The Libyan Journal of Science (An International Journal): Volume 19A, 2016

An Alternative Approach for Solving Ill-conditioned Systems of Linear Equations

Abdulraheem A. Beraam

Computer Science Dept., Faculty of Science, University of Tripoli
E-mail: a_beraam@tripoliUniv.edu.ly

Abstract

The techniques directed toward errors containment for solving ill-conditioned linear systems ***Ax=b***, is an important topic in both applied mathematics and computer science. Usually floating point numbers are used to represent real numbers, and any computation involving floating point is subject to several types of errors (inherent errors, truncation errors, and round-off errors). These errors are usually accepted. But in critical situations it is considered a catastrophic. The aim of this paper is to provide an alternative approach for solving ill-conditioned linear systems using rational numbers with long integer capacities, and demonstrate this by empirical tests of various known ill-conditioned cases. The results indicate computing with rational numbers does not suffer from round-off errors accumulation.

Keywords: ill-conditioned linear system; round-off error; floating point numbers; rational numbers.

المستخلص

التقنيات الموجهة نحو احتوى الأخطاء التراكمية نتيجة للعمليات الحسابية لإيجاد الحل للمعادلات الخطية على الصورة *Ax=b* وخاصة (المعتلّة والغير مستقرّة) موضوع هام في كل من الرياضيات التطبيقية وعلوم الحاسوب. تستخدم أرقام النقطة العائمة عادة لتمثيل الأعداد الحقيقية، وأن أي عمليات حسابية تنطوي على نقطة عائمة تخضع لعدة أنواع من الأخطاء (الأخطاء الكامنة، وأخطاء البتر، وأخطاء التقريب) وينتج عنه تقريب. هذا التقريب في العادة مقبول ولكن في الحالات الحرجة فهو يعتبر كارثيا.

Accepted for publication: 11/7/2016

الهدف من هذه الورقة هو تقديم نهج بديل لحل المعادلات الخطية تستخدم فيها الكسور الاعتيادية مع قدرات عدد صحيح طويل للتغلب على الأخطاء المصاحبة للنقطة العائمة. وتشير نتائج الاختبارات التي تم إجرائها للعديد من المعادلات الخطية وخاصة المعتلّة المعروفة أن الحوسبة باستخدام الكسور الاعتيادية لا تعاني من تراكم وانتشار للأخطاء.

Introduction

Methods for solving ill-conditioned linear systems ***Ax=b*** have been studied for a long time. When a system is ill-conditioned [1], several types of errors can occur in numerical calculations and round-off errors may accumulate, or exaggerated by the solution procedure and may produce meaningless result. Even though the errors cannot be eliminated, it is possible to have them contained. In the presence of rounding errors, ill-conditioned linear systems are inherently difficult to handle, and one must avoid ill-condition whenever possible. Virtually all previous numerical methods perform their calculations using floating point arithmetic. On the other side, by rewriting the linear system using rational numbers with long integer capability, the computed solution does not suffer from round-off errors accumulation and an exact rather than an approximate solution is obtained.

Floating point Numbers and Rounding Errors Background

There are infinitely many real numbers, but a computer can deal only with finitely many. In computing, floating-point numbers only approximate the much larger set of real numbers, but the exact value requires infinitely many digits and computers cannot handle no matter what precision used (double precisions or extended precision). For example, no way a computer can exactly compute (1/3) and has to be approximated within some tolerance (typically to 16 digits). On the other hand, certain numbers are well-defined in a decimal context e.g., the number (0.1) when convert it into binary number yields 0.0001100110011... This infinite expansion has to be truncated somewhere. Therefore, 0.1 cannot be accurately represented by a finite number of binary digits. Thus, 10×0.1 will not result in the exact value 1.0; instead it will be missed by about 10^{-16}. Furthermore, there are many situations in which we are unable to control the undesirable propagating effects of numerical errors. Consider the following: Set $a=1234.567$, $b=45.67834$ and $c=0.0004$: mathematically $(a+b)+c = a+(b+c)$. This is not the case with floating number computations (Try it!). The losses in the intermediate computations will differ, and you will have a different result for different ways numbers are added. Moreover; consider the following: Set $u = 1$, $w = 3$, $x =$

(1000/3), and y = 333, the expression *u-w.(x-y)* evaluate to zero. However, when (u, w, x, y) represented using floating point, the expression *u-w.(x-y)* will be evaluated to $5.684341886080802\text{x}10^{-14}$. Therefore, except for integers and some fractions, all binary representations of decimal numbers are approximations, and round-off errors are inevitable [2-4].

Current Techniques for Solving Ill-conditioned Linear Systems

There is an extensive research directed toward errors containment in solving system of linear equations $\boldsymbol{Ax = b}$, with and without the use of a computer. As systems are often ill-conditioned due to the finite precision representation of real numbers on a computer, various methods for solving ill-conditioned systems have been proposed. Possible previous remedies to minimize errors containment include [5]:

1. Partial or complete pivoting.
2. Work in double precision or extended precision.
3. Transform the problem into an equivalent system of linear equations by scaling.

In 1981, Rice stated "*if the problem is ill-conditioned, then no amount of effort, trickery, or talent used in the computation can produce accurate answers except by chance*" [6].

Rational Numbers Characteristics

In mathematics, a rational number is any number that can be expressed as the quotient or fraction (*p*/*q)* of two integers, a numerator p and a denominator q, with $q \neq 0$, and can be used to express real values (e.g., 0.1 will be represented by 1/10), and an integer value is equivalent to a rational value with a unit denominator [7]. Mathematicians define rational number (fraction) as an ordered pairs of integer (p, q) and $q \neq 0$, for which the operations addition, subtraction, multiplication, and division are defined as follows:

1. $(a, b) \pm (c, d) = (ad \ \pm bc, bd)$
2. $(a, b) \times (c, d) = (ac, bd)$
3. $(a, b) \div (c, d) = (ad, bc),\ when\ c \neq 0$
4. $(a, b)^n = (a^n, b^n), where\ n\ \in\ \mathbb{Z},$
5. $(a, b)^{-n} = (b^n, a^n), where\ n\ \in\ \mathbb{Z},\ and\ a\ \neq 0$

In computer science, rational numbers can be defined as “class” of ordered pairs of integers (p, q) together with extending the basic operations ('+', '–', '×', '÷', integer powers) performed by methods through operator overloading. Therefore, linear system of the form:

$$\boldsymbol{A}x = b, \quad \boldsymbol{A} \in \mathbf{F}^{nxn}\,,\; x \in \mathbf{F}^{n},\; b \in \mathbf{F}^{n} \tag{1}$$

Can be represented by:

$$\boldsymbol{A}x = b, \quad \boldsymbol{A} \in \mathbf{Q}^{nxn}\,,\; x \in \mathbf{Q}^{n},\; b \in \mathbf{Q}^{n} \tag{2}$$

Where **F** denotes the set of floating point numbers, and **Q** denotes the set of rational numbers.
Example: consider the following linear system [8].

$$\begin{bmatrix} 0.0184 & 0.1507 & 0.1851 \\ 0.1092 & -0.0172 & -0.2726 \\ -0.4781 & -0.8046 & -0.0184 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 0.3542 \\ -0.1807 \\ -1.5025 \end{bmatrix}$$

Can be transformed into rational Format

$$\begin{bmatrix} \frac{23}{1250} & \frac{1507}{10000} & \frac{1851}{10000} \\ \frac{273}{2500} & \frac{-43}{2500} & \frac{-1363}{5000} \\ \frac{-4781}{10000} & \frac{-4023}{5000} & \frac{-23}{1250} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} \frac{1771}{5000} \\ \frac{-1807}{10000} \\ \frac{-601}{400} \end{bmatrix}$$

And the computed solution vector **x** will be in rational form; i.e., all $\boldsymbol{x_i}$ in fraction form (***p*/*q***).

$$x_1 = -\frac{2887628370}{935917529} \approx -3.085344894742323$$

$$x_2 = \frac{6961723097}{1871835058} \approx 3.719196874343402$$

$$x_3 = -\frac{1511955533}{1871835058} \approx -0.8077397239345862$$

Proposed Solution

The basic idea of the rational scheme consists of three steps, and can be described as follows:

1. Convert the linear system $\boldsymbol{A}x = b$ from floating point format Eq. (1) into an equivalent rational numbers representation as shown below Eq. (3).

$$\textstyle\sum_{j=1}^{n} (a_p\,, a_q)_{ij} \cdot \left(x_p\,, x_q\right)_i = (b_p\,, b_q)_i\,, \qquad i = 1, \dots, n \tag{3}$$

Compute $\mathbf{A}^{-1}$ the inverse of a matrix **A** (assume **A** is nonsingular which may be ill-conditioned), where all arithmetic operations ('+', '–', '×', '÷', integer.

2. powers) and relational operations such as (=, ≠, >, <, ≤, ≥) are done in rational arithmetic.
3. Obtain the solution vector $\mathbf{x} = \mathbf{A}^{-1}\mathbf{b}$. The advantages of this method to solve linear equations, there is no need to re-calculate the $\mathbf{A}^{-1}$ each time if **b** is changed [9]. The resulting vector $\boldsymbol{x}$ will be in rational format $[x_i = (x_p/x_q)_i \ , i = 1, 2, \dots, n]$. The accuracy of the solution vector increases even if the solution will be transformed back to real numbers (i.e., best approximation, no round-off errors accumulation).

Verification Steps

In order to assess the performance of the rational model, and to show the accuracy of the approach certain measures were taken in consideration such as:

1. Computing the condition number $\kappa(\mathbf{A}) = ||\mathbf{A}||.||\mathbf{A}^{-1}||$ [10] which is an indication of how sever the ill-condition. A large condition number indicates ill-conditioning.
2. Suppose $\hat{x}$ is a computed solution of $\boldsymbol{Ax} = \boldsymbol{b}$. Computing the residual r=b-$A\hat{x}$, clearly *if r equal zero,* and x - $\hat{x}$ = 0 is an indication that *the solution is an exact* [11].
3. Computing the identity matrix $\mathbf{I} = \mathbf{A}.\mathbf{A}^{-1}$ which is an indication that *computed* $\boldsymbol{A}^{-1}$ *is the exact inverse of A.*
4. Computing $\mathbf{A}' = (\mathbf{A}^{-1})^{-1}$. *If A - A' = 0, shows that no round-off error occurred in the computation of* $(\boldsymbol{A}^{-1})^{-1}$.

Empirical Test Cases and Results

Several empirical tests were conducted to demonstrate the capability and accuracy of this approach using well known techniques where all computation are done using rational numbers. The results indicate that using rational numbers computation give the exact solution rather than an approximate solution even for the extremely ill-conditioned system. To demonstrate the capability of the proposed Algorithm, variety of linear systems of the form Eq. (3) have been tested and exact results obtained. To mention a few all examples presented by Acquah [12], an extremely ill-condition linear system [13], Hilbert matrix [14,15] with different values *for* $n \le 300$*,* and others. Appendix A provides a sample of cases that have been tested using this approach.

Conclusion

An alternative approach for solving linear system $\boldsymbol{Ax} = \boldsymbol{b}$, which may be ill-conditioned using rational numbers is presented with several examples demonstrate the power of the rational arithmetic approach. The conclusions drawn from testing our model with many test cases can be summarized as follows:

1. Exact solution obtained rather than numerical approximation.
2. No need to modify the ill-conditioned matrix in order to make it a better conditioned.
3. No round-off errors occurred during intermediate iteration, and no error propagation. Therefore, no round-off errors accumulation.

References

[1] Vahedipour Z. and Daneshian B. (2013). On the solution of ill-conditioned systems of linear equations, Journal of Expert Systems (JES). World Science Publisher, USA, **2** (1) 119-122.

[2] Farin G. and Hansford D., (2008). Mathematical Principles for Scientific Computing and Visualization, A. K. Peters Ltd., p 7.

[3] Tremblay J. and Bunt R. (1989). Introduction to Computer Science an Algorithmic Approach. 2nd ed., McGraw-Hill, p.768-770.

[4] Peralta-Alva, A. and Santos, M. (2012). Analysis of Numerical Errors, Federal Reserve Bank of St. Louis, Working Paper 2012-062A.

[5] Kreyszig, E. (2011). Advanced Engineering Mathematics, 10^{th} ed., John Wiley & Sons, p.844-850.

[6] Rice, J. (1981). Matrix Computation and Mathematical Software, McGraw-Hill, New York.

[7] Rosen, K. (2007). Discrete Mathematics and its Applications, 6th ed., McGraw-Hill, p. 85.

[8] Miyajima, S. Ogita, A. and Oishi, S. (2005). A method of generating linear systems with an arbitrarily ill-conditioned matrix and an arbitrary solution, International Symposium on Nonlinear Theory and its Applications (NOLTA2005), Bruges, Belgium, p. 741-744.

[9] Zarti, O. (1999). Numerical Method Using Fortran, 2nd ed., ELGA pub., p. 85-86.

[10] Stewart, G. (1973). Introduction to Matrix Computations, Academic Press, New York.

[11] Xue, J., Kozaczek, K., Kurtz, S. and Kurtz, D. (2000). A direct algorithm for solving ill-conditioned linear algebraic systems, International Centre for Diffraction Data, Advances in X-ray Analysis, **42**, 629-633.

[12] Acquah, J. (2009). Regularization of Ill-Conditioned Linear Systems, MSc Thesis, University of Cape Coast.
[13] Rump, S. (2009). Inversion of extremely Ill-conditioned matrices in floating-point, Japan Journal of Industrial and Applied Mathematics *(JJIAM)*, **26** (2-3), 249-277.
[14] Chein-Shan, L. and Chih-Wen C. (2009). Novel methods for solving severely ill-posed linear equations system, Journal of Marine Science and Technology, **17** (3), 216-227.
[15] Soleymani, F. (2013). A new method for solving ill-conditioned linear systems, Opuscula Math, **33** (2), 337–344.

Appendix - Computational Results

The following are samples of test cases that have been tested using rational arithmetic approach.

Test case 1 - Consider the following ill-condition linear system.

$$\begin{bmatrix} 0.5 & 0.5 \\ 0.50000000005 & 0.49999999995 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

Transformed into rational Format, we get

$$\begin{bmatrix} \frac{1}{2} & \frac{1}{2} \\ \frac{10000000001}{20000000000} & \frac{9999999999}{20000000000} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} \frac{1}{1} \\ \frac{1}{1} \end{bmatrix}$$

Results:

Computed κ**(A)** ≈ $2.0\text{x}10^{10}$, residual **r** = 0 and **A - A'** = 0.

The solution vector **x** is an exact $[1, 1]^T$.

Table 1 shows the behavior of ill-conditioned linear equations and how far off the *inv(A)* using floating point from the actual values even thought the solution vector x in both modes are equal.

Table 1. Comparison of the results rational model vs. floating point model

	Rational (R)	*Float* (F)	*Error* ε = |R – F|
Solution vector $\mathbf{x} = [x_1, x_2]^T$			
x_1	1	1.0	0
x_2	1	1.0	0
$inv(A) = A^{-1}$			
a_{11}	-9999999999	-9.9999991715963593e+09	827.4036407470703
a_{12}	10000000000	9.9999991725963593e+09	827.4036407470703
a_{21}	10000000001	9.9999991735963593e+09	827.4036407470703
a_{22}	-10000000000	-9.9999991725963593e+09	827.4036407470703

Test case 2 - Consider the ill-condition linear system.

$$\begin{bmatrix} -5046135670319638 & -3871391041510136 & -5206336348183639 & -6745986988231149 \\ -640032173419322 & 8694411469684959 & -564323984386760 & -2807912511823001 \\ -16935782447203334 & -1875242753803772 & -8188807358110413 & -14820968618548534 \\ -1069537498856711 & -14079150289610606 & 7074216604373039 & 7257960283978710 \end{bmatrix}$$

Results:

Computed **κ(A)** ≈ 71.31 and residual **r** = 0 and **A - A'** = 0.

The computed solution vector **x** is an exact.

$$x1 = \frac{950774706986633557652207357646313604657150960202}{1430715732975754194355660794164899374417975097483114725160500001}$$

$$x2 = -\frac{348452193980609970902537236435835289516685977855}{1430715732975754194355660794164899374417975097483114725160500001}$$

$$x3 = \frac{1895406150668745860323303651370268200443938402887}{1430715732975754194355660794164899374417975097483114725160500001}$$

$$x4 = -\frac{2186126530734990466003691011697825858819452952110}{1430715732975754194355660794164899374417975097483114725160500001}$$

The computed inverse $\mathbf{A}^{-1}$

$$a_{11} = \frac{630378158973611584495870444217721066472876895710}{1430715732975754194355660794164899374417975097483114725160500001}$$

$$a_{12} = \frac{465813558353997976529338004708078722470680732178}{1430715732975754194355660794164899374417975097483114725160500001}$$

$$a_{13} = -\frac{299648478948293336663855965638805206523414953537}{1430715732975754194355660794164899374417975097483114725160500001}$$

$$a_{14} = \frac{154231468607317333290854874359319022237008285851}{1430715732975754194355660794164899374417975097483114725160500001}$$

$$a_{21} = -\frac{253623748449507662196235226534804046831714618117}{1430715732975754194355660794164899374417975097483114725160500001}$$

$$a_{22} = -\frac{84344151693954129924885338033606051194207856713}{1430715732975754194355660794164899374417975097483114725160500001}$$

$$a_{23} = \frac{84772179201932005389050837299309974525922101809}{1430715732975754194355660794164899374417975097483114725160500001}$$

$$a_{24} = -\frac{95256473039080184170467509166735166016685604834}{1430715732975754194355660794164899374417975097483114725160500001}$$

$$a_{31} = \frac{684904080495976114020497106271301778477685883242}{1430715732975754194355660794164899374417975097483114725160500001}$$

$$a_{32} = \frac{1010588511299918762149886875127100045636049005080}{1430715732975754194355660794164899374417975097483114725160500001}$$

$$a_{33} = -\frac{2720708733828258713338661797960187129266728473 77}{1430715732975754194355660794164899374417975097483114725160500001}$$

$$a_{34} = \frac{4719844322556768554867858497678850892568763619 42}{1430715732975754194355660794164899374417975097483114725160500001}$$

$$a_{41} = -\frac{1066656926697323112804505295050570899234291897743}{1430715732975754194355660794164899374417975097483114725160500001}$$

$$a_{42} = -\frac{1079974350885125908759750916757611997944716331884}{1430715732975754194355660794164899374417975097483114725160500001}$$

$$a_{43} = \frac{3854696288264362500245324575847609717206704662 65}{1430715732975754194355660794164899374417975097483114725160500001}$$

$$a_{44} = -\frac{4249648819789776944639672574744039333611151887 48}{1430715732975754194355660794164899374417975097483114725160500001}$$

Test case 3 - Solving the famous ill-conditioned (Hilbert matrix) **Ax=b**, where **A** is ***n*** x ***n*** matrix.

$$a_{ij} = \frac{1}{i+j-1}, \quad b_i = \sum_{j=1}^{n} \frac{1}{i+j-1} \qquad for\ i, j = 1 \ldots n$$

Results:
The computed solution vector **x** is an exact. Solution vector $\boldsymbol{x} = \{x_i = 1, for\ i = 1 \ldots n\}$. Table 2 shows the condition number κ**(A)** and the residual **r** for different value for ***n*** of Hilbert matrix.

Table 2. Condition number and residual.

n	κ**(A) = ||A||.||A^{-1}||**	**r =b -** $\boldsymbol{A\hat{x}}$	**I** = **A**.**A^{-1}**	A - **A'**= 0
10	11235421822540	0	√	√
20	≈ 1.580695807900064x10^{28}	0	√	√
30	≈ 2.6149750373614254x10^{43}	0	√	√
50	≈ 8.459678377949566x10^{73}	0	√	√
100	≈ 2.157356948719007x10^{150}	0	√	√
200	≈ 1.957046806156672x10^{303}	0	√	√
300	≈ 2.059474534375262x10^{456}	0	√	√