Data Interpolation
An Efficient Sampling Alternative for Big Data Aggregation

by

Hadassa Daltrophe, Shlomi Dolev and Zvi Lotker

Technical Report #13-01

September 2012

Data Interpolation
An Efficient Sampling Alternative for Big Data Aggregation

(Technical Report)

Hadassa Daltrophe* Shlomi Dolev† Zvi Lotker‡

October 11, 2012

Abstract

Given a large set of measurement sensor data, in order to identify a simple function that captures the essence of the data gathered by the sensors, we suggest representing the data by (spatial) functions, in particular by polynomials. Given a (sampled) set of values, we interpolate the datapoints to define a polynomial that would represent the data. The interpolation is challenging, since in practice the data can be noisy and even Byzantine, where the Byzantine data represents an adversarial value that is not limited to being close to the correct measured data. We present two solutions, one that extends the Welch-Berlekamp technique in the case of multidimensional data, and copes with discrete noise and Byzantine data, and the other based on Arora and Khot techniques, extending them in the case of multidimensional noisy and Byzantine data.

*Department of Computer Science, Ben-Gurion University, Beer-Sheva, 84105, Israel. Email: hd@cs.bgu.ac.il. Partially supported by a Russian Israeli grant from the Israeli Ministry of Science and Technology and the Russian Foundation for Basic Research, the Rita Altura Trust Chair in Computer Sciences, the Lynne and William Frankel Center for Computer Sciences, Israel Science Foundation (grant number 428/11), Cabarnit Cyber Security MAGNET Consortium, Grant from the Institute for Future Defense Technologies Research named for the Medvedi of the Technion, MAFAT, and Israeli Internet Association

†Department of Computer Science, Ben-Gurion University, Beer-Sheva, 84105, Israel. Email:dolev@cs.bgu.ac.il. Partially supported by Deutsche Telekom, Rita Altura Trust Chair in Computer Sciences, Lynne and William Frankel Center for Computer Sciences, Israel Science Foundation (grant number 428/11) and Cabarnit Cyber Security MAGNET Consortium.

‡Department of Communication Systems Engineering, Ben-Gurion University, Beer-Sheva, 84105, Israel. Email: zvilo@bgu.ac.il.

1 Introduction

Consider the task of representing information in an error-tolerant way, such that it can be introduced even if it contains noise or even if the data is partially corrupted and destroyed. Polynomials are a common venue for such approximation, where the goal is to find a polynomial p of degree at most d that would represent the entire data correctly.

Our motivation comes from sensor data aggregation, and the need to extend the distributed aggregation to distributed interpolation, use sampling to cope with huge data and anticipate the value of missing data. For example, a sensor network may interact with the physical environment, while each node in the network is may sense the surrounding environment (e.g., temperature, humidity etc). The environmental measured values should be transmitted to a remote repository or remote server. Note that the environmental values usually contain noise, and there can be malicious inputs, i.e., part of the data may be corrupted.

In contrast to distributed data aggregation where the resulting computation is a function such as COUNT, SUM and AVERAGE (e.g. [9, 13, 16]), in distributed data interpolation, our goal is to represent every value of the data by a single (abstracting) function. Our computational model consists of sampling the sensor network data and estimating the missing information using polynomial manipulations.

The management of big data systems also gives motivation for the distributed interpolation method. The abstraction of big data becomes one of the most important tasks in the presence of the enormous amount of data produced these days. Communicating and analyzing the entire data does not scale, even when data aggregation techniques are used. This study suggests a method to represent the distributing big data by a simple abstract function (such as polynomial) which will lead to effective use of that data.

We suggest interpolating the big data in the scope of distributed systems by using local *data centers*. Each data center samples the data around it and computes a polynomial that reflects the local data. The local polynomials are merged to a global one by interpolation in a hierarchical manner. In the process of calculating the local polynomials noise and Byzantine data samples are eliminated.

Basic Definitions.

- For *multivariate polynomial* $p(\mathtt{x}) \in \mathbb{R}[\mathtt{x}] = \mathbb{R}[x_1, ..., x_k]$ let $\|p\|_\infty = sup\{|p(x_1, ..., x_k)| : x_1, ..., x_k \in \mathbb{R}\}$.
- A *monomial* in a collection of variable $x_i, ..., x_n$ is a product

  $$x_1^{\alpha_1} x_2^{\alpha_2} x_n^{\alpha_n}$$

  where α_i are non-negative integers.
- The *total degree* of a multivariate polynomial p is the maximum degree of any term in p, where the degree of particular term is the sum of the variable exponents.
- A polynomial q is a δ-*approximation* to p if $\|p - q\|_\infty \leq \delta$.

Polynomial Fitting to Noisy and Byzantine Data.

Formally, in this paper, we learn the following problem:

Definition 1.1 (Polynomial Fitting to Noisy and Byzantine Data Problem)**.** *Given a sample S of k dimension datapoints $\{(x_{1_i}, ..., x_{k_i})\}_{i=1}^{N}$ and a function f defined on those points $f(x_{1_i}, ..., x_{k_i}) = y_i$, a noise parameter $\delta > 0$ and a threshold $\rho > 0$, we have to find a polynomial p of total degree d satisfying*

$$p(x_1, ..., x_k) \in [y - \delta, y + \delta] \text{ for at least } \rho \text{ fraction of } S \tag{1}$$

Generally, we propose the use of polynomials to represent large amounts of sensor data. The process works by sampling the data and then using this sample to construct a polynomial whose distance (according to the ℓ_∞ metric) from the polynomial constructed using the *whole* data set is small. The main challenges to this approach are (i) the presence of noise (identified by the δ parameter), and (ii) arbitrarily corrupted data (Byzantine data, denoted by ρ) that can cause inaccurate sampling and, thus, lead to badly constructed polynomials.

Given that the function f is continuous, by the Weierstrass approximation Theorem [4] we know that for any given $\epsilon > 0$, there exists a polynomial p' such that

$$\left\| f - p' \right\|_\infty < \epsilon \tag{2}$$

This can tell us that our desired polynomial p exists (i.e., $p' = p$ and $\epsilon = \delta$, satisfying eq.1), and we can relate the data as arising from polynomial function (i.e., the unknown function f is d degree polynomial we need to reconstruct), and this is the underlying model assumed in the paper.

One obvious candidate to construct approximating polynomial is interpolation at equidistant points. However, the sequence of interpolation polynomials does *not* converge uniformly to f for all $f \in C[0, 1]$ due to Runge's phenomenon [7]. Chebyshev interpolation (i.e., interpolate f using the points defined by the Chebyshev polynomial) minimizes Runge's oscillation, but it is not suffice the polynomial fitting problem presented above (Definition 1.1) due to the randomly distributed data we have assumed.

Taylor polynomials are also not appropriate; for even setting aside questions of convergence, they are applicable only to functions that are *infinitely differentiable*, and not to all continuous functions.

Another classical polynomial sequence is suggested by S. Bernstein [3] as constructive proof of the Weierstrass Theorem. Bernstein polynomial: $B_n^f(x) = \sum_{i=0}^{n} f\left(\frac{i}{n}\right)\binom{n}{i}x^i(1-x)^{n-i}$ converges uniformly to any continuous function f which is bounded on $[0, 1]$. The formal Berenstein polynomial samples the function f in an *equidistant* fashion. To handle a random sample data, we can use Vitale [21] results which consider that the datapoints $S = x_1, ..., x_N$ are $i.i.d$ observations drawn from an unknown *density* function f. The *Bernstein polynomial estimate of* f defined as $\tilde{B}_n^f(x) = \frac{n+1}{N}\sum_{i=0}^{n}\mu_{in}^N\binom{n}{i}x^i(1-x)^{n-i}$ where μ_{in}^N is the number of points (x_i's) appear in the interval $[\frac{i}{n+1}, \frac{i+1}{n+1}]$. Vitale [21] showed that $\left\|\tilde{B}_n^f(x) - f\right\|_\infty \leq \epsilon$ for every given $\epsilon > 0$.

Tenbusch [20] extended Vitale's idea to multidimensional densities, where there is need to note that those works hold only when the datapoints are $i.i.d$ observations. Another reason not to use the Bernstien polynomial is the slow convergence rate (Voronovskaya's Theorem states that for functions that are twice differentiable, the rate of convergence is about $1/n$, see Davis [7]).

Considering other classical curve-fitting and approximation theories [17], most research has used the ℓ_2 norm of noise, such as the method of least square errors. These attitudes not suffice the

adversarial noise we have assumed here. To our knowledge, only [2] referred the ℓ_∞ noise that fits our considered problem and we further relate [2] study.

The polynomial fitting problem as stated in Definition 1.1 can also be studied by Error-Correcting Code Theory. From that point of view, extensive literature exists dealing with the noise-free case (i.e., $\delta = 0$ and $\rho < 1$). In the next section, we present an algorithm that handles a combination of discrete noise and Byzantine data based on the Welch-Berlekamp [22] error-elimination method. Moreover, the fundamental Welch-Berlekamp algorithm treats only the one-dimension case, where we suggest a means to deal with corrupted-noisy data appearing at one and multi-dimensional inputs.

Related to unrestricted noise, we refer to the polynomial-fitting problem as defined by Arora and Khot [2]. Based on their results in Section 3, we introduce the polynomial fit generalization, where we provide a polynomial-time algorithm dealing with multi-variate data.

Summarizing, this work provides the following contributions:

- We describe an algorithm that constructs a polynomial using the Welch-Berlekamp (WB) method as a subroutine. The algorithm is tolerated to discrete-noise and Byzantine data.
- We identify how the previous method can be generalized to handle multi-dimensional data. Moreover, we present a multivariate analogue of the WB method, under conditions which will be specified.
- Using linear programing minimization and the Markov-Bernstein Theorem, we generalized Arora and Khot algorithm to reconstruct an unknown *multi-dimensional* polynomial. Furthermore, we detail the way to eliminate the Byzantine appearance when such inputs exist.

Those three points stated in the three algorithms presented in the paper. The first Algorithm handles one-dimensional Byzantine data that contains discrete-noise. Algorithm 2 generalized the WB idea to deal with multivariate malicious data. Finally, Algorithm 3 summarized our approach to cope with unrestricted noise appeared in the (partially corrupted) data.

2 Discrete Finite Noise

In this section, we will study a simple aspect of the polynomial fitting problem posed in Definition 1.1, where the data function is a polynomial, and we relaxed the noise constraint to be finite and discrete, i.e., the noise δ is defined on a finite field $\mathbb{F}_q$ containing q elements.

Welch and Berlekamp related the problem of polynomial reconstruction in their decoding algorithm for Reed Solomon codes [22]. The main idea of their algorithm is to describe the received (partially corrupted) data as a ratio of polynomials. Their solution holds for noise-free cases and a limited fraction of the corrupted data ($\delta = 0, \rho > 1/2$). Almost 30 years later, Sudan's list decoding algorithm [19] relaxed the Byzantine constraint ($\delta = 0, \rho$ can be less than $1/2$) by using bivariate polynomial interpolation. Those concepts do not hold up well in the noisy case since they use the roots of the polynomial and the divisibility of one polynomial by other methods that are problematic for noisy data (as shown in [2], Section 1.2). Here, we will use the WB algorithm [22] as a "black box" to obtain an algorithm that handles the discrete-noise notation of the polynomial-fitting problem.

Given a data set $\{(x_i, y_i)\}_{i=1}^N$ that is within a distance of $t = \rho N$ from some polynomial $p(x)$ of degree $< d$, the WB approach to eliminate the irrelevant data is to use the roots of an object called

the error-locating polynomial, e. In other words, we want $e(x_i) = 0$ whenever $p(x_i) \neq y_i$. This is done by defining another polynomial $q(x) \equiv p(x)e(x)$. To resolve these polynomials we need to solve the linear system, $q(x_i) = y_i e(x_i)$ for all i.

Welch and Berlekamp show that $e(x)|q(x)$ and $p(x)$ can be found by the ratio $p(x) = q(x)/e(x)$ at $O(N^\omega)$ running time (where ω is the matrix multiplication complexity). In Algorithm 1, we are use the WB method as a subroutine to manage the noisy-corrupted data.

Algorithm 1 Reconstruct the polynomial $p(x)$ representing the true data

Require: $S, S' \subseteq S, \rho, d, \delta, \Delta = v_1, ..., v_{|S'|}$
  $i \leftarrow 0$
  repeat
    $i \leftarrow i + 1$
    $S'_i \leftarrow S' + v_i$
    $p_i(x) \leftarrow WB(S'_i, d, \rho)$
  until $p_i(x_j) \in [y_j - \delta, y_j + \delta]$ for at least ρ fraction of j's; $(x_j, y_j) \in S - S'_i$
  return $p(x) \leftarrow p_i(x)$

Given *any* sample S such that ρ fraction of S is not corrupted, we will choose a subset $S' \subseteq S$ in a size related to the desired degree d and ρ (the WB algorithm requires $2t + d$ points, where $t = \rho N$ is the number of the corrupted points). At every step i, we will add S' different values of noise as defined by the set Δ which contain all the vectors of length $|S'|$ assigned the elements of $\mathbb{F}_q$ in lexicographic order, i.e., $\Delta = \left\{(a_1, ..., a_{|S'|}) : a_i \in \mathbb{F}_q\right\}$. Now, we can reconstruct the polynomial p_i using the WB algorithm. The resulting polynomial p_i is tested by the original dataset S, where the criteria is that p_i is within δ from all nodes but the Byzantine nodes (according to the maximal number of Byzantine as defined by ρ).

Since we assume a discrete finite noise ($\delta \in \mathbb{F}_q$), for each datapoint at the subset S' (of size $O(d + \rho|S|)$), there is a possibly of q values (where q is a constant). Thus, in the worst case, when we run the WB polynomial algorithm for every possible value, it will cost $poly(d + N)$ time.

Note that if the desired polynomial's degree d is not given, we can search for the minimal degree of a polynomial that fits the δ and number of Byzantine node restrictions in a binary search fashion.

Multidimensional Data.

To generalize the former algorithm to handle multidimensional data, there is need to formalize the WB algorithm to deal with multivariate polynomials. This is a challenging task due to the *infinite* roots those polynomials may have (and as previously mentioned, the WB method is strictly based on the polynomials' roots).

A suggested method to handle 3-dimensional data is to assume that the values of datapoints in one direction (e.g., x-direction) are distinct. This can be achieved by assuming the inputs $S = (x_1, y_1, f(x_1, y_1))..., (x_N, y_N, f(x_N, y_N))$ are *i.i.d* observations. Moreover, we allow the malicious authority to change the observation input but not its distribution (i.e., to determine $z_i = f(x_i, y_i)$ value only). This assumption forces the data to have different x_i's values, which help us to define the error locating polynomial e in the x-direction only (or symmetrically over the y-axis).

The 3-dimensional polynomial reconstruction is described in Algorithm 2.

Algorithm 2 Reconstruct the polynomial $p(x, y)$ representing the true data

- **Input**: $0 < t = \rho N$ which is the Byzantine appearance bound, the total degree $d > 1$ of the goal polynomial and N triples $(x_i, y_i, z_i)_{i=1}^{N}$ with distinct x_i's .
- **Output**: Polynomial $p(x, y)$ of total degree at most d or fail.
- **Step 1**: Compute a non-zero univariate polynomial $e(x)$ of degree exactly t and a bivariate polynomial $q(x, y)$ of total degree $d + t$ such that:

$$z_i e(x_i) = q(x_i, y_i) \quad 1 \leq i \leq N \tag{3}$$

  If such polynomials do not exist, output fail.

- **Step 2**: If e does not divide q, output fail, else compute $p(x, y) = \dfrac{q(x, y)}{e(x)}$. If $\Delta(z_i, p(x_i, y_i)_i) > t$, output fail. else output $p(x, y)$.

Theorem 2.1. *Let p be an unknown d total degree polynomial with two variables. Given a threshold $\rho > 0$ and a sample S of $N = \binom{d+t+m}{d+m} + t$ ($t = \rho N$) random points $(x_i, y_i, z_i)_{i=1}^{N}$ such that*

$$z_i = p(x_i, y_i) \text{ for at least } \rho \text{ fraction of } S.$$

The algorithm above reconstructs p at $O(N^{\omega})$ running time (where ω is the matrix multiplication complexity).

Proof. The proof of the Theorem above follows from the subsequent claims.

Claim 2.2 (Correctness)**.** *There exist a pair of polynomials $e(x)$ and $q(x, y)$ that satisfy **Step 1** such that $q(x, y) = p(x, y)e(x)$.*

Proof. Taking the error locator polynomial $e(x)$ and $q(x, y) = p(x, y)e(x)$, where $deg(q) \leq deg(p) + deg(e) \leq t + d$. By definition, $e(x)$ is a degree t polynomial with the following property:

$$e(x) = 0 \text{ iff } z_i \neq p(x, y)$$

We now argue that $e(x)$ and $q(x, y)$ satisfy eq. 3. Note that if $e(x_i) = 0$, then $q(x_i, y_i) = z_i e(x_i) = 0$. When $e(x_i) \neq 0$, we know $p(x_i, y_i) = z_i$ and so we still have $p(x_i, y_i)e(x_i) = z_i e(x_i)$, as desired. □

Claim 2.3 (Uniqueness)**.** *If any two distinct solutions $(q_1(x, y), e_1(x)) \neq (q_2(x, y), e_2(x))$ satisfy **Step 1**, then they will satisfy $\dfrac{q_1(x, y)}{e_1(x)} = \dfrac{q_2(x, y)}{e_2(x)}$.*

Proof. It suffices us to prove that $q_1(x, y)e_2(x) = (q_2(x, y)e_1(x)$. Multiply this with z_i and substitute x, y with x_i, y_i, respectively,

$$q_1(x_i, y_i)e_2(x_i)z_i = q_2(x_i, y_i)e_1(x_i)z_i$$

We know, $\forall i \in [N]$ $q_1(x_i, y_i) = e_1(x_i)z_i$ and $q_2(x_i, y_i) = e_2(x_i)z_i$ If $z_i = 0$, then we are done. Otherwise, if $z_i \neq 0$, then $q_1(x_i, y_i) = 0, q_(x_i, y_i) = 0 \Rightarrow q_1(x, y)e_2(x) = (q_2(x, y)e_1(x)$ as desired. □

Claim 2.4 (Time complexity)**.** *Given $N = t + \binom{d+t+2}{d+t}$ data samples, we can reconstruct $p(x, y)$ using $O(N^\omega)$ running time.*

Proof. Generally, for m variate polynomial with degree d, there are $\binom{d+m}{d}$ terms [18]; thus, it is a necessary condition that we have $t + \binom{d+t+2}{d+t}$ distinct points for q and e to be uniquely defined. We have N linear equation in at most N variables, which we can solve e.g., by Gaussian elimination in time $O(N^\omega)$ (where ω is the matrix multiplication complexity).

Finally, **Step 2** can be implemented in time $O(NlogN)$ by long division [1]. Note that the general problem of deciding whether one multivariate polynomial divides another is related to computational algebraic geometry (specifically, this can be done using the Gröbner base). However, since the divider is a univariate polynomial, we can mimic long division, where we consider x to be the "variable" and y to just be some "number."

□

Example 1. Suppose the unknown polynomial is $p(x, y) = x + y$. Given the parameters: $d = 1$ (degree of p), $m = 2$ (number of variable at p) and $t = 1$ (number of corrupted inputs) and the set of $t + \binom{d+t+2}{d+t} = 7$ points:

$$(1,2,2),(2,2,4),(6,1,7),(4,3,7),(8,2,0),(9,1,10),(3,7,10)$$

that lie on $z = p(x, y)$. Following the algorithm, we define: $deg(e) = 1, deg(q) = 2$ and

$$q_i = \alpha_1 x_i^2 + \alpha_2 x_i y_i + \alpha_3 y_i^2 + \alpha_4 x_i + \alpha_5 y_i + \alpha_6 = z_i(x_i + \alpha_7)$$

for coefficients $\alpha_1, ..., \alpha_6, \beta$ and $1 \le i \le 12$. Note that we force $e(x)$ not to be the zero polynomial by define it to be monic (i.e., the leading coefficient equals to 1). Thus, we derive the linear system:

$$\begin{aligned}
\alpha_1 + \alpha_2 + \alpha_3 + \alpha_4 + \alpha_5 + \alpha_6 &= 2\beta + 2 \\
4\alpha_1 + 4\alpha_2 + 4\alpha_3 + 2\alpha_4 + 2\alpha_5 + \alpha_6 &= 4\beta + 8 \\
36\alpha_1 + 6\alpha_2 + \alpha_3 + 6\alpha_4 + \alpha_5 + \alpha_6 &= 7\beta + 42 \\
16\alpha_1 + 12\alpha_2 + 9\alpha_3 + 4\alpha_4 + 3\alpha_5 + \alpha_6 &= 7\beta + 28 \\
64\alpha_1 + 16\alpha_2 + 4\alpha_3 + 8\alpha_4 + 2\alpha_5 + \alpha_6 &= 0 \\
81\alpha_1 + 9\alpha_2 + \alpha_3 + 9\alpha_4 + \alpha_5 + \alpha_6 &= 10\beta + 90 \\
9\alpha_1 + 21\alpha_2 + 49\alpha_3 + 3\alpha_4 + 7\alpha_5 + \alpha_6 &= 10\beta + 30
\end{aligned}$$

Solving the system gives: $q(x, y) = x^2 + xy - 8x - 8y$ and $e(x) = x - 8$. Dividing those polynomials, we get the expected solution: $q(x, y)/e(x) = p(x) = x + y$.

Corollary 2.5 (Multivariate Polynomial Reconstruction)**.** *Let p be an unknown d total degree polynomial with m variable. Given a threshold $\rho > 0$, a noise parameter δ and a sample S of N random points $(x_{1_i}, ..., x_{m_i}, y_i)_{i=1}^N$ such that*

$$y_i \in [p(x_{1_i}, ..., x_{m_i}) - \delta, p(x_{1_i}, ..., x_{m_i}) + \delta] \text{ for at least } \rho \text{ fraction of } S$$

p can be reconstructed using $N = \binom{d+m+\rho m}{d+m} + \rho m$ datapoints.

Proof. Following Theorem 2.1, we can rewrite its proof for the multidimensional generalization. An interesting question is if there is any advantage to define the error-location polynomial e to be multivariate (instead of univariate as we previously presented). One "advantage" may be decreasing the required number of datapoints or improvement of the complexity. The size of the input data is strictly defined by the given bound on the corrupted data ($t = \rho N$) and the goal polynomial degree (d). Thus, there is no sense if the unknown coefficient comes from the highly *degree* polynomial or from the highly *dimension* polynomial, i.e., both options require the same size of sample, as illustrated in the Appendix.

Related to the complexity change, when the error-locating polynomial is multivariate, **Step 2** of Algorithm 2 is more challenging since it contains multivariate polynomial division. A related reference is [10] which is the most efficient implementation for the computation of Gröbner bases relies on linear algebra. Using Gröbner bases we can implement the division at close to $O(NlogN)$ time, as done in Algorithm 2.

To finish the proof, there is the need to explain how to deal with the noise. Since we assume only discrete noise, we can dismiss it using the method illustrated at Algorithm 1. We consistently insert a vector of possible noise and try to reconstruct the polynomial using Algorithm 2. □

3 Random Sample with Unrestricted Noise

Motivated by applications in vision, Arora and Khot [2] studied the univariate polynomial fitting to noisy data using $O(d^2)$ datapoints, where d is the polynomial degree. In this part, we generalized their results to k-dimensional data.

Since our motivation comes from sensor planar aggregation, we will focus on bivariate polynomial reconstruction, where the multivariate proof is symmetric. We assume by rescaling the data that each $x_i, y_i, f(x_i, y_i) \in [-1, 1]$. Allowing small noise at every point and large noise occasionally then there may be too many polynomials agreeing with the given data. Thus, given the noise parameter δ, our goal is to find a polynomial p that is a δ-approximation of f, i.e., p is δ-close in ℓ_∞ norm to the unknown polynomial.

Let I be a set of d^5 equally spaced points that cover the interval $[-1, 1]$. Given the random sample $S \subset I, |S| = \frac{d^2}{\delta} log(\frac{d}{\delta})$, we approach the reconstruction problem by defining a linear programming system with the fitting polynomial as its solution. To incorporate the constraint that the unknown polynomial must take values of $[-1, 1]$, we move to Chebyshev's representation of the polynomial. Thus, each of its coefficients is at most $\sqrt{2}$ (see eq. 5). We represent Chebyshev's polynomial by $T_i(\cdot), T_j(\cdot)$, and the variables c_{ij} at the system are the Chebyshev coefficients. We construct the LP:

$$\text{minimize } \delta$$

$$s.t.$$

$$f(x_k, y_k) - \delta \leq \sum_{i}^{n} \sum_{j}^{m} c_{ij} T_i(x_k) T_j(y_k) \leq f(x_k, y_k) + \delta, \quad k = 1, ..., |S| \tag{4}$$

$$|c_{ij}| \leq \sqrt{2}, \quad i = 1, ..., n; j = 1, ..., m \tag{5}$$

$$|\sum_{i}^{n}\sum_{j}^{m} c_{ij}T_i(x)T_j(y)| \leq 1, \quad \forall_{x,y} \in I \tag{6}$$

The following Theorem presents our main result for solving the polynomial fitting problem:

Theorem 3.1. *Let f be an unknown d total degree polynomial with two variables, such that $f(x, y) \in [-1, 1]$ when $x, y \in [-1, 1]$. Given a noise parameter $\delta > 0$, a threshold $\rho > 0$, a constant $c > 0$ (dependent on the dimension of the data) and a sample S of $O(\frac{d^2}{\delta} log(\frac{d}{\delta}))$ random points $x_i, y_i, z_i \in [-1, 1]$ such that $z_i \in [f(x_i, y_i) - \delta, f(x_i, y_i) + \delta]$ for at least ρ fraction of S. With probability at least $\frac{1}{2}$ (over the choice of S), any feasible solution p to the above LP is $c\delta$-approximation of f.*

Proof. For our proof, we need Bernstein-Markov inequality which we state below.

Theorem 3.2. *(Bernstein-Markov [8]) For a polynomial P_d of total degree d, a direction ξ and a bounded convex set $A \subset R^k$*

$$\left\| \frac{\partial}{\partial \xi} P_d \right\|_{\infty} \leq c_A d^2 \left\| P_d \right\|_{\infty} \tag{7}$$

where c_A is independent of d (and dependent on the geometric structure of A).

Let $p = p(x, y)$ be the $d = n + m$ total degree polynomial obtained from taking any solution to the above LP. We know p exists, i.e., the LP is feasible because the coefficient of f satisfies it.

Note that although the LP constraint that

$$f - \delta \leq p \leq f + \delta$$

it is NOT immediate that

$$\left\| f - p \right\|_{\infty} \leq \delta$$

(i.e., p is the δ-approximation of f) since eq. 4 stands for the discrete sample of S, where here we need to prove the δ-approximation in the continuous state.

Claim 3.3. $\left\| p \right\|_{\infty} \leq 1 + O(\frac{n^3 m + m^3 n}{|I|})$.

Since $\left\| T_i(x) \right\|_{\infty} = \left\| T_j(y) \right\|_{\infty} = 1$, from Bernstein-Markov (Theorem 3.2), we get $|T_i'(x)| = O(i^2)$. Thus:

$$\begin{aligned} |p_x'| &\leq \sum_{i}^{n}\sum_{j}^{m} |a_{ij}| \Big(T_i(x)T_j(y) \Big)'_x \\ &\leq \sum_{i}^{n}\sum_{j}^{m} \sqrt{2} O(i^2) \leq O(n^3 m) \end{aligned} \tag{8}$$

From symmetric consideration, we get:

$$\left|p'_y\right| \leq O(nm^3) \tag{9}$$

By construction, p takes all values in $[-1,1]$ for all points in I, and the distance (in x direction or y direction) between successive points of I is $2/|I|$ (I is equidistant). The claim follows from the fact that the derivative p' by definition gives the rate of change in p: $\|p\|_\infty$ is between two successive points of I that their values $= 1$ where the possible change at the interval of length $2/|I|$ is $p' = O(n^3m + m^3n)$.

Claim 3.4. $\|p'_x\|_\infty, \left\|p'_y\right\|_\infty \leq O((n+m)^2)$.

This follows from Bernstein-Markov (Theorem 3.2) and the estimate $\|p\|_\infty = 1 + O(1)$.

Let ϵ denote the largest distance between two successive points out of $(x_1, y_1), ..., (x_{|S|}, y_{|S|})$. Every interval of size ϵ contains at least one of the datapoints (forming ϵ-net). With high probability, $\epsilon = O(log|S|/|S|) = O(\frac{\delta}{(n+m)^2})$. Now, p and f are functions satisfying $\|p'_x\|_\infty, \|f'_x\|_\infty, \left\|p'_y\right\|_\infty, \left\|f'_y\right\|_\infty \leq O((n+m)^2)$; hence, $\|(p-f)'_x\|_\infty, \left\|(p-f)'_y\right\|_\infty \leq O((n+m)^2)$. Due to the LP constraint, p, f differs by at most δ on the points in the ϵ-net, so we get

$$\|(p-f)\|_\infty \leq 2\delta + O(\epsilon(n+m)^2) = c\delta \tag{10}$$

which is the finished proof of Theorem 3.1; □

Remarks:

- If we know that the derivative is bounded by Δ (i.e., $f'_x, f'_y \leq \Delta$), the above proof that gives us $\frac{\Delta}{\delta} log \frac{\Delta}{\delta}$ points is sufficient.
- The Bernstein-Markov Theorem 3.2 also holds for multivariate trigonometric polynomials (see [8]), thus, we can generalize the above proof also for this class of function. This generalization is important in the scope of wireless sensor networks since the use of trigonometric function is the appropriate way to represent the sensor data behavior (e.g., temperature).
- The presented method holds only when we assume equidistance or random sampling (as opposed to Section 2 that handles any given sample). Otherwise, when the dataset is dense, since we allow δ perturbation of the data, it can cause a sharp slope in the resulting function although the original data is close to the constant at the sampling interval.

Corollary 3.5. *Given the set S of $O(\frac{d^2}{\delta} log(\frac{d}{\delta}))$ k-dimensional random datapoints and a constant $c(S,k)$ dependent only on the geometry and the dimension of the data, we can reconstruct the unknown polynomial within $c(S,k)\delta$ error in ℓ_∞ norm with high probability over the choice of the sample.*

Proof. The two-dimensional proof holds for the general dimension, where the approximation accuracy dependent on the constant $c(S,k)$ comes from Theorem 3.2. This constant is independent of the polynomial degree, but dependent on the set of the data points (see [8]). Note that $c(S,k)$ increases exponentially when increasing the dimension. □

Byzantine Elimination.

Arora and Khot [2] do not deal with Byzantine inputs; however, the method they presented in Section 6 can be rewritten to eliminate corrupted data such that the input datapoints will contain only true (but noisy) values.

Assume that ρ fraction of the data is uncorrupted. For any point $x_i, y_i \in [-1, 1]$, consider a small square-interval $\Lambda = [x_i - \frac{\delta}{d^3}, x_i + \frac{\delta}{d^3}] \times [y_i - \frac{\delta}{d^3}, y_i + \frac{\delta}{d^3}]$ (where d is the total degree of the polynomial we need to find). For a sample of $d^4 \frac{log(1/\delta)}{\delta}$ points, with high probability $\Omega(log(d))$ of the samples lie in this square. We are given that ρ fraction of these sample points gives an approximate value of $f(x_i, y_i)$, i.e., the correct value lies in the interval $[f(x_i, y_i) - \delta, f(x_i, y_i) + \delta]$ and the rest of the sample is corrupted and, thus, is NOT in $[f(x_i, y_i) - \delta, f(x_i, y_i) + \delta]$. As shown in Claim 3.4, the derivatives are bounded by $O(d^2)$; thus, the value of the polynomial is essentially *constant* over Λ. Hence, at least ρ fraction of the values seen in this square will lie in $[f(x_i, y_i) - \delta, f(x_i, y_i) + \delta]$ and the rest is irrelevant corrupted data. Thus, at every point (x_i, y_i), we can reconstruct $f(x_i, y_i)$. The sample is large enough so that we can reconstruct the values of the polynomial at say, d^2/δ equally spaced points. Now, applying the techniques presented in Section 3 enables us to recover the polynomial.

Reconstructing the Multivariate Polynomial.

To conclude this section, we summarize the presented results in Algorithm 3: The algorithm requires

Algorithm 3 Reconstruct the polynomial $p(x, y)$ representing the true data

Require: S, ρ, d, δ
  $S' \leftarrow \emptyset$
  $i \leftarrow 1$
  repeat
    $\Lambda = [x_i - \frac{\delta}{d^3}, x_i + \frac{\delta}{d^3}] \times [y_i - \frac{\delta}{d^3}, y_i + \frac{\delta}{d^3}]$
    $c \leftarrow \frac{z_1 + \ldots + z_k}{k}, z_j : (x_j, y_i) \in \Lambda$
    $S' \leftarrow \{(x_j, y_j, z_j) | (x_j, y_j) \in \Lambda \wedge z_j \approx c\}$
    $i \leftarrow i + 1$
  until $|S'| > \frac{d^2}{\delta}$
  $p(x, y) \leftarrow$LP minimization (Equations 4-6) on the set S'
  return $p(x, y)$

the dataset S, the true-data fraction ρ, the total degree of the expected polynomial d and the noise parameter δ. In the first phase, we eliminate the Byzantine occurrence, as described in the former subsection. Assuming the given data lie in $[-1, 1] \times [-1, 1]$ (or translate to that interval), for the points in S, we are looking at the $\frac{\delta}{d^3}$-close interval and choose all the points that have *constant* value at this interval (this is done by the average operation). We repeat this process until we collect enough true-datapoints, i.e., at least $\frac{d^2}{\delta}$ points. This set (sign as S' in the algorithm) is the input for the linear-programming equations which finally give us the expected polynomial as proof at Theorem 3.1.

4 Conclusions

We have presented the concept of data interpolation in the scope of sensor data aggregation and representation, as well as the new big data challenge, where abstraction of the data is essential in order to understand the semantics and usefulness of the data. Interestingly, we found that classical techniques used in numeric analysis and function approximation, such as the Welsh-Berlekamp efficient removal of corrupted data, Arora Khot and the like, relate to the data interpolation problem. Since the sensor aggregation task is usually a collection of inputs from *spatial* sensors, for the first time we have extended existing classical techniques for the case of three or even more function dimensions, finding polynomials that approximate the data in the presence of noise and limited portion of completely corrupted data.

We believe that the mathematical techniques we have presented have applications beyond the scope of sensor data collection or big data, in addition to being an interesting problem that lies between the fields of error-correcting and the classical theory of approximation and curve fitting.

Throughout the research we have distinguished two different measures for the polynomial fitting to the Byzantine noisy data problem: the first being the Welsh-Berlekamp generalization for discrete-noise multidimensional data and the second being the linear-programming evaluation for multivariate polynomials.

Approached by the error-correcting code methods, we have suggested a way to represent a noisy-malicious input with a multivariate polynomial. This method assumes that the noise is discrete. When the noise is unrestricted, based on Bernstein-Markov Theorem and Arora & Khot algorithm, we have suggested a method to reconstruct algebraic or trigonometric polynomial that traverses ρ fraction of the the noisy multidimensional data.

We suggest to use polynomial to represent the abstract data since polynomial is dense in the function space on bounded domains (i.e., they can approximate other functions arbitrarily well) and have a simple and compact representation as oppose to spline e.g., [11] or others image processing methods.

Directions for further investigation might include the use of interval computation for representing the noisy data with interval polynomials.

References

[1] A. V. Aho, J. E. Hopcroft and J. D. Ullman, "The Design and Analysis of Computer Algorithms", *Addison-Wesley Publishing Company*, 8, 1974.

[2] S. Arora, and S. Khot,"Fitting algebraic curves to noisy data", *STOC*, pp. 162-169, 2002.

[3] S. Bernstein, "Demonstration du theoreme de Weierstrass, fondee sur le calcul des probabilities", *Communications of the Kharkov Mathematical Society*, 2(13) pp. 1-2, 1912-1913.

[4] E. Bishop "A generalization of the StoneWeierstrass theorem", *Pacific Journal of Mathematics* 11 (3) pp. 777783, 1961.

[5] C. W. Clenshow, and J. G. Hayes,"Curve and Surface Fitting", *J. Inst. Maths Applics*, 1, pp. 164-183, 1965.

[6] C. de Boor, A. Ron, "Computational aspects of polynomial interpolation in several variables", *Math. Comp.*, 58 pp. 705-727, 1992.

[7] P. J. Davis, "Interpolation and approximation", Dover, 1975.

[8] Z. Ditzian ,"Multivariate Bernstein and Markov inequalities", *Journal of Approximation Theory*, 70(3) pp. 273-283, 1992.

[9] E. Fasolo, M. Rossi, J. Widmer, M. Zorzi, "In-network aggregation techniques for wireless sensor networks: a survey", *IEEE Wireless Commun.*, 14 (2) pp. 7087, 2007.

[10] J. C. Faugre, "A new efficient algorithm for computing Gröbner base", *Journal of Pure and Applied Algebra*, 139(13) pp. 6188, 1999.

[11] N. Guenther, "Approximation by spline functions", Springer, Berlin, 1989.

[12] M. Hilbert and P. Lepez, "The World's Technological Capacity to Store, Communicate, and Compute Information", *Science*, 332(6025) pp. 60-65 2011.

[13] P. Jesus, C. Baquero and P. S. Almeida, "A Survey of Distributed Data Aggregation Algorithms", *CoRR*, abs/1110.0725, 2011.

[14] E. H. Kingsley, "Bernstein polynomials for functions of two variables of class $C^{(k)}$", *Proceedings of the American Mathematical Society*, pp. 64-71, 1951.

[15] C. Lynch, "How do your data grow?", *Nature*, 455 pp. 28-29, 2008.

[16] R. Rajagopalan and P.K. Varshney, "Data aggregation techniques in sensor networks: a survey", *IEEE Commun. Surveys Tutorials*, 8 (4), 2006.

[17] T. J. Rivlin, "An introduction to the approximation of function", *Blaisdell publishing company* , 1969.

[18] K. Saniee, "A Simple Expression for Multivariate Lagrange Interpolation", *SIAM Undergraduate Research Online*, 1(1) ,2008.

[19] M. Sudan, "Decoding of Reed Solomon codes beyond the error-correction bound", *Journal of Complexity*, 13(1), pp.180-193 1997.

[20] A. Tenbusch, "Two-dimensional Bernstein polynomial density estimators", *Metrika*, 41(1), pp.233-253 1994.

[21] R. A. Vitale, "A Bernstein polynomial approach to density estimation", *Statistical Inference and Related Topics*, 2, pp.87-100 1975.

[22] L. R. Welch, and E. R. Berlekamp, "Error correction for algebraic block codes", *US Patent 4 633 470*, 1986.

A Appendix

Given that the goal unknown polynomial p has $m = 2$ variable, $deg(p) = 1$ and that the data contain $t = 2$ Byzantine appearance, we can define the error-correcting polynomial e to be univariate polynomial, $deg(e) = 2$ and get the linear equation:

$$\alpha_1 x^3 + \alpha_2 x^2 y + \alpha_3 x y^2 + \alpha_4 y^3 + \alpha_5 x^2 + \alpha_6 xy + \alpha_7 y^2 + \alpha_8 x + \alpha_9 y + \alpha_{10} = z(x^2 + \beta_1 x + \beta_2) \quad (11)$$

when substitute the given data

(1,2,2),(-2,6,0;),(2,2,4),(6,1,7),(4,3,7),(9,1,10),(3,7,10),(5,7,12),(7,4,11),(10,3,13),(11,2,13),(12,4,16)

at (eq.11) we get:

$$q_1(x, y) = xy - 2y - 2x + x^2 y + x^2 + x^3$$
$$e_1(x) = x^2 + x - 2$$

Or, by defining e to be bivariate polynomial,$deg(e) = 1$:

$$\alpha_1 x^3 + \alpha_2 x^2 y + \alpha_3 x y^2 + \alpha_4 y^3 + \alpha_5 x^2 + \alpha_6 xy + \alpha_7 y^2 + \alpha_8 x + \alpha_9 y + \alpha_{10} = z(x + \beta_1 y + \beta_2) \quad (12)$$

which its solution is:

$$q_2(x, y) = x^2 + 7xy/4 - 5x/2 + 3y^2/4 - 5y/2$$
$$e_2(x, y) = x + 3y/4 - 5/2$$

At both cases the polynomial devision result is equals and gives the expected solution:

$$q_1(x, y)/e_1(x, y) = q_2(x, y)/e_2(x, y) = x + y = p(x). \quad (13)$$