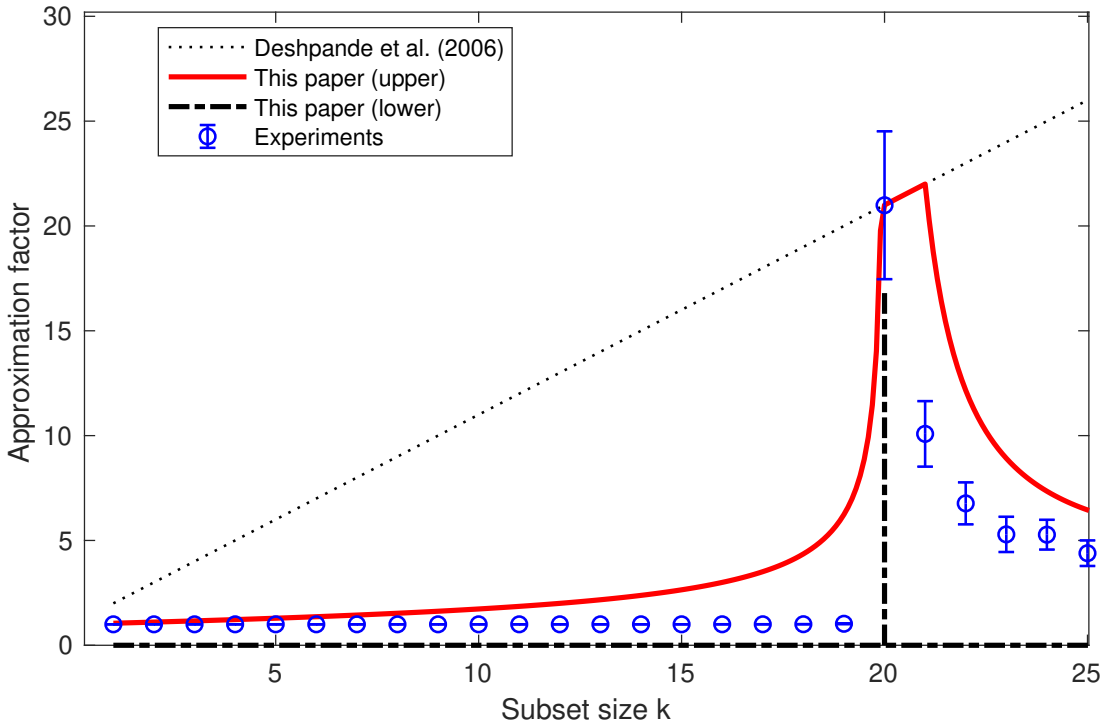
E.g., used in ICML 2019 Best Paper [BRVDW19] for GP regression



Prior worst-case guarantees

$$\underbrace{[\text{DRVW06}]}_{\text{CSSP}}, \underbrace{[\text{BW09}]}_{\text{Nyström}} : \quad \underbrace{\text{Approximation factor} \quad \leq \quad k+1}_{\text{worst-case optimal!}}$$

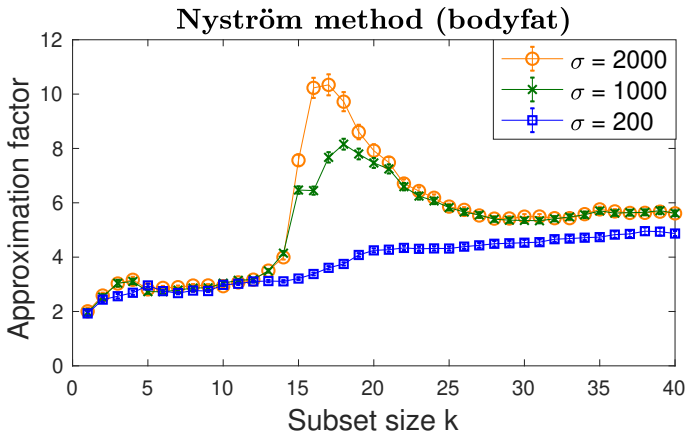
E.g., used in ICML 2019 Best Paper [BRVDW19] for GP regression

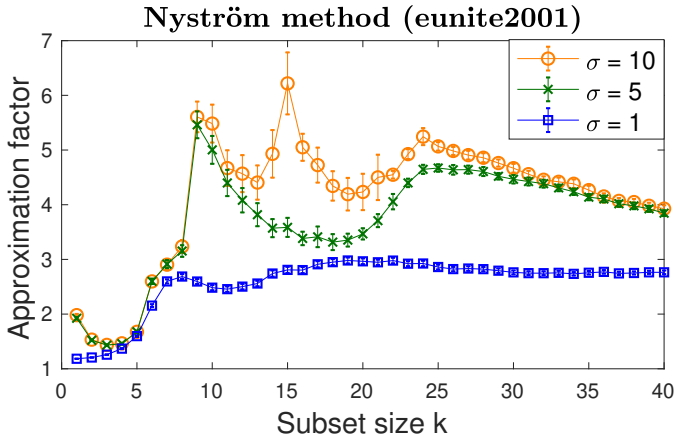


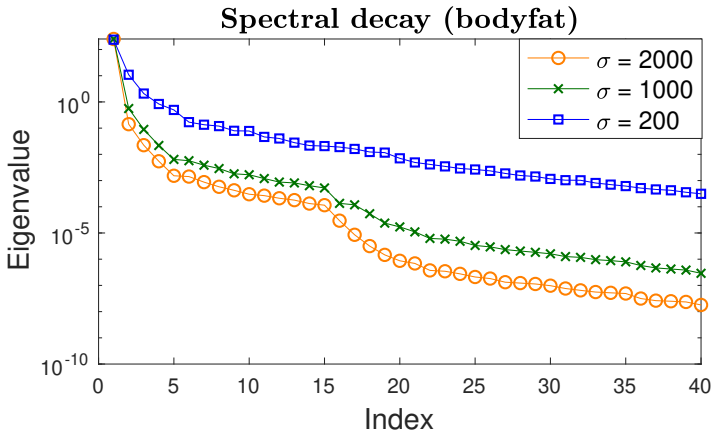
We go beyond worst case analysis!

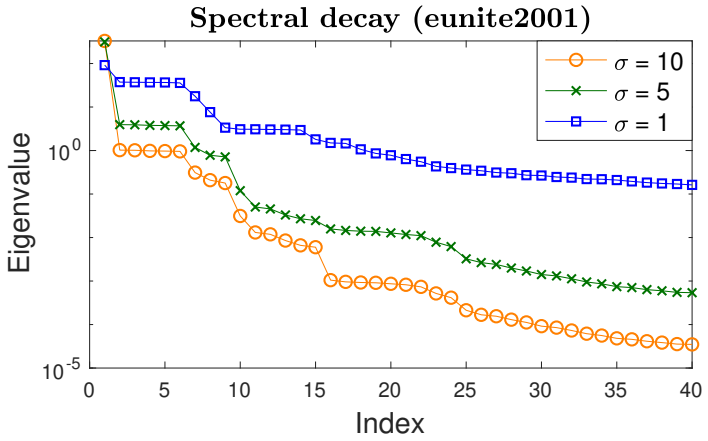
Multiple-descent in real-world subset selection

Kernel: Gaussian RBF, $\langle \mathbf{a}_i, \mathbf{a}_j \rangle_K = \exp(-\|\mathbf{a}_i - \mathbf{a}_j\|^2/\sigma^2)$









Datasets from the Libsvm repository [CL11]

New: Improved guarantees under smooth spectral decay

Smooth spectral decay $\implies$ no spikes!

1. Polynomial spectral decay: ith singular value $\asymp i^{-p}$

   Approximation factor $\leq O(1+p)$ for all k

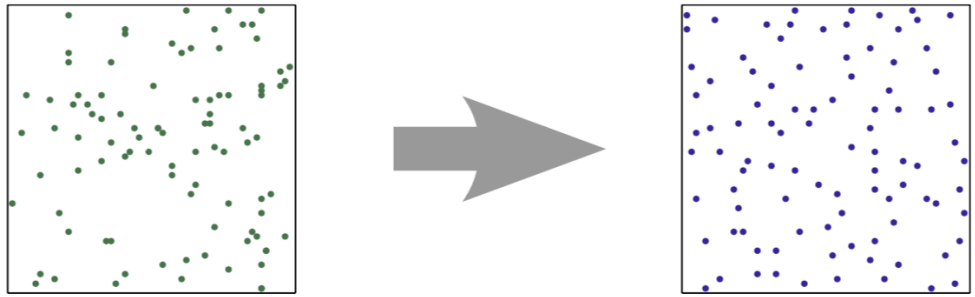
2. Exponential spectral decay: ith singular value $\asymp (1-\delta)^i$

   Approximation factor $\leq O(1+\delta k)$ for all k

Method: Determinantal Point Processes (DPPs)

Non-i.i.d. randomized selection of column subset S

Negative correlation: $\Pr(i \in S \mid j \in S) < \Pr(i \in S)$



i.i.d. (left) versus DPP (right)

- Fast algorithms: [CDV20] (here at *NeurIPS'20*)

  "Sampling from a k-DPP without looking at all items"

- Learn more: [DM20] (to appear in *Notices of the AMS*)

  "Determinantal point processes in randomized numerical linear algebra"

Image from [KT12]