Exact Computations as Background for Numerical Methods

Valentin Golodov
School of Electrical Engineering and Computer Science
South Ural State University
Chelyabinsk, Russia
golodovva@susu.ru

***Abstract*—This paper continues a couple of author's papers devoted to development of the exact rational computing and its applications in numerical methods of analysis and solving ill-conditioned tasks. Examples of application of the exact rational computing in numerical methods of solving ill-conditioned tasks are presented. Ill-conditioned linear equation systems, linear equation systems with interval uncertainty of the coefficients are examples of such tasks. Application of the interval regularization procedure to the Firordt method of the spectrophotometric analysis of the non-separated mixtures is one examples where exact computing allows improving of the result robustness.**

***Index Terms*—Exact computing, interval regularization, parallel algorithms.**

I. Introduction

This paper continues a couple of author's papers devoted to development of the exact rational computing [1] and its applications in numerical methods of analysis and solving ill-conditioned tasks [2], [3].

A lot of papers are dedicated to the requirements that scientists impose to numerical methods. Computational complexity, efficiency, implementation simplicity are objects of almost every paper describing some new numerical method or modification of the existing one. But small amount of the papers devoted to requirements to number representation or number precision. Number precision is very important part of the successful numerical methods application, if in some moment intermediate result goes out from area of guaranteed precision of the number presentation then it is impossible to ensure required precision or even correctness of the aggregate result [4].

II. Library of Classes Exact Computation

Current version is third version of the library. Previous implementations was developed in 1999 as simple overlong and rational C++ classes [5] with few distributed computing abilities that GMP [6] had not being provided.

Version 2.0 [7] implemented in 2013 was presented in the work [2], library architecture was redesigned and library code was refactored.

Version 1.0 and 2.0 based on positional big number representation with base 2^{16} and 2^{32} correspondingly, this bases are preferred to use advantages of 32 and 64-bit operating systems.

Most important feature implemented in version 2.0 is CUDA capable GPUs support since version 2.0 rational computing also available on the GPUs.

New refactored result version 3.0 provides *rational* and *overlong* classes for CPU and GPU architectures and simple interface. Also user can implement own number representation (only basic arithmetic operations) and test it in application without huge coding overhead.

There are a lot of scientific researches dealing with some arithmetic algorithms. Testing of the correctness may be performed independently from algorithm's practical application but important issues related with overhead during applications are often hard to predict.

Applications usually operates with numbers irrelatively of its representation, so single interface of the arithmetic operation may have independent implementations.

Applications are very important motive force of the exact computing library development. Effective realization of different algorithms requires various sets of basic functions. Simultaneous development of numerical methods and exact computing library in big project allows improving the weak parts and designing of new library features.

Next section presents cases of successful deployment of exact rational computing in applications.

III. Application of Rational Arithmetic for Scientific Tasks

In this section, we will consider the applications of the arbitrary precision arithmetic in certain tasks.

A. Ill-conditioned Linear Equation Systems

First of all, ill-conditioned problems such as solving linear and non-linear systems with almost singular or ill-conditioned matrices [8]. Commonly known example of such matrix is Hilbert matrix.

$$H = \begin{bmatrix} 1 & \frac{1}{2} & \frac{1}{3} \\ \frac{1}{2} & \frac{1}{3} & \frac{1}{4} \\ \frac{1}{3} & \frac{1}{4} & \frac{1}{5} \end{bmatrix} \tag{1}$$

The work was supported by Act 211 Government of the Russian Federation, contract 02.A03.21.0011.

Hard problem in exact computing is uncertainty in input data. Exact computing of the solution of the uncertain problem is meaningless procedure [9]. Uncertain problems requires another approaches such Lavrentiev's normal pseudosolution for the linear equation system [9], Tikhonov regularization procedure [10] or interval regularization approach proposed in [2], [3]. Aim of these solution procedures is decreasing influence on input uncertainty on the computed result, because of calculating errors are equivalent to infusion supplementary uncertainty in the input data.

Some new approach to ill-conditioned linear system of equations was proposed in [2], it was named *Interval Regularization Approach*, it uses interval analysis advancements to linear equation systems with interval uncertainty of the coefficients.

B. Interval Regularization Approach

Points of the tolerable solution set

$$\Xi_{tol}(\boldsymbol{A},\boldsymbol{b}) = \{x \in \mathbb{R}^n \mid (\forall A \in \boldsymbol{A})(\exists b \in \boldsymbol{b})(Ax = b)\}$$

of the interval linear equation system $\boldsymbol{A}_{m\times n}x = \boldsymbol{b}$ are considered as solutions of the initial ill-conditioned linear system $Ax = b$.

If solution does not exist ($\Xi_{tol} = \emptyset$) then minimal system right-hand part extension coefficient z^* may be found.

Point of the tolerable solution set may be found as solution of the linear programming task vector $x^* = x^{+^*} - x^{-^*}$, where x^{+^*} and $x^{-^*} \in \mathbb{R}^m$, $z^* \in \mathbb{R}$ are solution to the linear programming problem

$$\min_{x^+,\ x^-,\ z} z, \tag{2}$$

$$\sum_{j=1}^{m}(\underline{a}_{ij}x_j^+ - \overline{a}_{ij}x_j^-) \geq \underline{b}_i - zp_i,\ \ i = 1,\dots,m, \tag{3}$$

$$\sum_{j=1}^{m}(\overline{a}_{ij}x_j^+ - \underline{a}_{ij}x_j^-) \leq \overline{b}_i + zq_i,\ \ i = 1,\dots,m, \tag{4}$$

$$x_j^+, x_j^-, z \geq 0, \qquad j = 1,2,\dots,n \tag{5}$$

In addition, the vector $x^* = x^{+^*} - x^{-^*}$ belongs to $\Xi_{tol}(\boldsymbol{A},\boldsymbol{b}(z^*))$. Matrix $\boldsymbol{A} = [\underline{a}_{ij};\overline{a}_{ij}], i = 1,\dots,m, j = 1,\dots,n$, vectors $p, q \in \mathbb{R}^{+^m}$ are used to manipulate with form of the right-hand part extension.

Interval regularization approach uses simple method to solve linear programming task above, exact computations [7] and procedure described in [11] to prevent simplex method cycling, such synergy allows to solve ill-conditioned problems sensitive to the data precision.

C. Firordt Method of the Spectrophotometric Analysis of the Non-Separated Mixtures

The Firordt method is one of the methods of the analysis of the non-separated mixtures [12]. According to the Firordt's method, we can determine the concentration c_j of the each of the m components by solving the following system of the equations:

$$b_i = \sum_{j=1}^{m} a_{ij} \cdot c_j \cdot l, \tag{6}$$

where:

- b_i is the measured absorbancy of the analyzed mixture on the i-th analytical wave length(AWL),
- a_{ij} is an molar coefficient of the absorption (or extinction) of the j-th component on i-th AWL (measured in advance for each component),
- l is the thickness of the absorbing layer.

Number of the AWL(k) (number of the equations) usually is equal to the number of the components (m) in the mixture. Overdetermined systems with $k > m$ may be used for the enhanced accuracy.

Spectrophotometric measurements are always performed with some measurement errors, so, we have some imprecise system of linear algebraic equations for analysis with equations of the form (6).

$$\boldsymbol{b}_i = \sum_{j=1}^{m} \boldsymbol{a}_{ij} \cdot c_j \cdot l, \tag{7}$$

System (7) will become simpler if all measurements are performed using l equal to 1 centimeter, then system takes on the form

$$\boldsymbol{b}_i = \sum_{j=1}^{m} \boldsymbol{a}_{ij} \cdot c_j, \tag{8}$$

or, in matrix form, $\boldsymbol{A}x = \boldsymbol{b}$, where x – is the sought for vector of the components concentrations.

This uncertain linear equation system may be solved using interval regularization approach.

D. Number Theory and Arithmetic Algorithms

Every number presentation is simply sequence of byte in the device memory and every basic arithmetic algorithm is certain conversion of bits. This way wrapper for operations with numbers after being once implemented may be used to test different number representations and different arithmetic algorithms. Test of the new research issue leads to small changes in the existing library code that significantly reduces time of the research cycle and amount of possible bugs. New algorithm may be instantly checked on the existing code of the known application.

Big research area is parallel algorithms of the basic arithmetic operations, comparison, addition, multiplication, division, etc. Library provides very useful testing platform for newly developed arithmetic algorithms, allows testing one the application tasks without additional coding overhead.

All problems mentioned in section above requires arbitrary precision computing even in small tasks. Computing experiments with rational calculations will be introduced in section below.

IV. COMPUTING EXPERIMENTS

A. Linear Equation System with Hilbert Matrix

Abilities of the rational computations are demonstrated on the academic sample of the distributed computing realization of the Gauss method of solving linear equation system. Ill-conditioned matrices are the square Hilbert matrices (1) of different sizes. Examples are significant because standard double precision floating point can not provide sufficient precision even for very small dimensions about ten equations and ten variables [8].

Parallelism performed by cutting the matrix horizontally per one line, process with rank 0 handles lines 0, size, 2*size, etc., process with rank 1 handles lines 1, size+1, 2*size+1, etc. and so on.

Realization is academic and computing experiment demonstrates applying rational arithmetic to existing solution with changing only few lines of code. The applying also requires only linking with library files in standard way. Administrator account/skills, preliminary compilation or settings of the library are unneeded.

Linear equation systems was solved on rather weak Intel Core 2 Quad CPU Q8400, 2.66 GHz with different number of processes, see Table I. Calculation time demonstrates abilities of the exact rational computation even for non top computing systems.

B. Interval Regularization Approach

Consider the simple 2×2 system:

$$\begin{cases} (1+\varepsilon)x + y = 1, \quad \varepsilon \geq 0 \\ x + y = 1. \end{cases} \tag{9}$$

It has traditional solution $(x, y) = (0, 1)^\top$ for any $\varepsilon \neq 0$ and *normal pseudosolution* $(x, y) = (1/2, 1/2)^\top$ for $\varepsilon = 0$, consequently we have no convergence of the traditional solution to the normal pseudosolution when $\varepsilon \to 0$. For the nondegenerated systems pseudosolution is equal to its traditional solution.

Other well known method is the Tikhonov regularization procedure, with regard to the linear system of equations it leads to solving of the system $(A^\top A - \delta E)x = A^\top b$, where ($A^\top$ – is transposed matrix of the system, E – is unity matrix, δ – is the parameter of the regularization, selection of the δ is the theme of a lot of papers. However, e.g., for the system with Hilbert matrix the procedure doesn't leads to success [8].

Interval regularization approach solution of the nondegenerated system is equal to its solution in standard way. For the system (9) interval task is:

$$\begin{cases} [1; 1+\varepsilon]x + [1; 1]y = [1; 1], \ \varepsilon \geq 0 \\ [1; 1]x + [1; 1]y = [1; 1]. \end{cases} \tag{10}$$

And point $x = (0, 1)$ belongs to tolerable solution set.

C. Parallel Algorithms for Basic Arithmetic Operations

Algorithms of the basic arithmetic operations are important part effective numerical methods.

In the Table II experimental data of parallel addition on the GPU are provided. Data are obtained on the system with $Intel^{©}$Core i7–950 [3.06 GHz, 6 GB RAM]) and GPU of two different architectures: Fermi ($NVidia^{©}$GTX460 [700 MHz, 1 GB GDDR5]) and Kepler ($NVidia^{©}$GTX660 Ti [980 MHz, 2GB GDDR5]). Provided time is average of 1000 runs to reduce influence of the system tasks. Time of the parallel addition algorithm on the GTX460 GPU marked as Fer(P) and time on the GTX660 Ti marked as Kep(P).

D. Firordt Method of the Spectrophotometric Analysis of the Non-Separated Mixtures

Computing experiments of the interval analog of the are given in [3]. Interval regularization allows using all available data and gives robust solution as point of the tolerable solution set. Additional information about extension of the right-hand part of the interval system displays accuracy of the correlation between data and the Firordt method model.

V. CONCLUSION

A couple of methods based on exact computations was developed. Exact computing allows solving ill-conditioned tasks and improves robustness of the result. Simultaneous development of the numerical methods using exact computations and library features allows designing essential features necessary real numerical methods and applications.

REFERENCES

[1] V. Golodov, "Properties of mathematical number model provided exact computing," in *Proceedings of the 40th International Convention on Information and Communication Technology, Electronics and Microelectronics, MIPRO'2017, Opatija, Croatia, May 2226, 2017*. IEEE, 2017, pp. 242–245.

TABLE I. TIME OF EXACT SOLUTION OF LINEAR EQUATION SYSTEM ON CPU

Size	NProc	Time (Sec)
100	1	1.57
100	2	1.02
100	4	0.84
200	1	22.4
200	2	15.1
200	4	12.2
400	4	178
1000	4	7880

TABLE II. ADDITION TIME DEPENDING ON OPERANDS LENGTH

Length	Time in milliseconds			Speedup	
	CPU	Fer(P)	Kep(P)	Fer	Kep
10^1	0.18	0.24	0.24	0.004	0.004
10^2	0.20	0.24	0.24	0.004	0.004
10^3	0.62	0.24	0.24	0.004	0.004
10^4	3.04	0.24	0.24	0.004	0.004
10^5	4.01	0.54	0.70	0.925	0.714
10^6	42.2	0.88	0.86	6.818	6.976
10^7	409.2	4.30	2.76	14.0	21.814

[2] A. V. Panyukov and V. A. Golodov, “Computing best possible pseudo-solutions to interval linear systems of equations,” *Reliable Computing*, vol. 19, pp. 215 – 228, 2013, http://interval.louisiana.edu/reliable-computing-journal/volume-19/reliable-computing-19-pp-215-228.pdf.

[3] V. Golodov, “Interval regularization approach to the firordt method of the spectrophotometric analysis of the non-separated mixtures,” *Lecture Notes in Computer Science*, vol. 9553, pp. 201 – 208, 2013.

[4] R. Hammer, M. Hocks, U. Kulisch, and D. Ratz, *Numerical Toolbox for Verified Computing I*. Springer-Verlag, 1993.

[5] A. V. Panyukov, M. I. Germanenko, and V. V. Gorbik, “Library of classes exact computation,” p. 251, 2009.

[6] GMP. (2017, August) The GNU Multiple Precision Arithmetic Library. Free Software Foundation. [Online]. Available: http://gmplib.org/

[7] V. Golodov and A. Panyukov, “Library of classes exact computation 2.0,” 2013, programs, Data Bases and Topologies of VLIS. Official bulletin of Russian Ageny of Patients and Trademarks. March, 2013.

[8] V. V. Dikoussar, “Some numerical methods for solving linear algebraic equations,” *Soros educational Journal*, no. 9, pp. 111 – 120, 1998.

[9] V. V. Voevodin, *Mathematical Foundations of Parallel Computing*. World Scientific Publishing Company, 1992.

[10] A. N. Tikhonov and V. Y. Arsenin, *Ill-conditioned problems solving methods*. Moscow, Science, 1979.

[11] A. Shrejver, *Theory of Linear and Integer Programming*. John Wiley & Sons, 1986.

[12] I. V. Vlasova and V. I. Vershinin, “Determination of binary mixture components by the firordt method with errors below the specified limit. journal of analytical chemistry,” *Journal of Analytical Chemistry*, vol. 64, no. 6, pp. 553 –558, 2009.