Dynamic Matrix Approximations in the Data Streaming Scenario

Adam Kirsch
Mentors: G. Cormode and S. Muthukrishnan

DIMACS REU 2003

What's the "Data Streaming Scenario?"

- Processing enormous quantities of data:
  - Can't afford linear space in memory.
  - Can't even afford linear space on disk.
- Data is changing with time:
  - Dynamic updates come in arbitrary order and types.
  - We are only interested in functions of the current state of the data.

A Classical Data Streaming Problem

Scenario: A vector $V[1, \ldots, n]$, initially $\vec{0}$, is dynamically updated by increments and decrements to its entries. In other words, at time $t \in \mathbb{N}$, we add c_t to $V[i_t]$, where $1 \leq i_t \leq n$ and $c_t \in \mathbb{R}$.

Task: Design an $O(\log n)$-space data structure that can give approximate answers to point queries (queries of the form, "What is the value of $V[i]$?").

A Randomized Approach (Cormode and Muthukrishnan, 2003)

Solution: "Randomly" assign each coordinate of V to one of B buckets. When performing an update to V, update the appropriate bucket. When answering a point query, return the value stored in the appropriate bucket.

Analysis: The expected error in our response is at most $B||V||_1$. By Markov's Inequality, the probability that our error is at least $4B||V||_1$ is at most $1/4$. To lower the error probability exponentially, just run Solution 1 R independent times, updating each as before, and answering point queries by returning the median of the R independent trials.

On Randomness

- We need to store each assignment of coordinates to buckets, so we need to use a hash function for each assignment. We'll draw the hash functions uniformly at random from some family to ensure that each of the R trials is independent.

- We pay dearly for the amount of independence between the buckets where different coordinates are hashed. We need at least pairwise independence for our analysis to go through. We can afford that, but not much more.

General Observations

1. We can approximate big entries (those with a substantial contribution to $||V||_1$) well, but small entries are blurred together.

2. Randomness is very helpful, but if we have to keep track of random bits, we use up our very limited space.

Matrix Approximations

Task: Approximate an $n \times n$ matrix A by some $\hat{A}$ in the data streaming scenario.

Solution: It depends what we mean by "approximate." Are we really interested in the matrix as an operator, or are we thinking of it as a vector in the space of $n \times n$ matrices (i.e. as a table of values).

Competing Notions of Matrix Approximation

- If we want a good entry-wise approximation, we should try to minimize $||A - \hat{A}||_F$, where $||\cdot||_F$ is the *Frobenius matrix norm*:

$$||B||_F = \sqrt{\sum_{i=1}^{n} \sum_{j=1}^{n} b_{i,j}^2}$$

- But if we are interested in A as an operator, we should seek to minimize $||A - \hat{A}||_2$, where $||\cdot||_2$ is the *spectral matrix norm*:

$$\begin{aligned} ||B||_2 &= \max_{|x|=1} |Bx| \\ &= \max\{\sqrt{\lambda} : \lambda \text{ is an eigenvalue of } B^T B\} \end{aligned}$$

Why does the choice of norm matter?

- The Frobenius norm reflects the *average* linear trend of a matrix.

- The spectral norm captures the *largest* such trend.

- In many applications, we are interested in the subspace where A is "most active" (the subspace spanned by the eigenvectors corresponding to the largest, say k, eigenvalues).

Consequence: Frobenius norm approximations usually have to be extremely accurate (prohibitively so) to preserve spectral structure. Unfortunately, it's a lot easier to produce a good Frobenius norm approximation than a good spectral norm one.

Spectral Norm Approximations: State of the Art

- Use a random procedure that gives $E[\hat{a}_{i,j}] = a_{i,j}$ for all i, j.

- Design the procedure as to minimize $\max_{i,j} \mathbf{Var}[\hat{a}_{i,j} - a_{i,j}]$.

- Keep the $\hat{a}_{i,j}$'s independent. If A is known to be symmetric, keep $\hat{A}$ symmetric, and keep its entries from the main diagonal upwards independent.

Then applying a result from the theory of random matrices, we have that $||A - \hat{A}||_2$ is "small" with high probability.

Shortcomings

- It's too expensive to keep the entries of $\hat{A}$ independent, because we can't afford to retain too much randomness from step to step, and updates can come arbitrarily.

- The only practical such scheme seems to be random sampling (keep a small subset of the entries of A exactly, and assume the rest are 0), and this yields bad results.

- Bounding $||A - \hat{A}||_2$ only gives nice results when A is symmetric. If A is merely diagonalizable, we need information about its eigenvectors. If A is not diagonalizable, life is hard.

I Lied to You

- In most applications, the data is described in terms of an $n \times m$ matrix A, and we are interested in preserving the spectral structure of $A^T A$.

- Approximating A with $\hat{A}$ and then working with $\hat{A}^T \hat{A}$ is problematic, because $\hat{A}^T \hat{A}$ might not be sparse even if A is sparse, and the error of the approximation of A is essentially squared.

- Thus, we seek an approximation $\hat{A}$ of A that somehow preserves the *singular value decomposition* (SVD) of A.