Fourier Transform and Other Quadratic Problems under Interval Uncertainty

Oscar Galindo[1], Christopher Ibarra[1],
Vladik Kreinovich[1], and Michael Beer[2]
[1]Department of Computer Science
University of Texas at El Paso
500 W. University, El Paso, Texas 79968, USA
{ogalindomo,caibarra5}@miners.utep.edu, vladik@utep.edu
[2]Institute for Risk and Reliability, Leibniz University Hannover
30167 Hannover, Germany, beer@irz.uni-hannover.de

1. Need for data processing

- Computers are used:
  - to estimate the current values of physical quantities and
  - to predict their future values,
  - e.g., to predict tomorrow's temperature.
- In all these cases, we need to process data.

2. Need to take uncertainty into account

- The inputs $x_1, \ldots, x_n$ for such data processing come from measurements (or from expert estimates).
- Both measurements and expert estimates are not absolutely accurate.
- Measurement results $\widetilde{x}_i$ are, in general, somewhat different from the actual (unknown) values x_i of the corresponding quantities.
- These differences $\widetilde{x}_i - x_i$ are called *measurement errors*.
- Because of these differences:
  - the result $\widetilde{y} = f(\widetilde{x}_1, \ldots, \widetilde{x}_n)$ of data processing is also somewhat different from
  - the value $y = f(x_1, \ldots, x_n)$ that we would have obtained if we knew the exact values x_i of the inputs.

3. Need for interval uncertainty

- In many practical situations:
  - the only information that we have about measurement uncertainty is
  - the upper bound Δ_i on the absolute value of each measurement error.
- In such situations:
  - if the measurement result is $\widetilde{x}_i$,
  - then all we know about the actual value x_i of the corresponding quantity is that this value is in the interval $[\widetilde{x}_i - \Delta_i, \widetilde{x}_i + \Delta_i]$.
- Under such interval uncertainty, it is desirable to know the range of possible value of y.

4. Interval uncertainty: what is known and what we do

- In general, computing such a range is NP-hard already for quadratic functions $f(x_1, \ldots, x_n)$.
- Recently, a feasible algorithm was proposed for a practically important quadratic problem.
- This was a problem of estimating the absolute value (modulus) of Fourier coefficients.
- In this talk, we show that this feasible algorithm can be extended to a reasonable general class of quadratic problems.

5. Class of Quadratic Expressions for Which the Range Can Be Feasibly Computed

- A general quadratic function has the form

$$f = \sum_{i=1}^{n} \sum_{j=1}^{n} c_{i,j} \cdot x_i \cdot x_j + \sum_{i=1}^{n} c_i \cdot x_i + c_0.$$

- An important characteristic of the matrix $c_{i,j}$ is its *rank* – the number of non-zero eigenvalues.
- When we compute the square of the modulus of the Fourier coefficient, the rank of the corresponding matrix is 2.
- The general class is when the matrix $c_{i,j}$ has rank k, i.e., that it has k non-zero eigenvalues λ_j, $j = 1, \ldots, k$.
- We will denote the corresponding unit eigenvectors by $(e_{j,1}, \ldots, e_{j,n})$.

6. Our result

- We prove that for any fixed k, there is a feasible algorithm for estimating the range of the corresponding quadratic expression.
- This algorithm takes time $O(n^k)$ in the homogeneous case and $O(n^{k+1})$ in the general case.
- So, as k increases, the time grows fast, and for $k \approx n$, we get exponential time.
- This makes sense: since the problem is NP-hard, we cannot expect lower-than-exponential computation time.

7. Facts from Calculus: Reminder

- Computing the minimum of f is equivalent to computing the maximum of $-f$.
- Thus, it is sufficient to be able to compute the maximum.
- According to calculus, the maximum with respect to each variable $x_i \in [\underline{x}_i, \overline{x}_i]$ is attained:
  - either for $x_i = \underline{x}_i$, then $\dfrac{\partial f}{\partial x_i} \le 0$;
  - or for $x_i = \overline{x}_i$, then $\dfrac{\partial f}{\partial x_i} \ge 0$;
  - or for $x_i \in (\underline{x}_i, \overline{x}_i)$, then $\dfrac{\partial f}{\partial x_i} = 0$.

8. Let Us Apply These Facts to Our Problem

- We start with the quadratic expression

$$f=\sum_{i=1}^n \sum_{j=1}^n c_{i,j}\cdot x_i\cdot x_j+\sum_{i=1}^n c_i\cdot x_i+c_0.$$

- In terms of eigenvalues and eigenvectors, the quadratic expression takes the form

$$f=\sum_{j=1}^k \lambda_j\cdot\left(\sum_{i=1}^n e_{j,i}\cdot x_i\right)^2+\sum_{i=1}^n c_i\cdot x_i+c_0.$$

- Its partial derivative w.r.t. x_i is equal to:

$$\frac{\partial f}{\partial x_i}=2\sum_{j=1}^k \lambda_j\cdot\left(\sum_{\ell=1}^n e_{j,\ell}\cdot x_\ell\right)\cdot e_{j,i}+c_i.$$

- This expression can be described in terms of n $(k+1)$-dimensional vectors

$$e_i=(e_{1,i},\ldots,e_{k,i},c_i) \text{ and } e_i^*=(2\lambda_1\cdot e_{1,i},\ldots,2\lambda_k\cdot e_{k,i},0).$$

9. Let Us Apply These Facts to Our Problem (cont-d)

- In terms of the dot (scalar) product, we get $\dfrac{\partial f}{\partial x_i} = e_i \cdot S$, where:

$$S \stackrel{\text{def}}{=} \sum_{\ell=1}^{n} x_\ell \cdot e_\ell^* + (0, \ldots, 0, 1).$$

- Thus, all the $(k+1)$-dimensional points e_i for which $\dfrac{\partial f}{\partial x_i} = 0$ are located on a k-dimensional plane $\{e : e \cdot S = 0\}$.
- Let us first consider the non-degenerate case, when every group of $k+1$ vectors e_i is linearly independent.
- We can have no more than k linearly independent vectors on the same k-dimensional plane.
- Thus, we can have no more than k indices i for which partial derivative is 0.

10. Let Us Apply These Facts to Our Problem (cont-d)

- For points on one side of the plane, where $\dfrac{\partial f}{\partial x_i} < 0$, maximum is attained for $x_i = \underline{x}_i$.
- For points on the other side of the plane, where $\dfrac{\partial f}{\partial x_i} > 0$, maximum is attained for $x_i = \overline{x}_i$.
- If there are fewer than k points at which the derivative is 0, we can move the plane a little bit until it reaches exactly k points.
- So, we arrive at the following algorithm.

11. Resulting Algorithm: Non-Degenerate Case

- *Given:*
  - a quadratic expression with matrix of rank k:
$$f=\sum_{i=1}^n\sum_{j=1}^n c_{i,j}\cdot x_i\cdot x_j+\sum_{i=1}^n c_i\cdot x_i+c_0;\ \text{ and}$$
  - intervals $[\underline{x}_i,\overline{x}_i]$.
- *Find:* the range $[\underline{y},\overline{y}]$ of the expression f.
- We consider all possible selections $1\le i_1<\ldots<i_j<\ldots<i_k\le n$ of k different indices.
- There are $O(n^k)$ such selections.
- For each selection, we solve a system of k linear equations with k unknowns $S_1,\ldots,S_k$:
$$\sum_{j'=1}^k e_{j',i_j}\cdot S_{j'}+c_{i_j}=0,\quad j=1,\ldots,k.$$

12. Algorithm, Non-Degenerate Case (cont-d)

- We consider all 3^k possible divisions of the set $\{1,\ldots,k\}$ into 3 subsets L (lower), U (upper), and I (inside).
- For each division, we consider two possible signs $\varepsilon \in \{-,+\}$.
- For each division and sign:
  - we set $x_i = \underline{x}_i$ if $(e_i \cdot S < 0$ and $\varepsilon = +)$ or $(e_i \cdot S > 0$ and $\varepsilon = -)$;
  - we set $x_i = \overline{x}_i$ if $(e_i \cdot S > 0$ and $\varepsilon = +)$ or $(e_i \cdot S < 0$ and $\varepsilon = -)$;
  - we set $x_{i_j} = \underline{x}_{i_j}$ for $j \in L$ and $x_{i_j} = \overline{x}_{i_j}$ for $j \in U$;
  - the remaining values x_{i_j} for $j \in I$, from the system of equations:

$$\frac{\partial f}{\partial x_{i_j}} = 2\sum_{j'=1}^{k} \lambda_{j'} \cdot \left(\sum_{\ell=1}^{n} e_{j',\ell} \cdot x_\ell\right) \cdot e_{j',i_j} + c_{i_j} = 0, \quad j = 1,\ldots,k;$$

  - if the resulting values x_{i_j} are in $[\underline{x}_{i_j}, \overline{x}_{i_j}]$, then we compute the value $f(x_1,\ldots,x_n)$.

13. Algorithm, Non-Degenerate Case (cont-d)

- The largest of the corresponding values of the expression f is $\overline{y}$, the smallest is $\underline{y}$.
- Computing f by using eigenvectors takes time $O(n \cdot k) = O(n)$.
- We perform it for all $O(n^k) \cdot 2 \cdot 3^k = O(n^k)$ cases, so overall time is $O(n^{k+1})$, which is feasible.

14. General Case

- For each $\delta > 0$, we can add δ-small random changes to the values c_{ij} and c_i.
- For example, we can add values uniformly distributed on the interval $[-\delta, \delta]$.
- With probability 1, the resulting system is non-degenerate.
- The difference between the original and new objective functions does not exceed
$$\delta \cdot \left(\sum_{i=1}^{n} \sum_{j=1}^{n} |x_i| \cdot |x_j| + \sum_{i=1}^{n} |x_i| \right).$$
- We can use straightforward interval computations to get the bound B on the expression in parentheses.
- So, for any given $\varepsilon > 0$, if we take $\delta = \varepsilon / B$, we get a non-degenerate objective function which is ε-close to the original one.

15. General Case (cont-d)

- The bounds for the new objective function are ε-close to the bounds on the original one.
- Thus, we have a feasible $O(n^{k+1})$ algorithm for computing $\underline{y}$ and $\overline{y}$ with any given accuracy $\varepsilon > 0$.

16. Homogeneous Case

- In the Fourier transform case, $c_i = 0$, so $f = \sum_{i=1}^{n} \sum_{j=1}^{n} c_{i,j} \cdot x_i \cdot x_j + c_0$.
- In such *homogeneous* case, we can consider k-dimensional vectors

$$e_i = (e_{1,i}, \ldots, e_{k,i}) \text{ and } e_i^* = (2\lambda_1 \cdot e_{1,i}, \ldots, 2\lambda_k \cdot e_{k,i}).$$

- In non-degenerate case, we thus have $\le k-1$ indices i at which the derivative is 0.
- So, we have a similar algorithm, but with $k-1$ instead of k.
- This algorithm requires time $O(n^k)$.

17. Acknowledgments

- This work was supported in part by the National Science Foundation grants:
  - 1623190 (A Model of Change for Preparing a New Generation for Professional Practice in Computer Science), and
  - HRD-1834620 and HRD-2034030 (CAHSI Includes).
- It was also supported by the AT&T Fellowship in Information Technology.
- It was also supported by the program of the development of the Scientific-Educational Mathematical Center of Volga Federal District No. 075-02-2020-1478.