Sketching as a Tool for Numerical Linear Algebra

David Woodruff
IBM Almaden

Talk Outline

- Exact Regression Algorithms
- Sketching to speed up Least Squares Regression
- Sketching to speed up Least Absolute Deviation (l_1) Regression
- Sketching to speed up Low Rank Approximation

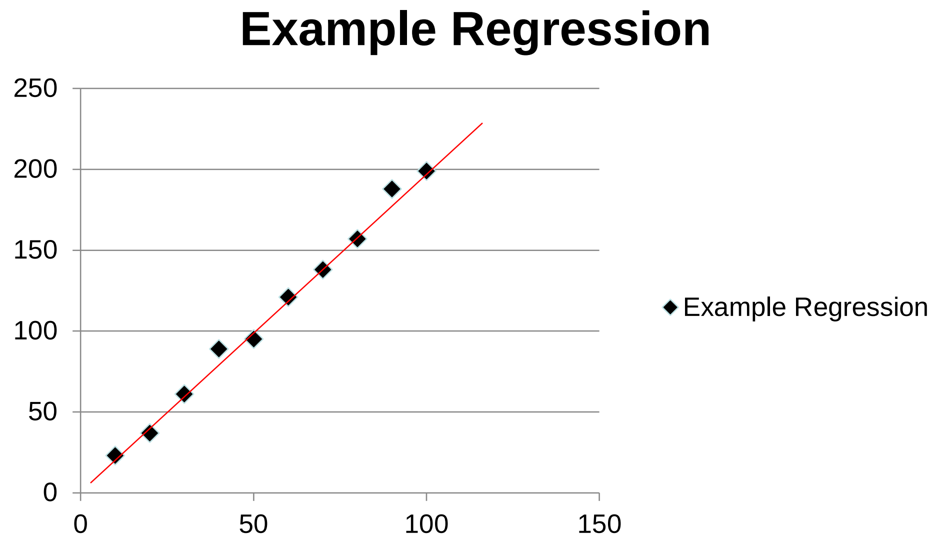
Regression

Linear Regression

- Statistical method to study linear dependencies between variables in the presence of noise.

Example

- Ohm's law V = R · I
- Find linear function that best fits the data



Regression

Standard Setting

- One measured variable b
- A set of predictor variables $a_1 ,\ldots, a_d$
- Assumption:

$$b = x_0 + a_1 x_1 + \ldots + a_d x_d + \varepsilon$$

- ε is assumed to be noise and the x_i are model parameters we want to learn
- Can assume $x_0 = 0$
- Now consider n observations of b

Regression analysis

Matrix form

Input: $n \times d$-matrix A and a vector $b=(b_1,\ldots, b_n)$
n is the number of observations; d is the number of predictor variables

Output: x^* so that Ax^* and b are close

- Consider the over-constrained case, when n À d
- Can assume that A has full column rank

Regression analysis

Least Squares Method

- Find x* that minimizes $|Ax-b|_2^2 = \Sigma (b_i - \langle A_{i*}, x \rangle)^2$
- A_{i*} is i-th row of A
- Certain desirable statistical properties
- Closed form solution: $x = (A^T A)^{-1} A^T b$

Method of least absolute deviation (l_1 -regression)

- Find x* that minimizes $|Ax-b|_1 = \Sigma |b_i - \langle A_{i*}, x \rangle|$
- Cost is less sensitive to outliers than least squares
- Can solve via linear programming

Time complexities are at least $n*d^2$, we want better!

Talk Outline

- Exact Regression Algorithms
- Sketching to speed up Least Squares Regression
- Sketching to speed up Least Absolute Deviation (l_1) Regression
- Sketching to speed up Low Rank Approximation

Sketching to solve least squares regression

- How to find an approximate solution x to $\min_x |Ax-b|_2$?

- Goal: output x‘ for which $|Ax'-b|_2 \cdot (1+\varepsilon) \min_x |Ax-b|_2$ with high probability

- Draw S from a k x n random family of matrices, for a value k << n

- Compute S*A and S*b

- Output the solution x‘ to $\min_{x'} |(SA)x-(Sb)|_2$

How to choose the right sketching matrix S?

- Recall: output the solution x' to $\min_{x'} |(SA)x-(Sb)|_2$
- Lots of matrices work
- S is d/ε^2 x n matrix of i.i.d. Normal random variables
- Computing S*A may be slow…

How to choose the right sketching matrix S? [S]

- S is a Johnson Lindenstrauss Transform
  - S = P*H*D
  - D is a diagonal matrix with +1, -1 on diagonals
  - H is the Hadamard transform
  - P just chooses a random (small) subset of rows of H*D
  - S*A can be computed much faster

Even faster sketching matrices [CW,MM,NN]

- CountSketch matrix
- Define k x n matrix S, for $k = d^2/\varepsilon^2$
- S is really sparse: single randomly chosen non-zero entry per column

$$\begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & 1 & 0 & -1 & 0 \\ 0 & -1 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

Surprisingly,

this works!

Talk Outline

- Exact Regression Algorithms
- Sketching to speed up Least Squares Regression
- Sketching to speed up Least Absolute Deviation (l_1) Regression
- Sketching to speed up Low Rank Approximation

Sketching to solve l_1-regression

- How to find an approximate solution x to $\min_x |Ax-b|_1$?
- Goal: output x‘ for which $|Ax'-b|_1 \cdot (1+\varepsilon) \min_x |Ax-b|_1$ with high probability
- Natural attempt: Draw S from a k x n random family of matrices, for a value k << n
- Compute S*A and S*b
- Output the solution x‘ to $\min_{x'} |(SA)x-(Sb)|_1$
- Turns out this does not work!

Sketching to solve l_1-regression [SW]

- Why doesn't outputting the solution x' to $\min_{x'}$ $|(SA)x-(Sb)|_1$ work?

- Don't know of k x n matrices S with small k for which if x' is solution to $\min_x |(SA)x-(Sb)|_1$ then

$$|Ax'-b|_1 \cdot \ (1+\varepsilon) \min_x |Ax-b|_1$$

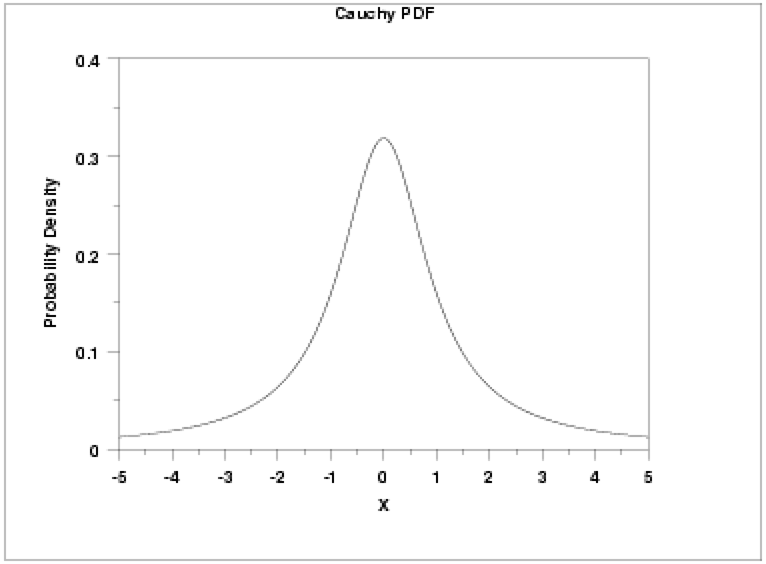
  with high probability

- Instead: can find an S so that

$$|Ax'-b|_1 \cdot \ (d \log d) \min_x |Ax-b|_1$$

- S is a matrix of i.i.d. Cauchy random variables

Cauchy random variables



- Cauchy random variables not as nice as Normal (Gaussian) random variables

- They don't have a mean and have infinite variance

- Ratio of two independent Normal random variables is Cauchy

Sketching to solve l_1-regression

- How to find an approximate solution x to $\min_x |Ax-b|_1$?

- Want x‘ for which if x‘ is solution to $\min_x |(SA)x-(Sb)|_1$, then $|Ax'-b|_1 \cdot\ (1+\varepsilon) \min_x |Ax-b|_1$ with high probability

- For d log d x n matrix S of Cauchy random variables:

$$|Ax'-b|_1 \cdot\ (d \log d) \min_x |Ax-b|_1$$

- For this “poor” solution x’, let b’ = Ax’-b

- Might as well solve regression problem with A and b’

Sketching to solve l_1-regression

- Main Idea: Compute a QR-factorization of S*A
- Q has orthonormal columns and Q*R = S*A
- $A*R^{-1}$ turns out to be a “well-conditioning” of original matrix A
- Compute $A*R^{-1}$ and sample $d^{3.5}/\varepsilon^2$ rows of $[A*R^{-1}, b']$ where the i-th row is sampled proportional to its 1-norm
- Solve regression problem on the (reweighted) samples

Sketching to solve l_1-regression [MM]

- Most expensive operation is computing S*A where S is the matrix of i.i.d. Cauchy random variables
- All other operations are in the "smaller space"
- Can speed this up by choosing S as follows:

$$\begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & 1 & 0 & -1 & 0 \\ 0 & -1 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} ¢ \begin{bmatrix} C_1 & & & & \\ & C_2 & & & \\ & & C_3 & & \\ & & & \cdots & \\ & & & & C_n \end{bmatrix}$$

Further sketching improvements [WZ]

- Can show you need a fewer number of sampled rows in later steps if instead choose S as follows

- Instead of diagonal of Cauchy random variables, choose diagonal of reciprocals of exponential random variables

$$\begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & 1 & 0 & -1 & 0 \\ 0 & -1 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \text{¢} \begin{bmatrix} 1/E_1 & & & & \\ & 1/E_2 & & & \\ & & 1/E_3 & & \\ & & & \cdots & \\ & & & & 1/E_n \end{bmatrix}$$

Talk Outline

- Exact regression algorithms
- Sketching to speed up Least Squares Regression
- Sketching to speed up Least Absolute Deviation (l_1) Regression
- Sketching to speed up Low Rank Approximation

Low rank approximation

- A is an n x n matrix

- Typically well-approximated by low rank matrix
  - E.g., only high rank because of noise

- Want to output a rank k matrix A', so that

$$|A-A'|_F \cdot \ (1+\varepsilon)\ |A-A_k|_F,$$

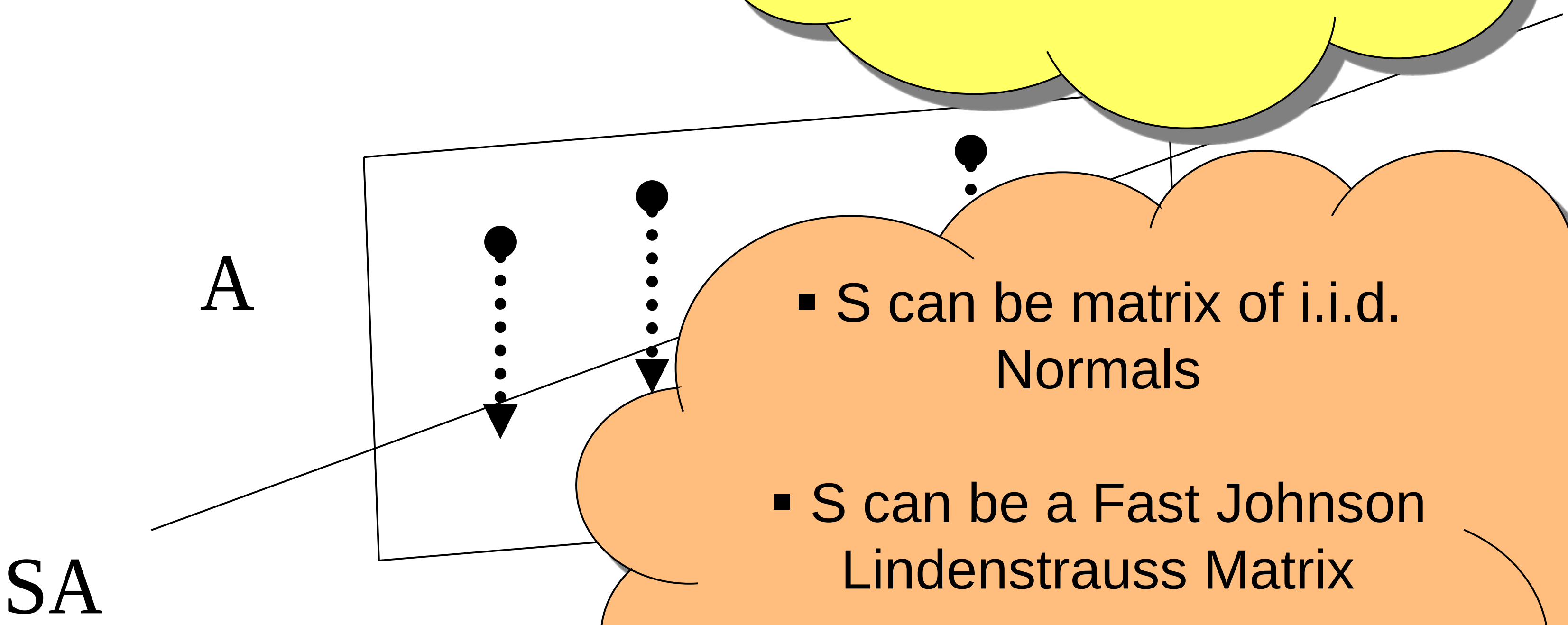
w.h.p., where $A_k = \text{argmin}_{\text{rank } k \text{ matrices } B} |A-B|_F$

- For matrix C, $|C|_F = (\Sigma_{i,j} C_{i,j}^2)^{1/2}$

Solution to low-rank approxim

- Given n x n input matrix
- Compute S*A using a ske rows. S*A takes random A

Most time-consuming step is computing S*A



- S can be matrix of i.i.d. Normals
- S can be a Fast Johnson Lindenstrauss Matrix
- S can be a CountSketch matrix

- Project rows of A or approximation to points i

Conclusion

- Gave fast sketching-based algorithms for
  - Least Squares Regression
  - Least Absolute Deviation (l_1) Regression
  - Low Rank Approximation
- Sketching also provides “dimensionality reduction”
  - Communication-efficient solutions for these problems