

**IE531: Algorithms for Data Analytics**  
**Spring, 2018**  
**Programming Assignment 5: Edo Liberty's Matrix**  
**Sketching Algorithm**  
**Due Date: April 27, 2018**  
©Prof. R.S. Sreenivas

For this assignment you are going to implement Edo Liberty's Matrix Sketching algorithm [1]. Although this procedure was inspired by the family of *Misra-Gries Streaming Algorithms*, you do not have to implement it as such. That is, a non-streaming version should be fine.

You should take a look at Dr. Liberty's [talk \(click here\)](#) at the Simons Institute to get at a [possible implementation-strategy](#) for this assignment. Essentially we have a (large) data-matrix  $\mathbf{A} \in \mathcal{R}^{m \times n}$ , where  $n \gg m$ <sup>1</sup>. The *NEWMAT library* has a robust and fast-implementation of the *SVD-Decomposition* algorithm – `SVD(A, D, U, V)`; – where  $\mathbf{A} = \mathbf{U} \times \mathbf{D} \times \mathbf{V.t}()$ . As you can gather from the [on line reference](#), NEWMAT assumes  $\mathbf{A}$  is an  $m \times n$  matrix where  $m \geq n$ . If  $n > m$ , then you have to `work with A.t()`, and `make appropriate changes to U and V` to make things work.

I have provided a hint in `hint.cpp` on Compass. Keep in mind that I tend to have my datasets appear in `columnar-format`. That said, you have to figure out the implementation details yourself. I have also provided two datasets on Compass. Sample outputs (using these two datasets) are shown in figure 1.

You have to take a measured-and-slow approach to getting this assignment done. The `first-part` involves understanding Dr. Liberty's algorithm. Following this, you have to think about a `slightly-different implementation` (from what is in his talk) – `and this has to do with zeroing-out just the lowest-singular value` (as opposed to everything below the median-singular-value). Then, there is the issue with getting NEWMAT's SVD to do the needful. In the midst of all this, there are routine-things like computing the `Frobenius Norm` of a matrix (which BTW, can also be computed using its *Singular Values*). As you can see from the Frobenius Norms of the Original- and Sketch-Matrices in figure 1, my implementation comfortably meets the  $\epsilon$ -specification (i.e. the reduction is Frobenius-Norm is not much at all).

## References

- [1] Edo Liberty. Simple and deterministic matrix sketching. In *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '13, pages 581–588, New York, NY, USA, 2013. ACM.

---

<sup>1</sup>At least, as I like to think about it. Keep in mind that some authors may have more rows in the data-matrix than columns (i.e.  $m \gg n$ ), you need to get used to these things! Nothing that “transposing of matrices” cannot fix! Liberty's paper uses the row-version, while his talk uses the column-version of his algorithm.

```

MacBook-Air:Debug sreenivas$ ./Matrix\ Sketch 50 1000 0.09 matrix_sketch1 matrix_sketch1_out
Edo Liberty's Matrix Sketching Algorithm
=====
Original Data-Matrix has 50-rows & 1000-cols
Epsilon = 0.09 (i.e. max. of 9% reduction of Frobenius-Norm of the Sketch Matrix)
Input File = matrix_sketch1
Frobenius Norm of the (50 x 1000) Data Matrix = 12569.8
Frobenius Norm of the (50 x 23) Sketch Matrix = 12374.5
Change in Frobenius-Norm between Sketch & Original = -1.55%
File 'matrix_sketch1_out' contains a (50 x 23) Matrix-Sketch
MacBook-Air:Debug sreenivas$ ./Matrix\ Sketch 50 1000 0.1 matrix_sketch1 matrix_sketch1_out
Edo Liberty's Matrix Sketching Algorithm
=====
Original Data-Matrix has 50-rows & 1000-cols
Epsilon = 0.1 (i.e. max. of 10% reduction of Frobenius-Norm of the Sketch Matrix)
Input File = matrix_sketch1
Frobenius Norm of the (50 x 1000) Data Matrix = 12569.8
Frobenius Norm of the (50 x 20) Sketch Matrix = 12345.2
Change in Frobenius-Norm between Sketch & Original = -1.79%
File 'matrix_sketch1_out' contains a (50 x 20) Matrix-Sketch
MacBook-Air:Debug sreenivas$ ./Matrix\ Sketch 9 329 0.2 Data x
Edo Liberty's Matrix Sketching Algorithm
=====
Original Data-Matrix has 9-rows & 329-cols
Epsilon = 0.2 (i.e. max. of 20% reduction of Frobenius-Norm of the Sketch Matrix)
Input File = Data
Frobenius Norm of the (9 x 329) Data Matrix = 32803.6
Frobenius Norm of the (9 x 9) Sketch Matrix = 32803.6
Change in Frobenius-Norm between Sketch & Original = 2.22e-14%
File 'x' contains a (9 x 9) Matrix-Sketch
MacBook-Air:Debug sreenivas$ █

```

Figure 1: An illustration of the command-line variables.