Manuscript on Stable Distributions, Pseudorandom Generators, Embeddings, and Data Stream Computation

Introduction

One problem with big data streams is that any computation, which uses the entire data set, would need a large amount of bits and could not be handles by a single computer within reasonable time. On solution would be to let multiple computers work on the same calculation at the same time and then put together the calculations of each computation. We could also try to use only a part of the data set, that represents the data well enough or ad additional calculations to make sure the partial data is representative of the entire data set. But how can we manipulate the partial data so that it will represent the complete data set well? And how much bits would the calculation still need? If we do nothing and always use the entirety of the data, we have a linear relationship between the number of tuples and bits needed for the computation and for the storage. The ultimate (so far unattainable) goal would be a logarithmic correlation. Thus, we are in search of an algorithm that allows us to build a satisfactory approximation of a desired calculations, which uses less bits than without the algorithm.

In this paper Indyk finds an algorithm that satisfies these terms for $p \in (0, 2]$. His proposed computation only uses $\mathcal{O}(\log n/\varepsilon^2)$ words of storage. Previous scholars have also come up with algorithms for this problem, but not for all $p \in (0, 2]$. To do so Indyk picks up on Feigenbaum's usage of sketches to approximate the sum of all attributes of all integers and generalizes an earlier algorithm of Wasserman and Blum, 1997, which worked for the case p=2. Before that we take a quick detour and have a look at the algorithm for $p = 2$ that Alon et al., 1996, came up with. First, I will give you a few definitions and some background on Alon et al. and Feigenbaum et al. and then dig into the main arguments of Indyk's paper.

Definition 1. A distribution is p-stable if for all X, Y, Z random variable drawn from this distribution, $aX + bY$ and $(|a|^p + |b|^p)^{1/p}Z$ are identically distributed.

Example 2. Gaussian (normal) distribution is p-stable. To show this, let $a \in \mathbb{R}^n$, and $X_1, ..., X_n$ be i.i.d. samples from $\mathcal{N}(0, 1)$. Then $\sum_{i=1}^{n} a_i X_i$ is a linear combination of the i.i.d. Gaussian variables and is itself Gaussian. As a sum of independent Gaussians is a Gaussian as well. To proof that the

Gaussian distribution is 2-stable, we will look at the calculation of the mean and the variance.

$$\mathbb{E}\left[\sum_{i=1}^{n} a_i X_i\right] = \sum v_i \mathbb{E}[g_i] = 0,$$

$$\mathbb{E}\left[\left(\sum_{i=1}^{n} a_i X_i\right)^2\right] = \sum_{i\neq j} \mathbb{E}\left[a_i a_j X_i X_j\right] + \sum_{i=1}^{n} \mathbb{E}\left[a_i^2 X_i^2\right] = \sum_{i\neq j} a_i a_j \mathbb{E}[X_i]\mathbb{E}[X_j] + \sum_i a_i^2 = 0 + \|a\|_2^2.$$

Thus the linear combination $\sum_{i=1}^{n} a_i X_i$ is $\mathcal{N}(0, \|a\|_2^2)$ and therefore identically distributed as a standard Gaussian random variable multiplied by the square root of the variance, i.e. by $\|a\|_2 = \left(\sum_{i=1}^{n} a_i^2\right)^{1/2}$.

The main object of the stream computations consists of the data stream itself, which we could map in a form of a matrix with two columns for better understanding. In reality the algorithm sees it as a sequence of pairs $< coordinate, update >$.

Definition 3. A stream S of data consists of pairs (i, a), where i is the "identifier" and a is the "attribute"; $i \in [n] = \{0, \ldots, n-1\}$, $a \in \{-M, -M+1, \ldots, 0, \ldots, M-1, M\}$.

Example 4. An example of a stream is the following object:

$$S = \begin{pmatrix} 1 & 10 \\ 1 & 20 \\ 2 & 12 \\ 2 & 23 \end{pmatrix},$$

where i represents a specific person and a this person's bank account balance(s). Here we would see person 1 and person 2, where person 1 has 10 CHF on their daily allowance account and 20 CHF savings account and person 2 has 12 CHF and 23 CHF.

Definition 5. The l_p-norm[1] of the stream S is defined by $L_p(S) = \|V(S)\|_p$, where $V(S)_i = \sum_{(i,a)\in S} a$.

Example 6. Referring to Example 4, the l_1-norm of the stream can be computed as follows:

$$V(S)_1 = 10 + 20 = 30$$
$$V(S)_2 = 12 + 23 = 35$$
$$L_1(S) = 30 + 35 = 65,$$

The problem is that the exact computation of $L_p(S)$ is expensive in space needed ($\mathcal{O}(M)$). Are there more efficient ways? Or even simple algorithms to approximate the quantity? The idea is to generate a random vector r that follows a p-stable distribution. It is then known that for a vector $u \in \mathbb{R}^n$, $r \cdot u \sim \|u\|_p$ such that we can approximate $L_p(S) = \|V(S)\|_p$ by $r \cdot V(S)$.

As in sub-linear space finding a deterministic and/or exact algorithm to solve this problem is proven to be impossible, we will have to settle a randomized approximation algorithm with $\delta, \varepsilon > 0$. Where we want to keep the decrease in the likelihood of our approximation being close to the real value, $1 - \delta$, and the multiplicative approximation, $1 + \varepsilon$, as small as possible. But keep in mind that, the smaller δ and ε are, the bigger k will get and so will the computing effort.

[1]The function $\|\cdot\|$ is a norm if the following three conditions are satisfied. (1) $\|x\| \geq 0$ and $\|x\| = 0$ if and only if $x = 0$; (2) $\|\alpha x\| = |\alpha|\|x\|$; (3) $\|x + y\| \leq \|x\| + \|y\|$.

Previous Findings

Alon et al. 1996, proposed a randomized scheme for approximating $L_2(S)$ using $\mathcal{O}(1/\varepsilon^2)$ integers, each $\mathcal{O}(\log(n+M))$-bits long.

Feigenbaum et al. 1999, proposed a different algorithm for estimating $L_1(S)$. Their algorithm works in a restricted setting, where for each i (identifier), the stream contains at most two pairs (i,a), i.e. only two attributes to i where one is positve and one negative. This should be overcome by the algorithm presented in this paper. There are two ways to view their result:

- Assume two streams S_r (red) and S_b (blue) containing at most one attribute to each index (else they are separated in red and blue).
- Compute sketches $C(S_r)$ and $C(S_b)$ of small size such that $L_p(S_r,S_b) = \sum_{i=1}^{n} \left| \sum_{(i,a)\in S_r} a - \sum_{(i,a)\in S_b} a \right|$ can be quickly evaluated from $C(S_r)$ and $C(S_b)$ by applying some function F.

In Alon et al., they proceed the following way:

- Draw a random vector r from a 4-wise independent family.
- Compute $s = V(S)\cdot r$ (determinant product).
- They showed that the second moment of s is equal to $L_2(S)^2 = \|V(S)\|_2^2$.
- Advantage: 4-wise independent family
- Disadvantage: Only works for $p=2$ and it is not clear how to generalize to $L_p(S)$, $p<2$.

In Feigenbaum et al. (Dimension Reduction), the streams S_r and S_b can be viewed as points in n-dimensional space and $L_p(S_r,S_b)$ is the l_p-distance between the points. Then the sketch operator C is $C : l_p^n \to C_i$ maps the points into the "sketch space" such that

- each point in C_i can be described using only m numbers,
- $L_p(S_r,S_b) \approx F(C(S_r),C(S_b))$, for a function F.

But (C_i,F) is not a normed space, i.e. F not a norm. E.g. for l_1, $F((x_1,...,x_m),(y_1,...,y_m)) = \text{median}(|x_1-y_1|,...,|x_m-y_m|)$. To overcome this, we observe that for l_2, that if we modify our algorithm by replacing the median by $\|\cdot\|_2$, the accuracy of the estimation does not change.

Preliminaries

Definition 7. A distribution $\mathcal{D}$ (e.g. Gaussian/Cauchy) over $\mathbb{R}$ is p-stable if for all random variables $X_1,...,X_n$ drawn from $\mathcal{D}$, the linear combination $\sum_{i=1}^n a_i X_i$ is identically distributed as $(\sum_{i=1}^n |a_i|^p)^{1/p} X$, where X is also drawn from $\mathcal{D}$.

Lemma 8. *For all $p \in]0,2]$, there exists a p-stable distribution $\mathcal{D}$.*

Example 9. The Cauchy distribution $\mathcal{D}_C$ defined by the density $f_C(x) = 1/\pi 1/(1+x^2)$ is 1-stable. The Gaussian (normal) distribution $\mathcal{D}_G$ defined by the density $f_G(x) = 1/\sqrt{2\pi}e^{-x^2/2}$ is 2-stable.

Generally, a random variable of a p-stable distribution can be generated by

$$X = \frac{\sin(p\Theta)}{\cos(\Theta)^{1/p}} \left(\frac{\cos(\Theta(1-p))}{-\log(r)} \right)^{\frac{1-p}{p}},$$

where Θ and r follow an uniform distribution over the intervals $[-\pi/2,\pi/2]$ and r over $[0,1]$, respectively.

Approximation of the L_p-Norm for data streams

Let S be a data stream containing pairs (i, a), $i \in [n]$ and $a \in \{-M, ..., M\}$. An algorithm for approximating $L_1(S)$ with restrictions:

- Assumes infinite precision of the calculations, i.e. uses arithmetic operations on $\mathbb{R}$ and not on computed numbers.
- Although it uses only $\mathcal{O}(1/\varepsilon^2)$ words of storage, it performs random access to as many as $\Theta(n)$ random numbers. Thus, a natural implementation of the algorithm would request $\Theta(n)$ storage!

For these restrictions ways to remove them will be proposed and proven by Indyk later in the Paper. So, let as take a look at an ideal algorithm, where $l = c/\varepsilon^2 \log(1/\delta)$ for a constant c, which will be specified later. The algorithm proceeds as follows:

1. Initialize $n \cdot l$ independent random variables X_i^j, $i \in [n]$ and $j \in [l]$, drawn from the Cauchy distribution. Set $S^j = 0$, for $j \in [l]$.
2. For each new pair $(i, a) \in S$, perform $S^j = S^j + aX_i^j$, for all $j \in [l]$.
3. Return the median of $(|S^0|, ..., |S^{l-1}|)$ as result $A(S)$ of the algorithm A.

The correctness of the algorithm is given by the following. Let $c_i = \sum_{(i,a)\in S} a$ such that $L_1(S) = C = \sum_{i=1}^n |c_i|$.

Claim 10. Each S^j has the same distribution as CX, where X is a random variable drawn from the Cauchy distribution (because the Cauchy distribution is 1-stable).

Lemma 11. *Let X be Cauchy distributed. Then, median$(|X|) = 1$. Therefore, median$(|X|) = a$, $a > 0$ (calculate on basis of Cauchy density).*

Claim 12. For all distributions $\mathcal{D}$ on $\mathbb{R}$ with cumulative distribution function F, take $l = c/\varepsilon^2 \log(1/\delta)$ independent samples $X_0, ..., X_{l-1}$ of $\mathcal{D}$, and also let $X = \text{median}(X_0, ..., X_{l-1})$. Then, for suitable C,

$$\mathbb{P}\left[F(X) \in [1/2 - \varepsilon, 1/2 + \varepsilon]\right] > 1 - \delta,$$

which is called the Chernoff-bound.

Lemma 13. *Let F be the cumulative distribution function of $|X|$, where X is Cauchy distributed. Let $z > 0$ such that $F(z) \in [1/2 - \varepsilon, 1/2 + \varepsilon]$, then for ε small enough, $z \in [1 - 4\varepsilon, 1 + 4\varepsilon]$. (This follows from the boundedness of $F^{-1}(x) = \tan(x\pi/2)$ around point $1/2$).*

Theorem 14. *The "ideal" algorithm correctly estimates $L_1(S)$ up to the factor $1 \pm \varepsilon$ with probability of at least $1 - \delta$, i.e.*

$$\mathbb{P}\left[A(S) \geq (1 - \varepsilon)L_1(S)\right] \geq 1 - \delta$$
$$\mathbb{P}\left[A(S) \leq (1 + \varepsilon)L_1(S)\right] \geq 1 - \delta$$

Bounded Precision

So far we worked under the assumption that our numbers have infinite precision. But we know that computer generated and computer stored numbers are limited in their precision. Since the attributes in data streams are integers, we only have to take a closer look at the random variables X_i^j. We need to show that the computed numbers are precise enough to represent $\mathcal{O}(log(n + M))$ bits.

Indyk shows that $\forall i, j$ a general approximation $\tilde{X}_i^j$ to each X_i^j exists so that for each of them $|\tilde{X}_i^j - X_i^j| \leq \alpha$ with probability $1 - p$ applies, only using $\mathcal{O}\left(\log\left(\frac{1}{p} + \frac{1}{\alpha}\right)\right)$ bits.

Randomness Reduction

The remaining issue is that we still need $\mathcal{O}(n)$ memory words to make sure that if we access a specific X_i^j multiple times, its value is always the same. Indyk suggest the employing the following algorithm to avoid this.

As R_0 is a seed[2], we can use it to generate the matrix to access a specific X_i^j multiple times. For example, if we wanted to generate it column by column we could set the first few entries of the first column according to R_0 and the next few entries according to R_1 and so on. Given i and j, we can easily find which bit of which R_t is used to generate the entry a_{ij}. Hence, we are able to re-compute the ith column of the matrix simply by starting the pseudorandom generator all over from R_0 and compute the appropriate R_t 's for the desired column. This as described, maybe need a lot of time, as we need to make about nk steps of the pseudorandom generator for a typical column, but this algorithm needs practically no extra memory.

Computing $L_2(S)$

So far, we tried to estimate $L_1(S)$. Now, we try to compute $L_2(S)$, where $p = 2$. Sketch of the stream computed by $y = AV(S)$, with an implicitly defined matrix A and $L_2(S) = \|V(S)\|_2$ is estimated by $\|y\|_2$. In other words, the algorithm provides a streaming version of the dimensionality theorem by Johnson and Lindenstrauss. The first algorithm is obtained by replacing the Cauchy distribution by the Gaussian distribution. As before, the final estimator is a median of $|S^0|, ..., |S^{l-1}|$. Lemma 13 still holds since Gaussian density is differentiable around the median and therefore bounded. Moreover, Gaussian random variable can be generated from uniform distribution. Then, one can verify that claim 12 holds with $B = 1/p^{\mathcal{O}(1)}$.

Theorem 15. *There exists an algorithm such that for all $0 < \varepsilon, \delta < 1$, $L_1(S)$ or $L_2(S)$ is estimated correctly up to the factor $(1 \pm \varepsilon)$ with probability of at least $1 - \delta - 1/n$ and uses*

- $\mathcal{O}(\log(Mn/\delta\varepsilon)\log(1/\delta)/\varepsilon^2)$ *bits of random access storage,*
- $\mathcal{O}(\log(Mn/\delta\varepsilon)\log(n/\delta\varepsilon)\log(1/\delta))/\varepsilon^2)$ *random bits,*
- $\mathcal{O}(\log(n/\delta\varepsilon)\log(1/\delta))/\varepsilon^2)$ *arithmetic operations per pair $(i, a) \in S$.*

For a more elegant approach, we estimate $L_2(S)$ by the 2-norm $\|\cdot\|_2$ instead of the median. The modified algorithm returns $\|(S^0, ..., S^{l-1}\|_2$ as estimation of $L_2(S)$. The correctness of the algorithm is given by the following two contributions. Indyk et al., for truly independent X_i^j, the algorithm is correct. And on the other hand, if the random variables are generated using Nisan (as previously done), the resulting different in the probability of correctness is negligible (can be shown in the same way as for the median-based algorithm).

Theorem 16. *There exists an algorithm such that for all $0 < \varepsilon, \delta < 1$, an implicit representation of the $k \times n$-matrix A, where $k = \mathcal{O}(\log(1/\delta)/\varepsilon^2)$ can be obtained satisfying the following properties.*

- *For each (i, j), the algorithm returns $A[i, j]$ in $\mathcal{O}(\log(n))$ arithmetic operations.*
- *The algorithm uses $\mathcal{O}(\log(Mn/\delta\varepsilon)\log(n/\delta\varepsilon)\log(1/\delta))/\varepsilon^2)$ bits of space (same as before).*
- *Each entry of A can be represented using $\mathcal{O}(\log(n/\delta\varepsilon))$ bits.*
- *For all $x \in \mathbb{R}^n$, $\mathbb{P}\left[\|Ax\|_2 - \|x\|_2 > \varepsilon\|x\|_2\right] \leq \delta$.*

[2]The "random" numbers used in actual computations are not random but pseudorandom. One starts with an integer R_0 in range from 0 to $M - 1$, where M is a large number. This R_0 is called the seed and we may think of it as truly random.

This means that the dimensional reduction is contingent on the chosen ε and δ.

For general $p \in]0, 2]$, the algorithm becomes more involved mainly because no explicit formulas are known for the densities and the distribution functions. However, one can generate p-stable random variables as in Example 9 in the Preliminaries.

Lemma 17. *Let F be a cumulative distribution function of a random variable $|Z|$, where Z is drawn from a p-stable distribution. Then, for some constants $c_1, c_2, c_3 > 0$ and for all $p, \varepsilon > 0$, there exists $t \in [c_1, c_2]$ such that*

$$\left| F^{-1}\left(t - \frac{\varepsilon}{c_3}\right) - F^{-1}\left(t + \frac{\varepsilon}{c_3}\right) \right| \leq \varepsilon$$

Given the Lemma, we can estimate $L_p(S)$ by taking the t-quantile (instead of the median) of $|S^0|, ..., |S^{l-1}|$. Note that, unlike for $p = 1, 2$, the value of t depends on ε.

Theorem 18. *For all $p \in]0, 2[$ and for all $0 < \varepsilon, \delta < 1$, there exists a non-uniform algorithm that estimates $L_p(S)$ up to a factor $1 \pm \varepsilon$ with probability $1 - \delta$ and uses*

- *$\mathcal{O}(\log(Mn/\delta\varepsilon)\log(1/\delta)/\varepsilon^2)$ bits of random access storage,*
- *$\mathcal{O}(\log(Mn/\delta\varepsilon)\log(n/\delta\varepsilon)\log(1/\delta))/\varepsilon^2)$ random bits,*
- *$\mathcal{O}(\log(n/\delta\varepsilon))$ arithmetic operations per pair $(i, a) \in S$, compared to $\mathcal{O}(\log(n/\delta\varepsilon)\log(1/\delta))/\varepsilon^2)$ operations for $p = 1, 2$.*

Dimensionality Reduction For L_1

We show how to obtain the sketch function C. We describe C in terms of dimensionality reduction of l_1^n, the adaption to the stream model can be done as in the previous section.

Theorem 19. *For all $1/2 \geq \varepsilon, \delta > 0$, with $\varepsilon > \gamma > 0$, there exists a probability space over linear mappings $f : l_1^n \to l_k^1$, where $k = \log(1/\delta)^{1/(\varepsilon-\gamma)}/c(\gamma)$, for a function $c(\gamma) > 0$ depending only on γ, such that for all $p, q \in l_1^n$,*

$$\mathbb{P}\left[\|f(p) - f(q)\|_1 \leq (1+\varepsilon)\|p - q\|_1\right] \leq \delta,$$
$$\mathbb{P}\left[\|f(p) - f(q)\|_1 \geq (1-\varepsilon)\|p - q\|_1\right] \leq \frac{1+\gamma}{1+\varepsilon}.$$