Aalto University
School of Science
and Technology

Simple and Deterministic Matrix Sketching[1]

Presented by:
Hristo Georgiev and Huibin Shen

Department of Information and Computer Science
Aalto University, School of Science

March 24, 2014

[1] Authored by Edo Liberty and and won the *KDD-2013 best paper* award[4].

Content

- Background
- Related work
- Frequent directions
- Experiments and Results
- Conclusion

What is a sketch?

Well, not exactly . . .

What is a sketch? (cont.)

- A sketch of a matrix A is another matrix B which is significantly smaller than A, but still approximates it well.
- A good sketch matrix is one on which some computations can be performed, without *much* loss of precision.
- Formally, consider a large matrix $A \in \mathbb{R}^{n \times d}$ with n rows and d columns
  - a sketch matrix B is one s.t. $B \in \mathbb{R}^{\ell \times d}$,
  - containing only $\ell \ll n$ rows and $A^T A \approx B^T B$.
- Especially useful when working with data streams.

Why would we need a sketch?

A range of common ML/DM tasks, e.g.

- Dimensionality reduction
- Clustering
- Classification
- Regression
- Signal denoising
- Approximate matrix multiplication
- Recommendation
- Reconstruction
- etc.

Content

- Background
- Related work
- Frequent directions
- Experiments and Results
- Conclusion

How do we get a sketch?

Three existing main classes:

- Random-projection: use a (random) projection matrix for dimensionality reduction
- Hashing: use a subspace embedding S that embeds the row space of the original matrix into a lower-dimensional subspace - in $O(\mathtt{nnz})$ time!
- Sampling: Column Subset Selection problem
  - select a set of rows directly, thus implicitly maintaining sparsity

How do we get a sketch? (cont.)

Proposed fourth approach, Frequent-directions

- $O(d\ell)$ space complexity, error decays proportionally to $1/\ell$,
- as opposed to the $1/\sqrt{\ell}$ error-decay of the existing approaches.
- Sketch update operations - per row in A - require amortised $O(d\ell)$ operations.

A novel *sampling* approach?


Anyone fancy a 'cool' PhD topic?

Item frequency estimation

- Used to uncover frequent items in an item stream
- (Re-)Invented (at least!) four times [6, 1, 3, 5][2]

Goal. Use $O(\ell)$ space as opposed to $O(d)$, where $\ell \ll d$

- to produce estimates g_i, s.t.
- $|f_i - g_i| \leq 2n/\ell$, for all item types i simultaneously.

Matrix setting. Use Frequent-directions to uncover any direction in space x

[2] [Misra and Gries, 1982; Demaine et al., 2002; Karp et al., 2003; Metwally, 2005]

Item frequency estimation

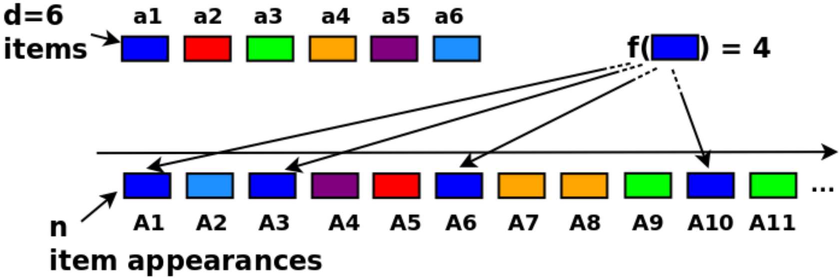
- The algorithm:

Repeat until there are less than ℓ unique items left {

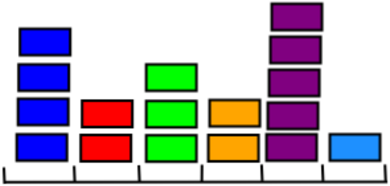
- Get item $A_t = a_i$ from stream, for $t = 1, \ldots, n$
- *If* a bucket/counter for item type i already exists
  - Increment f_i accordingly.
- *Else If* a free slot k exists, s.t. $1 \leq k \leq \ell$
  - Create new bucket/counter at position k for item type i, and
  - Store the item in f_i
- *Else*
  - Find median count $\delta = f_{\ell/2}$ of items, and
  - Remove exactly $\min(\delta, f_i)$ appearances from each bucket $i = 1, \ldots, \ell$

}

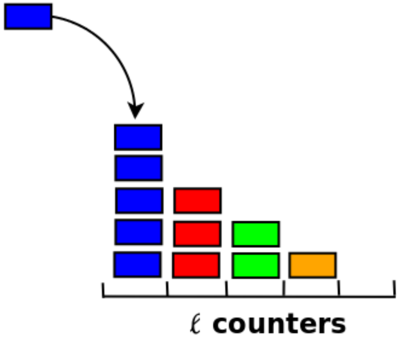
Item frequency estimation (cont.)



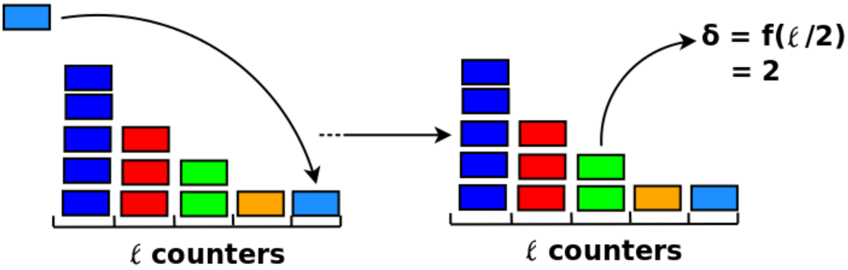
Item frequency estimation (cont.)



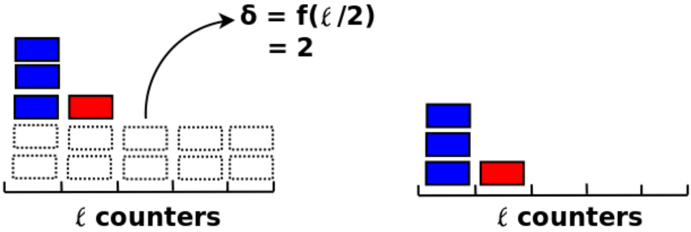
Item frequency estimation (cont.)



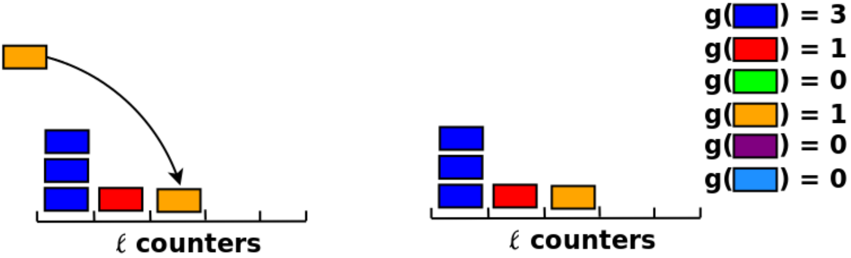
Item frequency estimation (cont.)



Item frequency estimation (cont.)



Item frequency estimation (cont.)



Claim. For each item type i,

- g_i is a good approximation for its true frequency f_i (even in the case of $g_i = 0$),
- 'Good': $|f_i - g_i| \leq 2n/\ell$.

Item frequency estimation (cont.)

Proof.

- Each item-type is deleted at most once per iteration:
  - $g_i \leq f_i$
- Each counter is decreased by at most δ_t at time t:
  - $g_i \geq f_i - \sum_t \delta_t \Leftrightarrow f_i - g_i \leq \sum_t \delta_t$
- Putting this together:
  - $0 \leq \sum_i g_i \leq \sum_t 1 - (\ell/2) \cdot \delta_t = n - (\ell/2) \cdot \sum_t \delta_t$
  - $\sum_t \delta_t \leq 2n/\ell \Leftrightarrow |f_i - g_i| \leq 2n/\ell$.

$\square$

Item frequency estimation (cont.)

- If one sets $\ell > 1/\varepsilon$,
  - Then any item that appears more than εn times in the stream must appear in the final sketch.
- Set $\ell = 2/\varepsilon$:
  - $|f_i - g_i| \leq \varepsilon n$.
- Further, if one takes k/ε instead of $2/\varepsilon$, one gets a rank-k approximation result![2]

Content

- Background
- Related work
- Frequent directions
- Experiments and Results
- Conclusion

The *Frequent-directions* algorithm

Represent the frequency of a direction (unit vector):

- Assume the directions of A are indicator vectors of the items:

$$A = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix}$$

- Frequency of second item $e_2 = (0, 1, 0, 0)^T$: $\|Ae_2\|^2 = \|(0, 1, 0, 1)^T\|^2 = 0^2 + 1^2 + 0^2 + 1^2 = 2$.
- Generalize the directions to unit vector $\{x : \|x\| = 1\}$ and the frequency of a direction is $\|Ax\|^2$.

The *Frequent-directions* algorithm (cont.)

Connection to SVD of A:

- $A = U\Sigma V^T \Leftrightarrow U^T A = \Sigma V^T \Leftrightarrow Au = \sigma v$.
- $\|Au\|^2 = \|\sigma v\|^2 = \sigma^2$.

Change u to x:

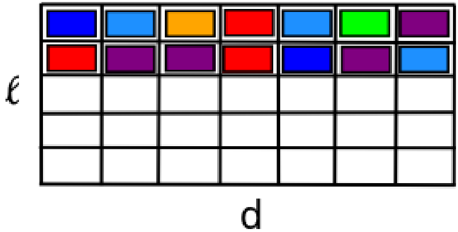
The frequency of a direction is indicated by the square of corresponding singular value σ^2.

The *Frequent-directions* algorithm (cont.)

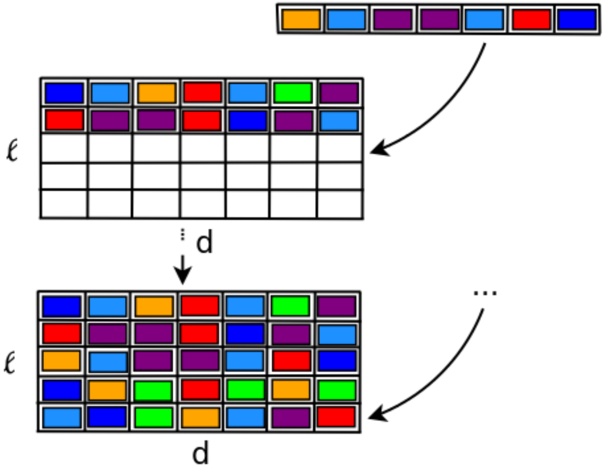
The algorithm:

Input: ℓ, $A \in \mathbb{R}^{n \times d}$
$B \leftarrow$ all zeros matrix $\in \mathbb{R}^{\ell \times d}$
for $i = 1, \ldots, n$ **do**
  Insert i^{th} row of A into zero valued row of B
  if B has no zero valued rows **then**
    $[U, \Sigma, V] \leftarrow SVD(B)$
    $\delta \leftarrow \sigma^2_{\ell/2}$
    $\check{\Sigma} \leftarrow \sqrt{\max(\Sigma^2 - I_\ell \delta, 0)}$
    $B \leftarrow \check{\Sigma} V^T$
  end if
end for

The *Frequent-directions* algorithm (cont.)

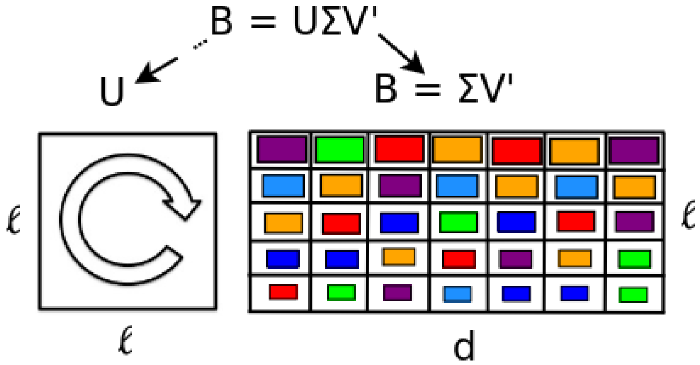


The *Frequent-directions* algorithm (cont.)



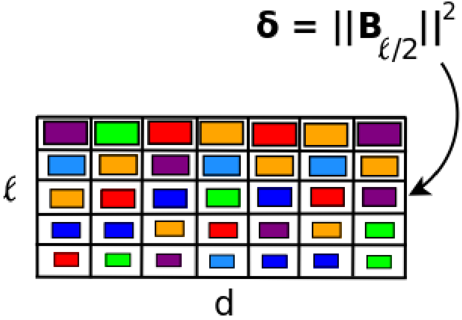
The *Frequent-directions* algorithm (cont.)

$[U, \Sigma, V] \leftarrow SVD(B)$.



The *Frequent-directions* algorithm (cont.)

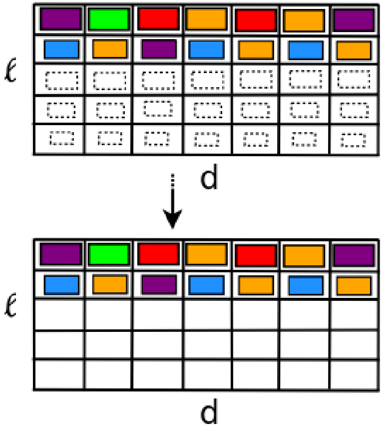
$\delta \leftarrow \sigma^2_{\ell/2}$.



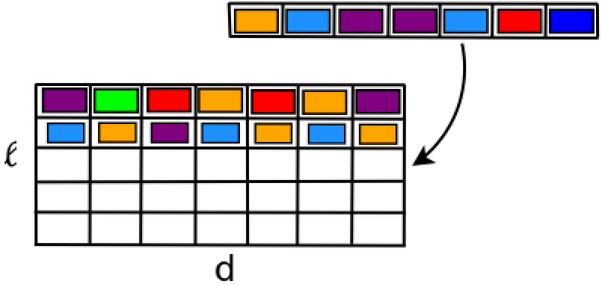
The *Frequent-directions* algorithm (cont.)

$\check{\Sigma} \leftarrow \sqrt{\max(\Sigma^2 - I_\ell \delta, 0)}$

$B \leftarrow \check{\Sigma} V^T$



The *Frequent-directions* algorithm (cont.)



Properties of the sketch matrix B

In summary:

- $A^T A \succeq B^T B \succeq 0$.
- $||A^T A - B^T B|| \leq 2||A||_f^2/\ell$.
- Let $A = [A_1; A_2]$ and B_1, B_2 is the sketches of A_1 and A_2. A sketch C of $B = [B_1; B_2]$ can be shown that:

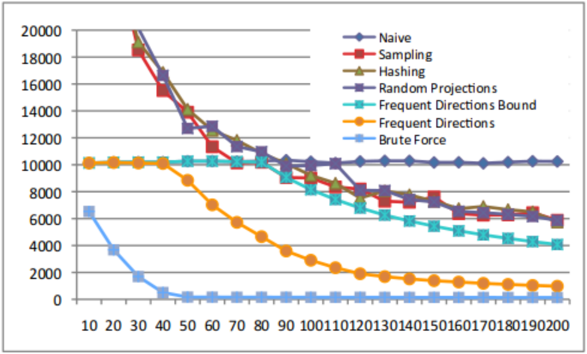
$$||A^T A - C^T C|| \leq 2||A||_f^2/\ell.$$

Content

- Background
- Related work
- Frequent directions
- Experiments and Results
- Conclusion

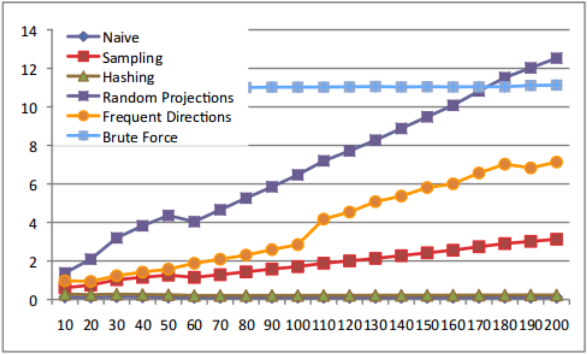
Experiments

For a synthetic matrix $n = 10000, d = 1000$. Error $||A^T A - B^T B||$ against sketch size ℓ with.



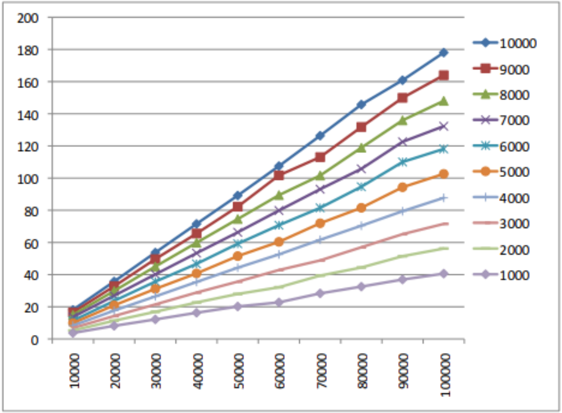
Experiments (cont.)

Running time.

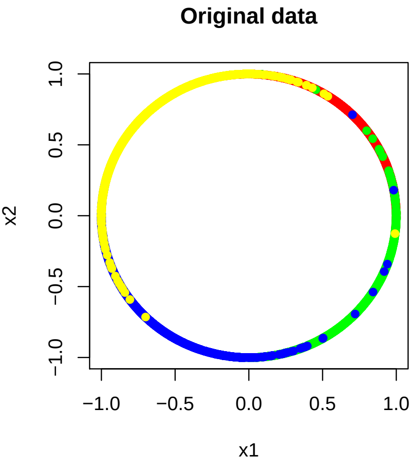


Experiments (cont.)

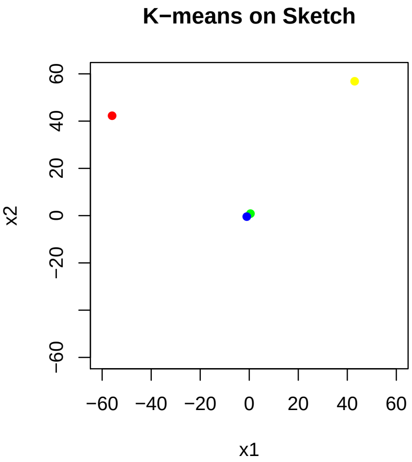
Linear in n and m (ℓ fix to 100).



Clustering experiment



Clustering experiment (cont.)



Clustering experiment (cont.)

$$B = \begin{pmatrix} 43.0030 & 56.8110 \\ -55.9340 & 32.3390 \\ 0.5011 & 0.8654 \\ -0.9427 & -0.3336 \\ 0 & 0 \\ \vdots & \vdots \end{pmatrix}$$

$$A^T A = \begin{pmatrix} 4979.0 & 75.7 \\ 75.7 & 5021.0 \end{pmatrix}, B^T B = \begin{pmatrix} 4979.0 & 75.6 \\ 75.6 & 5020.9 \end{pmatrix}$$

Conclusion

- In terms of $||A^T A - B^T B||$, the proposed sketching algorithm is more accurate than sampling, hashing and random projections.
- The proposed algorithm runs reasonably fast
  - in fact, faster than random projection, slower than sampling.
- The proposed algorithm is linear in the scale of the input size.
- Choose your sketching algorithm according to your task!

Thank you!

- Questions?

References

Erik D Demaine, Alejandro López-Ortiz, and J Ian Munro.
Frequency estimation of internet packet streams with limited space.
In *Algorithms-ESA 2002*, pages 348–360. Springer, 2002.

Mina Ghashami and Jeff M Phillips.
Relative errors for deterministic low-rank matrix approximations.
arXiv preprint arXiv:1307.7454, 2013.

Richard M Karp, Scott Shenker, and Christos H Papadimitriou.
A simple algorithm for finding frequent elements in streams and bags.
ACM Transactions on Database Systems (TODS), 28(1):51–55, 2003.

Edo Liberty.
Simple and deterministic matrix sketching.
In *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 581–588. ACM, 2013.

References (cont.)

Ahmed Metwally, Divyakant Agrawal, and Amr El Abbadi.
Efficient computation of frequent and top-k elements in data streams.
In *Database Theory-ICDT 2005*, pages 398–412. Springer, 2005.

Jayadev Misra and David Gries.
Finding repeated elements.
Science of computer programming, 2(2):143–152, 1982.

Roman Vershynin.
Spectral norm of products of random and deterministic matrices.
Probability Theory and Related Fields, 150(3-4):471–509, 2011.