Big Data Interpolation: An Effcient Sampling Alternative for Sensor Data Aggregation

Hadassa Daltrophe, Shlomi Dolev, Zvi Lotker
Ben-Gurion University

Outline

- Introduction
  - Motivation
  - Problem definition
- General data with Discrete Finite Noise
  - Welsh & Berlekamp Algorithm
  - Multidimensional Data
- Random Sample with Unrestricted Noise
  - Byzantine elimination
- Conclusion

Outline

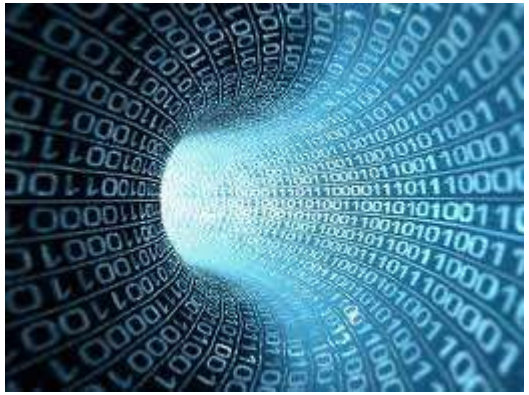
- Introduction
  - Motivation
  - Problem definition
- General data with Discrete Finite Noise
  - Welsh & Berlekamp Algorithm
  - Multidimensional Data
- Random Sample with Unrestricted Noise
  - Byzantine elimination
- Conclusion

Motivation

- Given a large set of measurment sensor data we would like to capture the essence of the data gathered by the sensor.

Motivation

- Given a **large set** of measurment sensor data we would like to capture the essence of the data gathered by the sensors.



Big Data Age

- The abstraction of big data becomes one of the most important tasks in the presence of the enormous amount of data produced these days.

web logs, sensor networks, social networks, social data,

astronomy, atmospheric science, genomics, biogeochemical, biological

military surveillance,

medical records, photography archives,

Internet text and documents, Internet search indexing, call detail records,

video archives…

Motivations

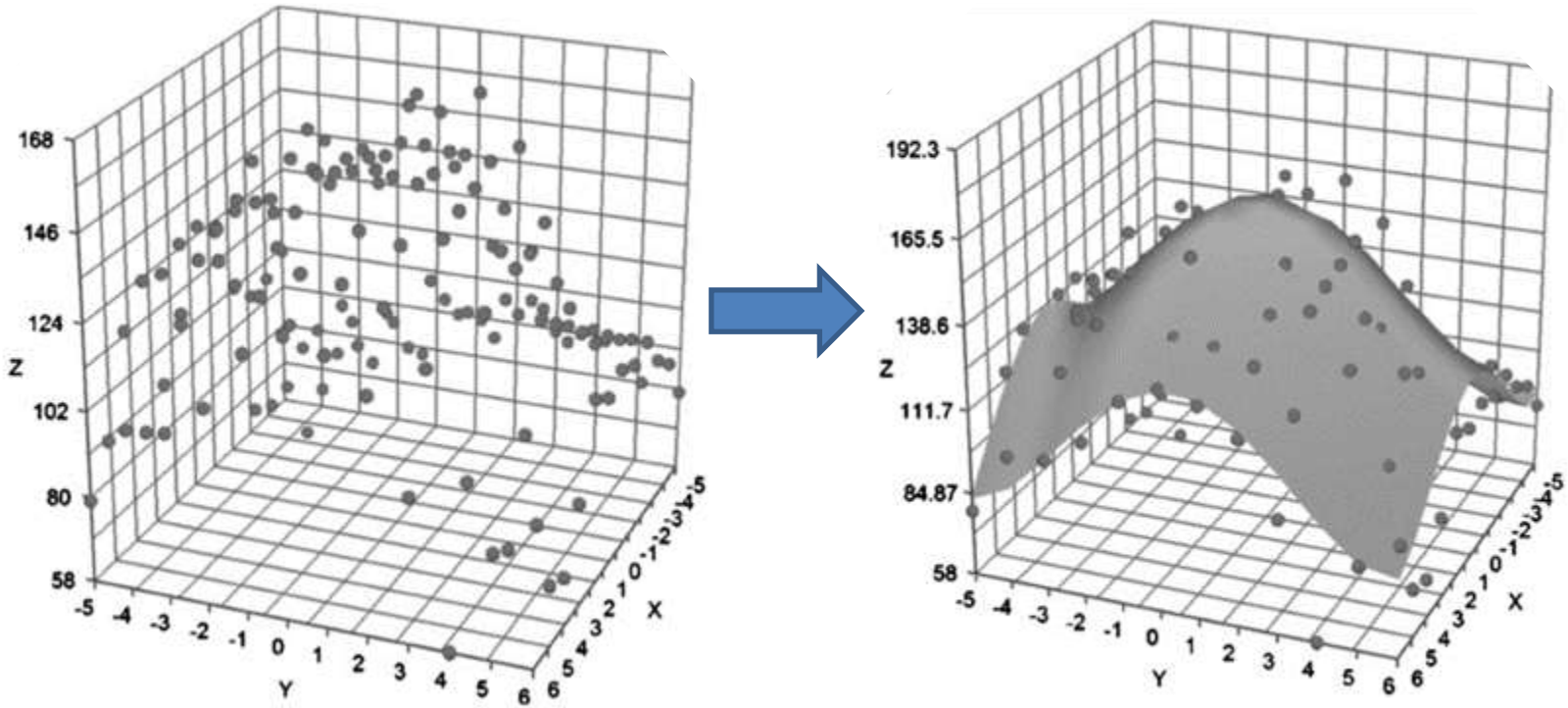
- Given a large set of measurment sensor data we would like to **capture the essence of the data** gathered by the sensors.

Data Aggregation

- Compute a function- COUNT, SUM , AVERAGE,...
- Condition queries ("Where temp > 35")
- Focus on specific domin

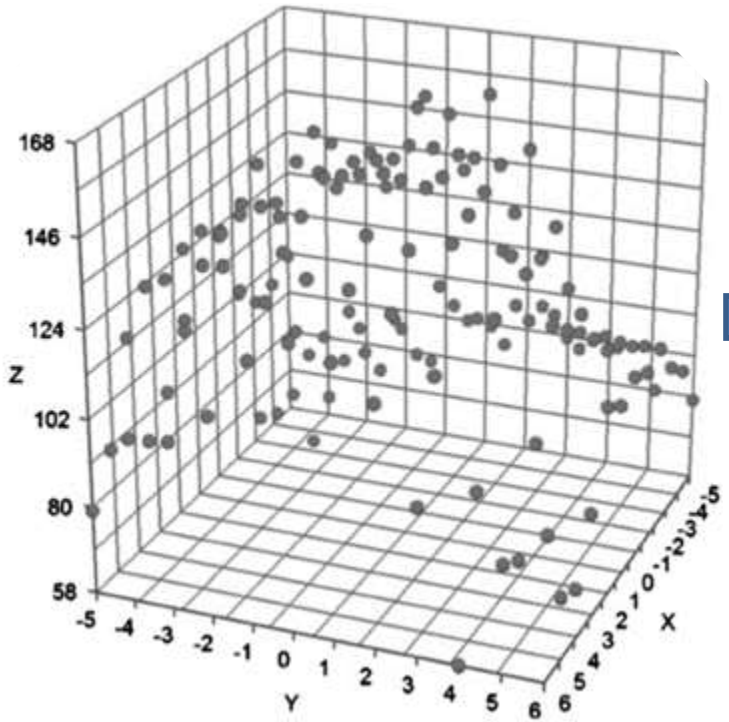
Distributed Big Data Interpolation

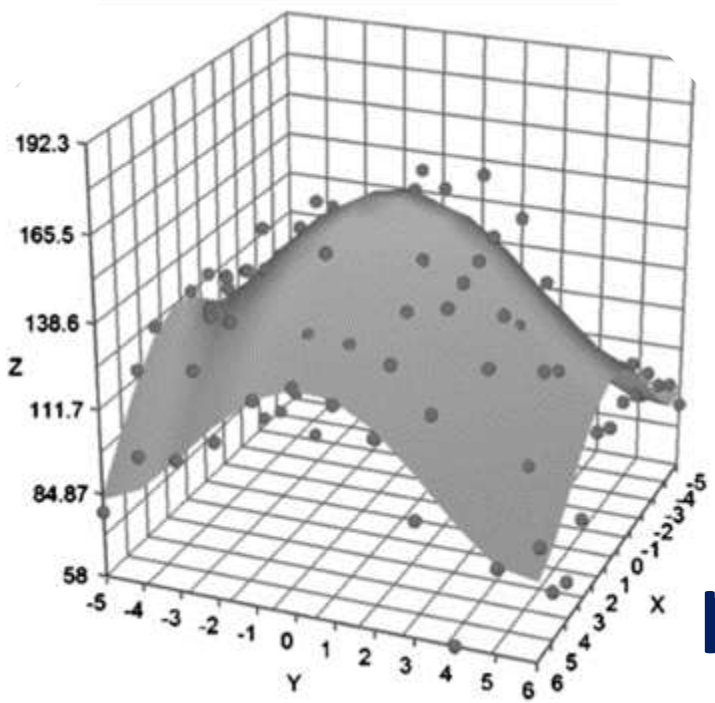
- Our goal is to represent every value of the data by a single (abstracting) function.



Distributed Big Data Interpolation

- Given a (sampled) set of values, we interpolate the datapoints to define a polynomial that would represent the data.

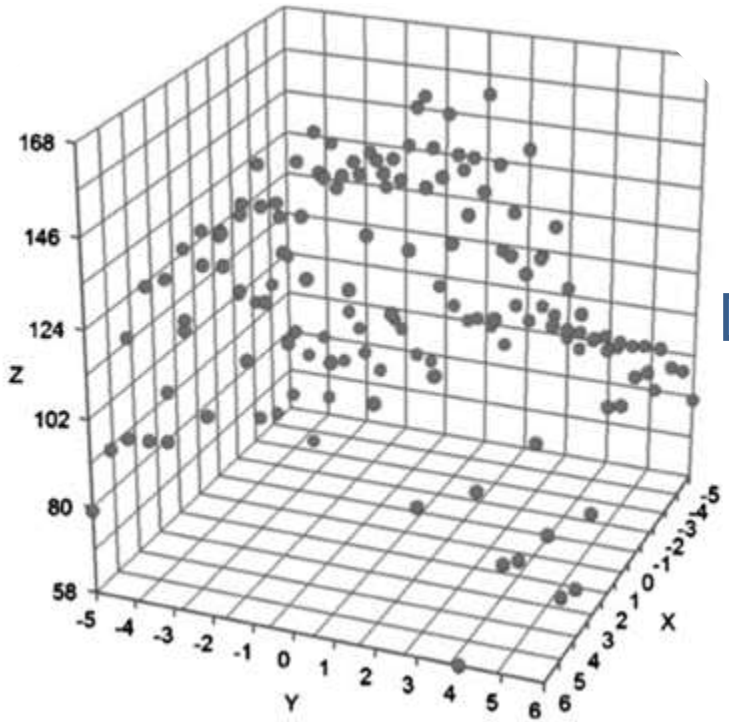


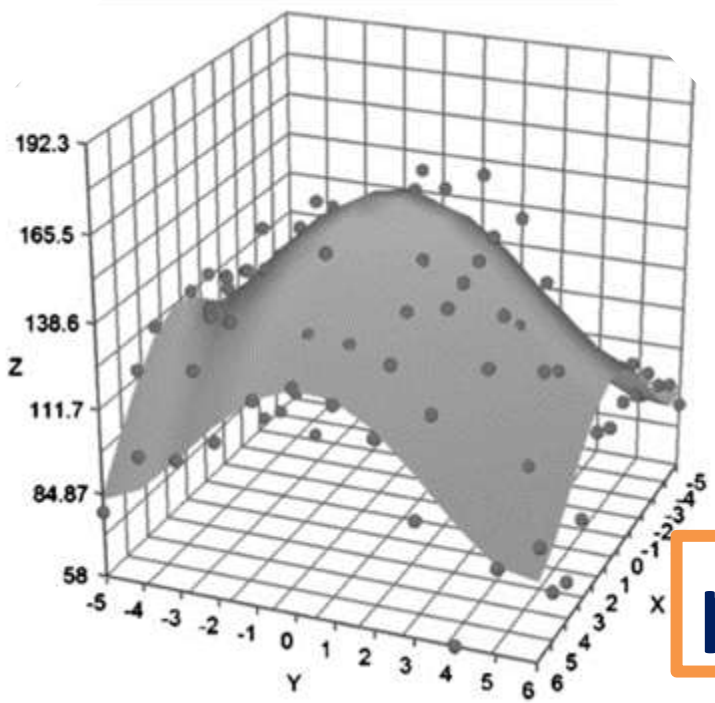


p(x,y)

Distributed Big Data Interpolation

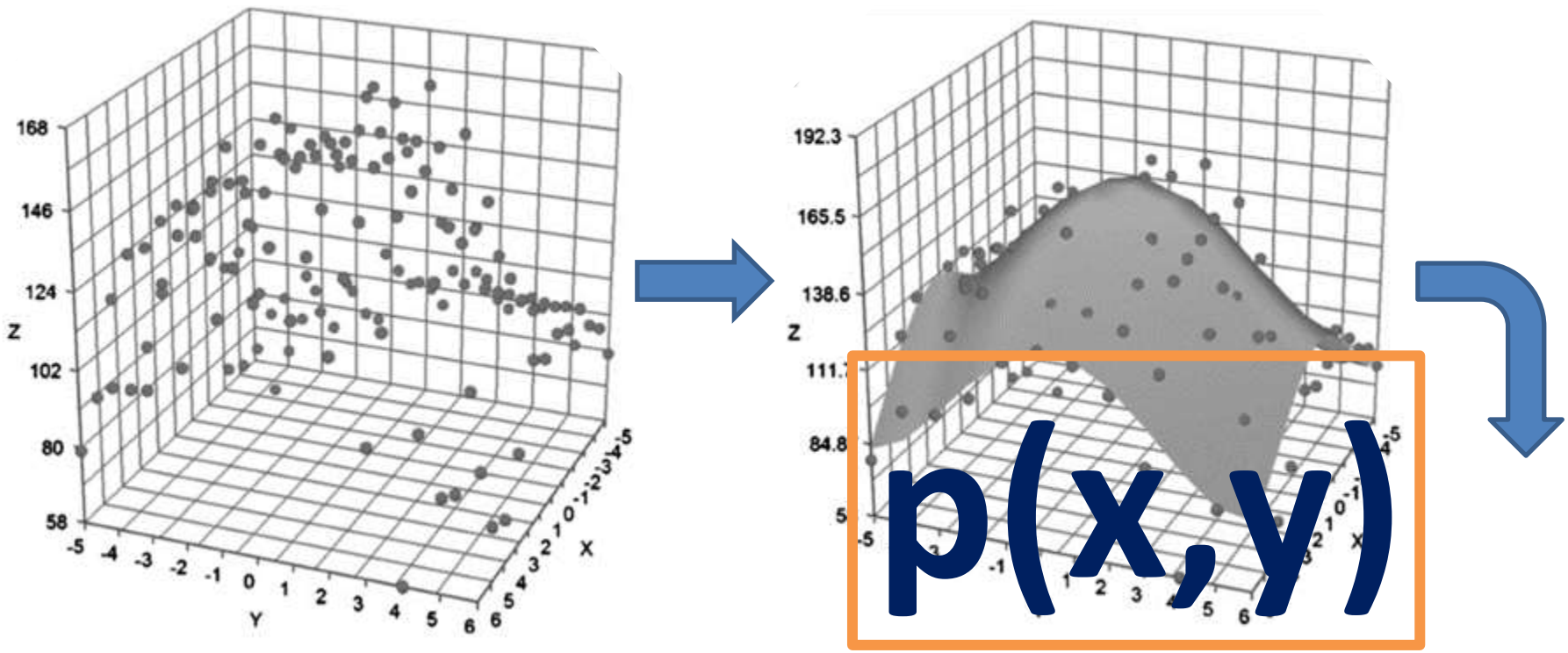
- Given a (sampled) set of values, we interpolate the datapoints to define a polynomial that would represent the data.





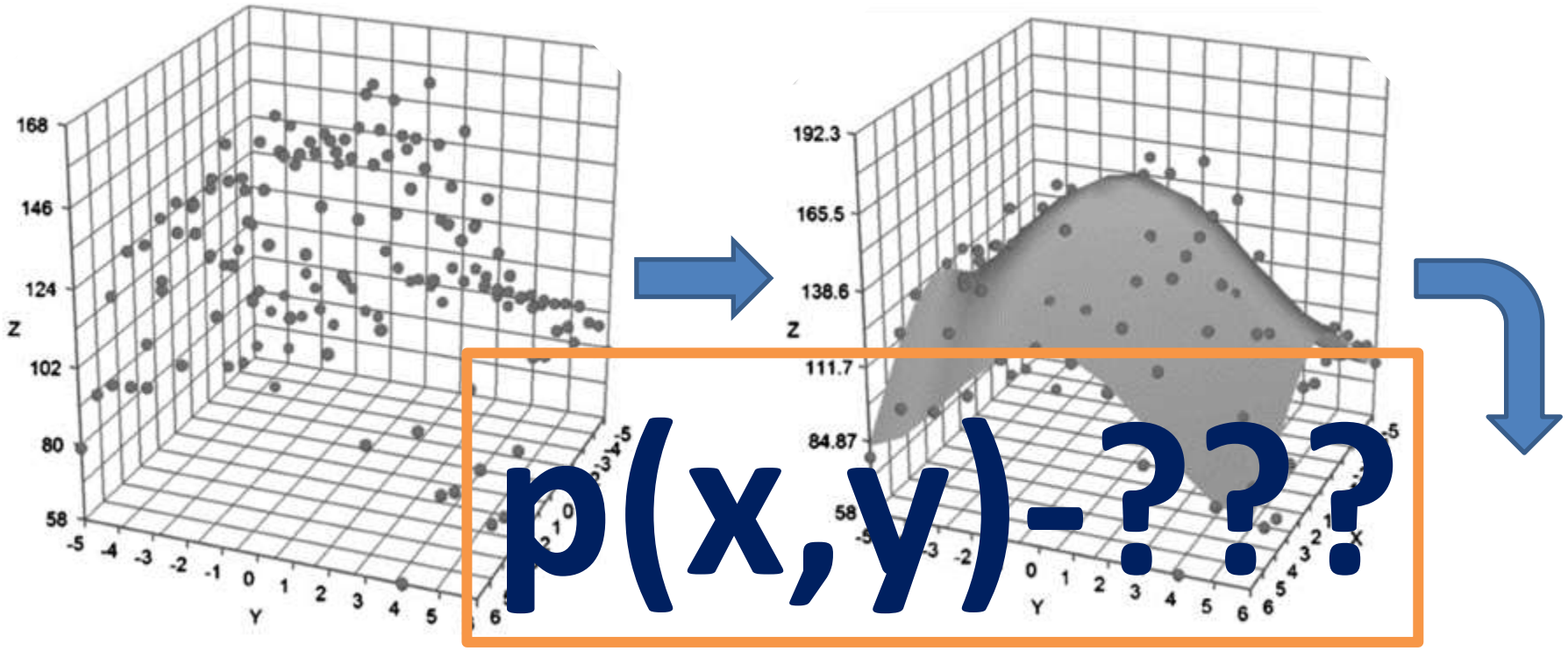
p(x,y)

Distributed Big Data Interpolation



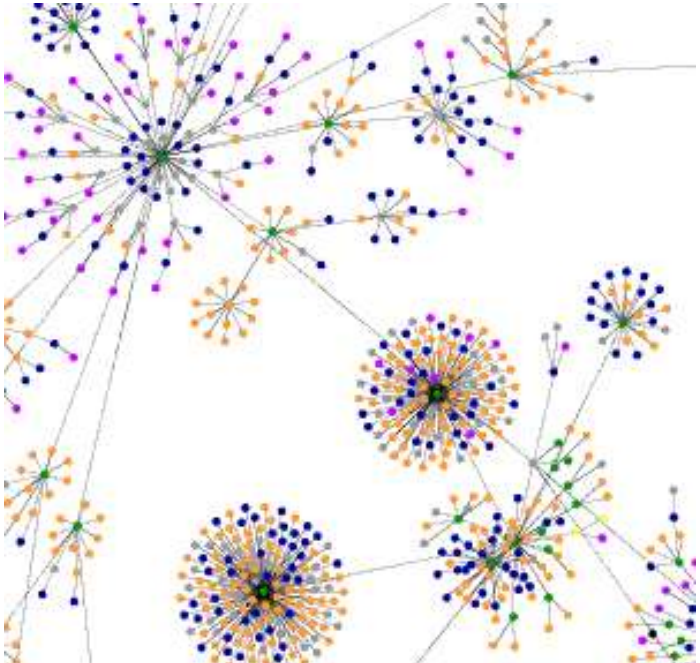
Distributed Big Data Interpolation

- **Weierstrass approximation Theorem:** for any given $\varepsilon > 0$, there exists a polynomial p such that $\|p - f\|_\infty \leq \varepsilon$



Distributed Big Data Interpolation

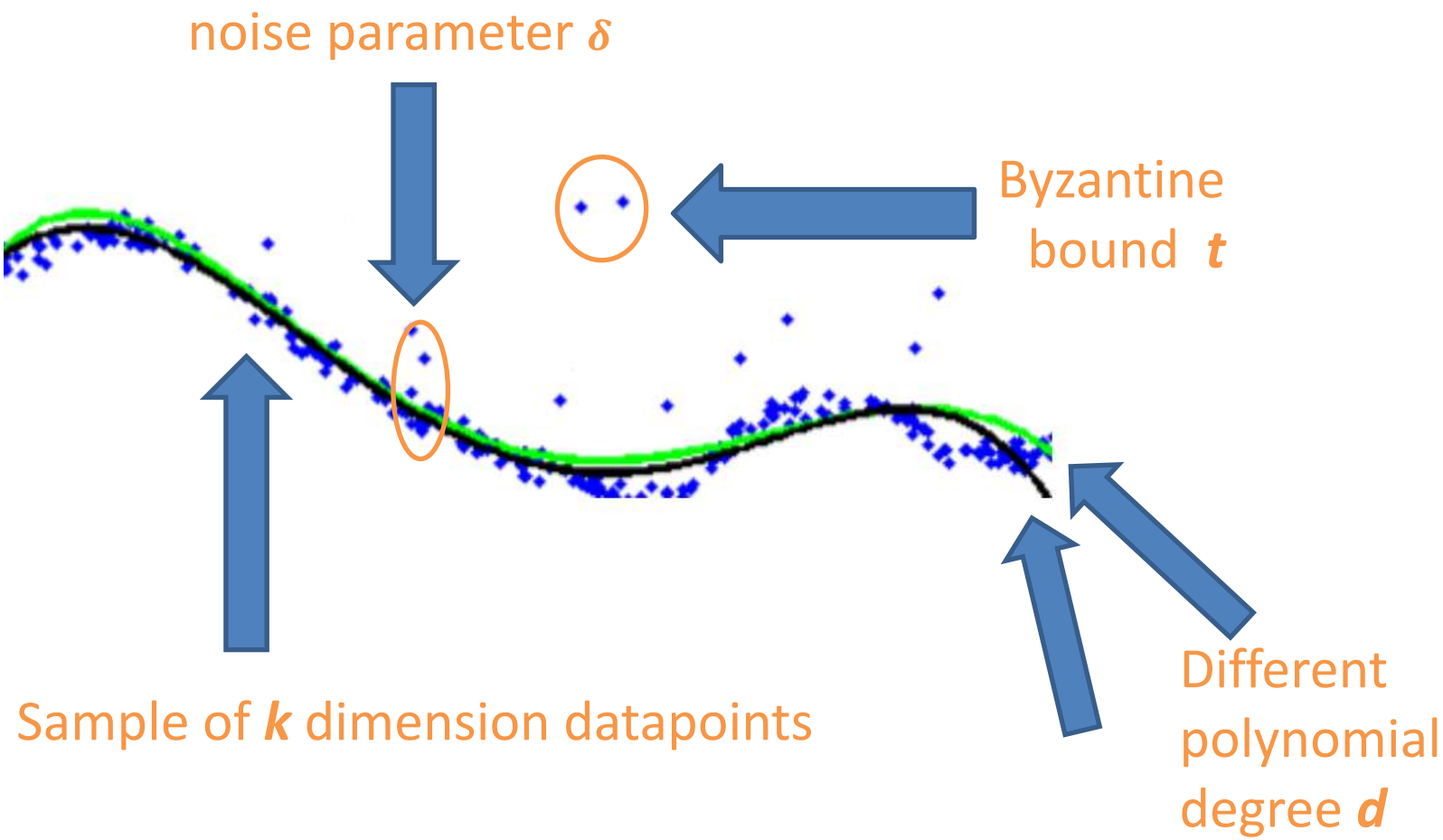
- The interpolation task would carried out by local **data centers.**
- The local polynomials are merged to a global one by interpolation in a **hierarchical manner**.



Challenges

- 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.

Polynomial Fitting to Noisy and Byzantine Data



$p_d(\mathrm{x}) = y \pm \delta$ **for at least** N**-** t **points**

Definition: Polynomial Fitting to Noisy and Byzantine Data problem

Given a sample S of k dimension datapoints $\{(x_{1_i}, \dots, x_{k_i})\}_{i=1}^{N}$ and a function f defined on those points

$f(x_{1_i}, \dots, x_{k_i}) = y_i$, a noise parameter $\delta > 0$, and Byzantine bound t

we have to find a polynomial p of total degree d satisfying:

$p(x_1, \dots, x_k) \in [y - \delta, y + \delta]$ for at least N- t points

Polynomial Fitting to Noisy and Byzantine Data

Polynomial Fitting to Noisy and **Byzantine** Data

Error Correcting Code approach:

- Byzantine elimination via polynomial division.
- Handle multidimensional <u>general</u> data
- Tolerated to <u>discrete</u>-noise and Byzantine appearance.

Polynomial Fitting to **Noisy** and Byzantine Data

Error Correcting Code approach:

- Byzantine elimunation via polynomial division
- Handle multidimensional <u>general</u> data
- Tolerated to <u>discrete</u>-noise and Byzantine appearance

Curve-fitting & approximation approach:

- Noise decreasing using linear programming.
- Handle <u>random</u> sample with <u>unrestricted</u> noise.

Outline

- Introduction
  - Motivation
  - Problem definition
- General data with Discrete Finite Noise
  - Welsh & Berlekamp Algorithm
  - Multidimensional Reconstruction
- Random Sample with Unrestricted Noise
  - Noise decreasing using linear programming
  - Byzantine elimination
- Conclusions

Outline

- Introduction
  - Motivation
  - Problem definition
- General data with Discrete Finite Noise
  - Welsh & Berlekamp Algorithm
  - Multidimensional Reconstruction
- Random Sample with Unrestricted Noise
  - Noise decreasing using linear programming
  - Byzantine elimination
- Conclusions

Welch and Berlekamp (WB) Algorithm

[Welch & Berlekamp, 1986]

Welch and Berlekamp (WB) Algorithm

- Handle Byzantine data
- No noise
- Using *error-locating polynomial*, $\boldsymbol{e}$.
- $\boldsymbol{e(x_i) = 0}$ whenever $\boldsymbol{p(x_i) \neq y_i}$.
- defining the polynomial $\boldsymbol{q(x) = p(x)e(x)}$
- solve $\boldsymbol{q(x_i) = y_i e(x_i)}$ for all i
- $p(x)$ can be found by $\boldsymbol{p(x) = q(x) / e(x)}$

[Welch & Berlekamp, 1986]

WB Algorithm : 2D data

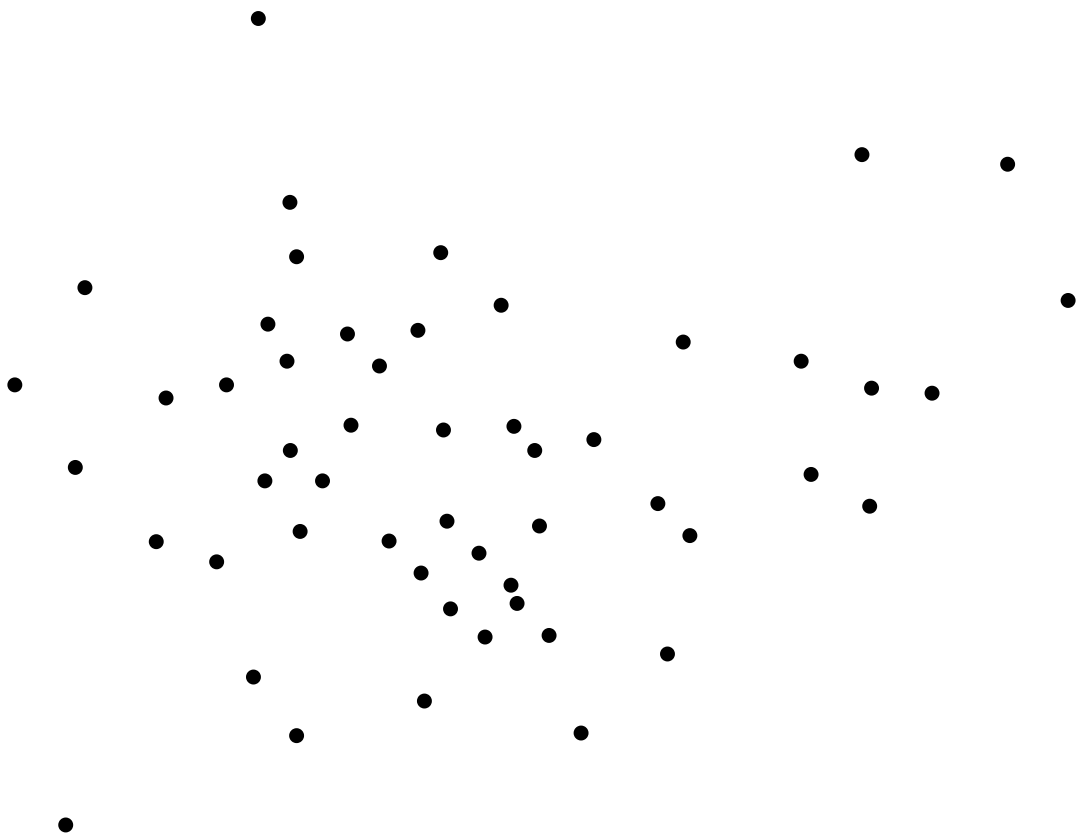
↓

3D polynomial reconstruction

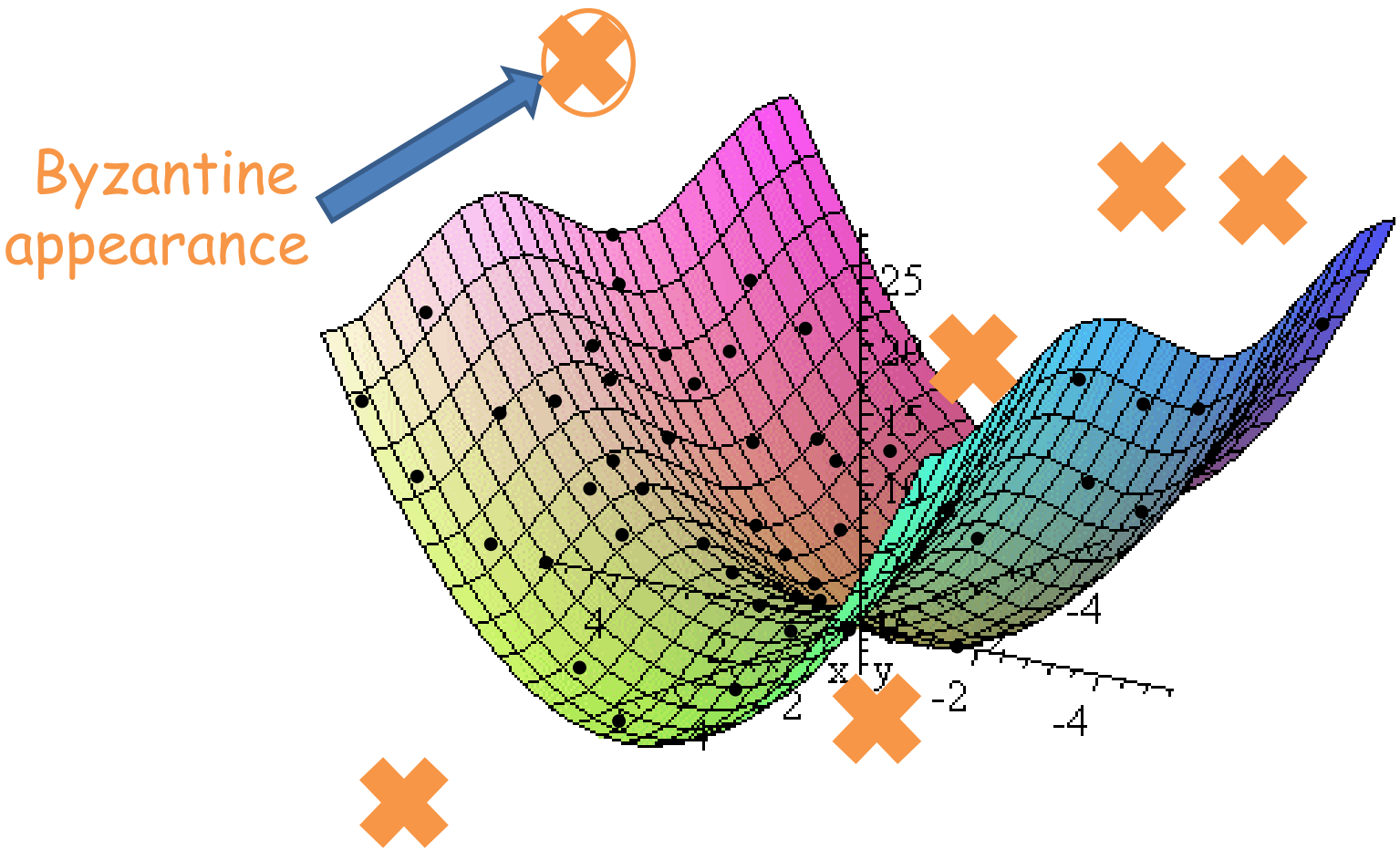
↓

Multidimensional data reconstruction

3D polynomial reconstruction



3D polynomial reconstruction



3D polynomial reconstruction

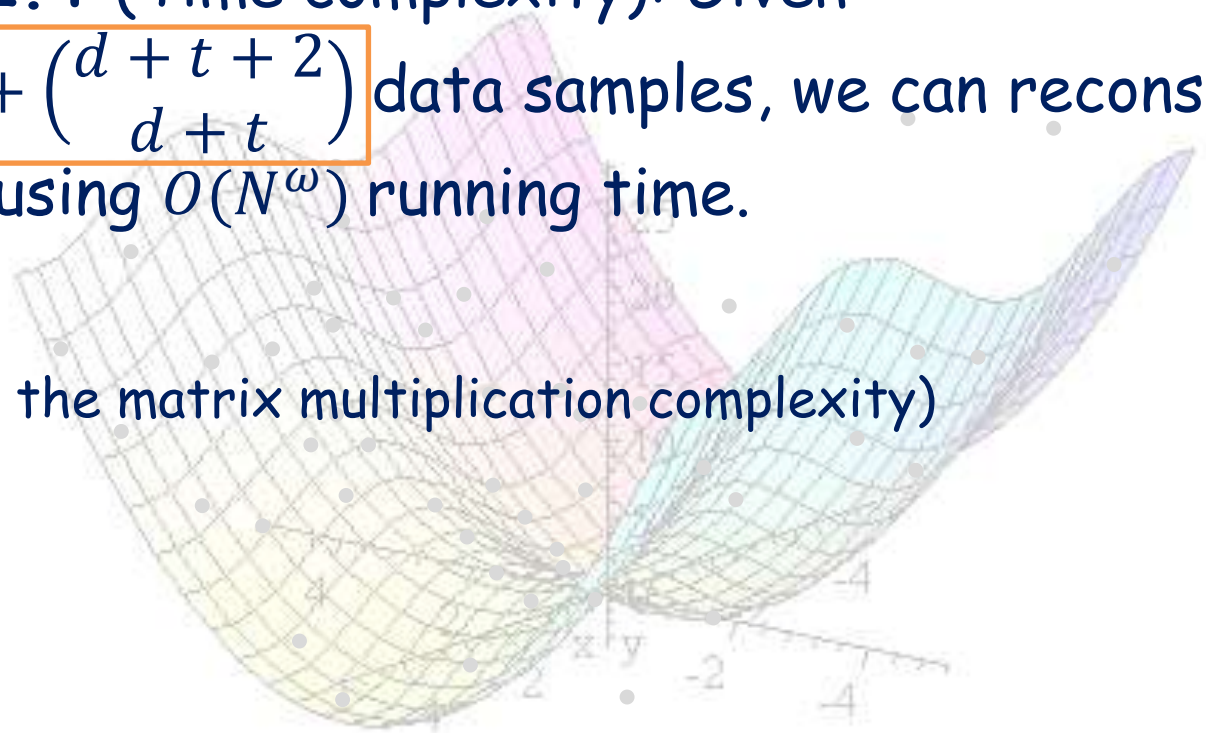
- **Input**: $t, d, \{(x_i, y_i, z_i)\}_{i=1}^{N}$
- **Output**: $p(x, y)\ \deg(p) = d$
- **Step 1**: compute $e(x), q(x, y)$ $(\deg(e) = t, \deg(q) = d + t)$ by solving:

  $$q(x_i, y_i) = z_i e(x_i)\ 1 \leq i \leq N$$
- **Step 2**: $p(x, y) = q(x, y)/e(x)$

3D polynomial reconstruction

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

(where ω is the matrix multiplication complexity)



3D polynomial reconstruction

- **Proof**: m variate polynomial with degree d
- $\Longrightarrow$ $\binom{d+m}{d}$ terms.
- $\Longrightarrow$ Necessary to have $\binom{d+t+2}{d+t}$ distinct points.

- **Step 1:** We have N linear equation in at most N variables, which we can be solve e.g., by Gaussian elimination in time $O(N^{\omega})$.

- **Step 2:** The general problem -can be done using the Gröbner base.
- Since the divider is a *univariate polynomial*, we can mimic long division
- can be implemented in $O(NlogN)$ running time

3D polynomial reconstruction

↓

Multidimensional data reconstruction

3D polynomial reconstruction

- e and q are x-variate polynomial
- Using Gröbner bases we can implement the polynomial division at close to $O(NlogN)$ time
- Noise: dismiss it by consistently insert a vector of possible noise, reconstruct the polynomial, and test it by the original dataset S.

Multidimensional data reconstruction

Outline

- Introduction
  - Motivation
  - Problem definition
- General data with Discrete Finite Noise
  - Welsh & Berlekamp Algorithm
  - Multidimensional Reconstruction
- Random Sample with Unrestricted Noise
  - Noise decreasing using linear programming
  - Byzantine elimination
- Conclusions

Outline

- Introduction
  - Motivation
  - Problem definition
- General data with Discrete Finite Noise
  - Welsh & Berlekamp Algorithm
  - Multidimensional Reconstruction
- Random Sample with Unrestricted Noise
  - Noise decreasing using linear programming
  - Byzantine elimination
- Conclusions

Random Sample with Unrestricted Noise

- Most research has used the L_2 norm of noise (LS).
- Not suffice the adversarial noise
- Extend Arora & Khot (2002) to handle L_∞ noise

Random Sample with Unrestricted Noise

Small noise at every point

&

large noise occasionally

Too many polynomials agreeing with the given data.

- Thus, our goal is to find a polynomial p that is δ-approximation of f

$$\|p - q\|_\infty \leq \delta$$

Random Sample with Unrestricted Noise

- Given a random sample $\{(x_i, y_i, f(x_i, y_i) = z_i\ )\}_{i=1}^{N}$
- We assume by rescaling the data that each $x_i, y_i, z_i \in [-1,1]$.
- Define a linear programming system (LP) with the fitting polynomial as its solution.

Random Sample with Unrestricted Noise

Noise parameter

$$\text{minimize } \delta$$

$$s.t.$$

move to Chebyshev's representation of the polynomial- $T_i(\cdot), T_j(\cdot)$

$$f(x_k, y_k) - \delta \leq \sum_i^n \sum_j^m \boxed{c_{ij}} T_i(x_k) T_j(y_k) \leq f(x_k, y_k) + \delta, \quad k = 1, ..., |S|$$

each of its coefficients is at most $\sqrt{2}$ due to Chebyshev

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

$$|\sum_i^n \sum_j^m c_{ij} T_i(x) T_j(y)| \leq 1, \quad \forall_{x,y} \in I$$

Random Sample with Unrestricted Noise

- the output of the LP minimization p is the respected δ-approximation of f

$$i.e., \|f - p\| \leq \delta$$

Random Sample with Unrestricted Noise

- Bernstein-Markov Theorem applies

$$\|(p-f)'\|_\infty \le O(d^2)$$

- Let ε denote the largest distance between two successive points $(x_1, y_1), \dots, (x_{|S|}, y_{|S|})$
- Every interval of size ε contains at least one of the datapoints (forming ε-net).
- With high probability $\varepsilon = O\left(\frac{\log|S|}{|S|}\right) = O(\delta/d^2)$
- Due to the LP constraint p, f differ by at most δ on the points in the ε -net,
- $\|p-f\|_\infty \le 2\delta + O(\varepsilon d^2) = c\delta$
- p is the respected δ-approximation of f

Outline

- Introduction
  - Motivation
  - Problem definition
- General data with Discrete Finite Noise
  - Welsh & Berlekamp Algorithm
  - Multidimensional Reconstruction
- Random Sample with Unrestricted Noise
  - Noise decreasing using linear programming
  - Byzantine elimination
- Conclusions

Byzantine elimination

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 $p(x, y)$

For any point, consider a small sqaure interval Λ.

Due to the derivative bound, the true value of the polynomial is essentialy constant over Λ.
$\Rightarrow$ we can eliminate the byzantine appearance.

Outline

- Introduction
  - Motivation
  - Problem definition
- General data with Discrete Finite Noise
  - Welsh & Berlekamp Algorithm
  - Multidimensional Reconstruction
- Random Sample with Unrestricted Noise
  - Noise decreasing using linear programming
  - Byzantine elimination
- Conclusions

Outline

- Introduction
  - Motivation
  - Problem definition
- General data with Discrete Finite Noise
  - Welsh & Berlekamp Algorithm
  - Multidimensional Reconstruction
- Random Sample with Unrestricted Noise
  - Noise decreasing using linear programming
  - Byzantine elimination
- Conclusions

General Byzantine data with Discrete Finite Noise

- Solving linear system
- Polynomial division
- $N = t + \binom{d+t+2}{d+t}$
- constant δ

Random Byzantine Sample with Unrestricted Noise

- LP minimization
- $N = \frac{d^4}{\delta} log\left(\frac{1}{\delta}\right)$

Outline

- Introduction
  - Motivation
  - Problem definition
- General data with Discrete Finite Noise
  - Welsh & Berlekamp Algorithm
  - Multidimensional Reconstruction
- Random Sample with Unrestricted Noise
  - Noise decreasing using linear programming
  - Byzantine elimination
- Conclusions

Conclusions

- Presented the concept of **data interpolation** in the scope of sensor **data aggregation** as well as the new **big data** challenge.

Conclusions

- Constructs a polynomial using the WB method as a subroutine.
- Tolerated to discrete-noise and Byzantine multi-dimensional data.
- Presented a multivariate analogue of the WB method.
- Using linear programing minimization we reconstruct an unknown multi-dimensional polynomial.
- Detail the way to eliminate the Byzantine appearance.

Thank you…

e is multivariate or univariate

- Given that p has m=2 variable, deg(p)=1
- the data contain t = 2 Byzantine appearance

(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)

- When e univariate:

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

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

- When e is bivariate:

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

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

- Both give the same expected solution:

$$q_1(x,y)/e_1(x,y) = q_2(x,y)/e_2(x,y) = x + y = p(x)$$

back

Random Sample with Unrestricted Noise

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

- proof: Since $\|T_i(x)\|_\infty = \|T_j(y)\|_\infty = 1$

using Bernstein-Markov theorem

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

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

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

We get $|T_i'(x)| = O(i^2)$

thus:

$$|p_x'| \leq \sum_i^n \sum_j^m |a_{ij}| \Big(T_i(x)T_j(y)\Big)'_x$$

$$\leq \sum_i^n \sum_j^m \sqrt{2} O(i^2) \leq O(n^3 m)$$

Random Sample with Unrestricted Noise

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

- From symmetric consideration $|p'_y| \leq O(nm^3)$
- By construction, p takes all values in [-1,1] for all points in I, and the distance between successive points of I is $2/|I|$ (I is equidistant).
- The claim follows from the fact that the derivative p' by denition gives the rate of change in p

Random Sample with Unrestricted Noise

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

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

3D polynomial reconstruction

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:

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

If such polynomials do not exist, output fail.

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

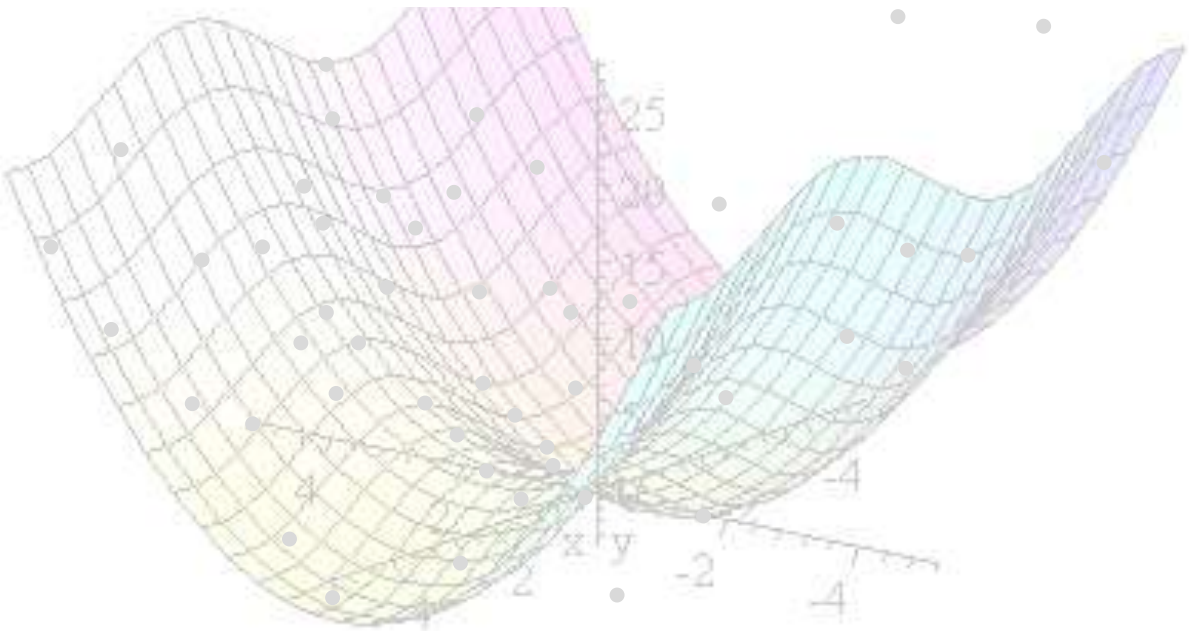
proof: If $e(x_i) = 0$, then $q(x_i, y_i) = z_i e(x_i) = 0$.

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.

3D polynomial reconstruction

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

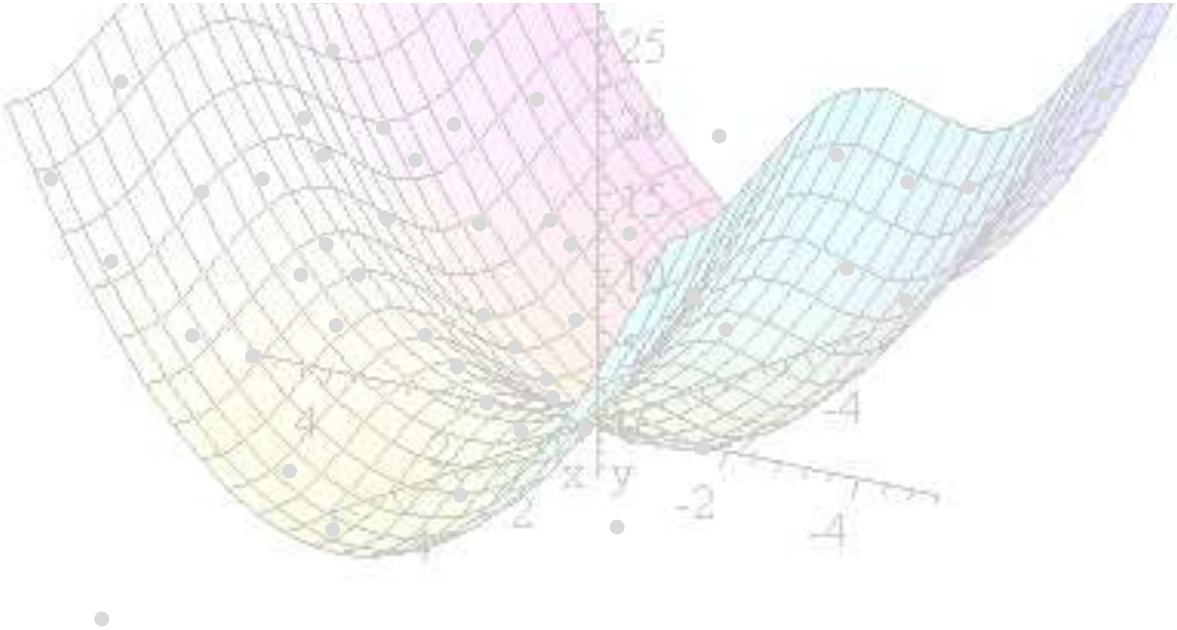
- **Input:** $0 < t = \rho N$ which is the Byzantine appearance bound, the total degree $d > 1$ of the goal polynomial and N triples $(x_i, y_i, z_i)_{i=1}^{N}$ with distinct x_i's .



3D polynomial reconstruction

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.



3D polynomial reconstruction

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:

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

If such polynomials do not exist, output fail.

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

3D polynomial reconstruction

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 $\boxed{p(x, y) = \frac{q(x, y)}{e(x)}}$. If $\Delta(z_i, p(x_i, y_i)_i) > t$, output fail. else output $p(x, y)$.

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

$$q_1(x, y)/e_1(x) = q_2(x, y)/e_2(x)$$

3D polynomial reconstruction

- **Claim 2.2** (Correctness): There exist a pair of polynomials $e(x)$ and $q(x, y)$ that satisfy **Step 1** such that $q(x, y) = p(x, y)e(x)$
- **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 $q_1(x, y)/e_1(x) = q_2(x, y)/e_2(x)$