Matrix Inverse Computation And Condition Number Effect

Khaled Hallak

May 13, 2017

Introduction

In most of the numerical analysis literature, complexity and stability of numerical algorithms are usually estimated in terms of the problem instance dimension and of a 'condition number'.There exists many methods and algorithms to compute the inverse of a matrix : Gauss elimination, division-free methods, symbolic or numerical computations, . . .
In this paper,We would like to compare three such methods to compute the inverse of a matrix in $GL_n(Q)$:
1. a symbolic Gaussian elimination algorithm ;
2. a floating-point symbolical-numerical algorithm ;
3. a p-adic symbolical-numerical algorithm.

After comparing them we will deduce which method is the most suitable for each matrix . The next step we will discuss the effects of the condition number on the matrix inversion or in solving a linear system of equations .

1 Gaussian elimination

Definition 1.1. *In linear algebra, an n-by-n square matrix A is called invertible (also nonsingular or nondegenerate) if there exists an n-by-n square matrix B such that*

$$AB = BA = I_n$$

A variant of Gaussian elimination called Gauss–Jordan elimination can be used for finding the inverse of a matrix, if it exists. If A is a n by n square

matrix, then one can use row reduction to compute its inverse matrix, if it exists.

First, the n by n identity matrix is augmented to the right of A, forming a n by 2n block matrix $[A \mid I]$. Now through application of elementary row operations, find the reduced echelon form of this n by 2n matrix. The matrix A is invertible if and only if the left block can be reduced to the identity matrix I; in this case the right block of the final matrix is A^{-1}. If the algorithm is unable to reduce the left block to I, then A is not invertible.

Example 1.2.

$$A = \left|\begin{array}{ccc|ccc} 1 & 1 & -1 & 1 & 0 & 0 \\ 1 & -2 & 3 & 0 & 1 & 0 \\ -1 & 2 & -1 & 0 & 0 & 1 \end{array}\right|$$

We can do the following operations to the joint matrix:

Swap two rows Multiply a row by a non-zero scalar Sum two rows and replace one of them with the result Now let us apply these operations to our joint matrix.

1. *Sum rows 2 and 3 and store the result in row 3*

$$A = \left|\begin{array}{ccc|ccc} 1 & 1 & -1 & 1 & 0 & 0 \\ 1 & -2 & 3 & 0 & 1 & 0 \\ 0 & 0 & 2 & 0 & 1 & 1 \end{array}\right|$$

2. *Multiply row 1 by -1, sum with row 2 and store result in 2:*

$$A = \left|\begin{array}{ccc|ccc} 1 & 1 & -1 & 1 & 0 & 0 \\ 0 & -3 & 0 & -1 & -1 & 2 \\ 0 & 0 & 2 & 0 & 1 & 1 \end{array}\right|$$

3. *Multiply row 2 by -1/3 and row 3 by 1/2:*

$$A = \left|\begin{array}{ccc|ccc} 1 & 1 & -1 & 1 & 0 & 0 \\ 0 & 1 & 0 & \frac{1}{3} & \frac{1}{3} & \frac{1}{3} \\ 0 & 0 & 1 & 0 & \frac{1}{2} & \frac{1}{2} \end{array}\right|$$

4. Mutiply row 2 by -1 and sum to row 1, then sum row 3 to row 1:

$$A = \left|\begin{array}{ccc|ccc} 1 & 0 & 0 & \frac{2}{3} & \frac{1}{6} & -\frac{1}{6} \\ 0 & 1 & 0 & \frac{1}{3} & \frac{1}{3} & \frac{1}{3} \\ 0 & 0 & 1 & 0 & \frac{1}{2} & \frac{1}{2} \end{array}\right|$$

5. We end with the inverse:

$$A^{-1} = \begin{bmatrix} \frac{2}{3} & \frac{1}{6} & -\frac{1}{6} \\ \frac{1}{3} & \frac{1}{3} & \frac{1}{3} \\ 0 & \frac{1}{2} & \frac{1}{2} \end{bmatrix}$$

Definition 1.3. *The time complexity of an algorithm is the number of arithmetic operations required to perform this algorithm.*

Time complexity is a way of measuring an algorithm's computational efficiency.

Proposition 1.4. *To solve a system of n equations for n unknowns by performing row operations on the matrix until it is in echelon form, and then solving for each unknown in reverse order, requires* $\frac{n(n-1)}{2}$ *divisions,* $\frac{2n^3+3n^2-5n}{6}$ *multiplications, and* $\frac{2n^3+3n^2-5n}{6}$ *subtractions,then for a total of approximately* $\frac{2n^3}{3}$ *operations. Thus it has arithmetic complexity of* $O(n^3)$.

2 Hilbert Matrix

Definition 2.1. *Hilbert Matrix is a square matrix with entries being the unit fractions such that :*

$$H_{ij} = \frac{1}{i+j-1}$$

Proposition 2.2. *Hilbert matrix is totally positive (which means that the determinant of every submatrix is positive),symmetric and positive definite.*

Theorem 2.3. *We have* $(H^{-1})_{ij} = (-1)^{i+j}(i+j-1)\left(\binom{n+i-1}{n-j}\right)\left(\binom{n+i-1}{n-i}\right)\left(\binom{j+i-2}{i-1}\right)^2$

Where $\binom{n}{k} = \frac{n!}{k!(n-k)!}$

It follows that the entries of the inverse matrix are all integer.For more details check [11]

Proposition 2.4. *The determinant of the n by n Hilbert matrix is*

$$det(H) = \frac{{c_n}^4}{c_{2n}}$$

2.1 the Condition Numbers of Hilbert Matrix

n	1	15	30	70	100
cond inf	1	$6.116565790 * 10^{20}$	$4.227806390 * 10^{43}$	$5.150835229 * 10^{104}$	$3.77648627 * 10^{150}$
$cond10pr$	1	$8.609000647 * 10^{11}$	$9.302273155 * 10^{12}$	$6.528310566 * 10^{12}$	$2.25072215 * 10^{10}$
$cond20pr$	1	$4.403306542 * 10^{17}$	$1.428037881 * 10^{18}$	$3.790225393 * 10^{18}$	$1.81351108 * 10^{20}$
$cond100pr$	1	$4.403306542 * 10^{17}$	$1.428037881 * 10^{18}$	$3.790225393 * 10^{18}$	$1.81351108 * 10^{20}$

Remark 2.5. *The condition number of the n-by-n Hilbert matrix grows as*

$$O\left(\frac{(1+\sqrt{2})^{4n}}{\sqrt{n}}\right)$$

Remark 2.6. *The Hilbert matrices are canonical examples of ill-conditioned matrices(a problem with a high condition number is said to be ill-conditioned.)*

Conclusion 2.7. *As the number of precisions increase , the condition number decrease to the minimum condition number .*

Conclusion 2.8. *Hilbert matrix is notoriously difficult to use in numerical computation*

2.2 Numerical Calculations

We consider H_n as a matrix with real coefficients and compute its inverse using standard Gaussian elimination (with choice of pivot) and $IEEE$ floating-point arithmetics (with 53 bits of precision).

Here is the result we get with n=4

$$H_4^{-1} \approx \begin{pmatrix} 15.9999999999995 & -119.999999999993 & 239.99999999998 & -139.99999999999 \\ -119.999999999993 & 1199.99999999992 & -2699.99999999981 & 1679.99999999987 \\ 239.999999999983 & -2699.99999999987 & 6479.99999999952 & -4199.99999999968 \\ -139.999999999989 & 1679.99999999986 & -4199.99999999967 & 2799.99999999979 \end{pmatrix}$$

We observe that the accuracy of the computed result is acceptable but not so high: the number of correct binary digits is about 45 (on average on each entry of the matrix), meaning then that the number of incorrect digits is about 8. Let us now increase the size of the matrix and observe how the accuracy behaves:

matrix size	5	6	7	8	9	10	11	12	13
correct degits	40	34	28	25	19	14	9	4	0

We see that the losses of accuracy are enormous. For more details ,check [6]

2.3 Time Needed to Calculate Inverse of Hilbert Matrix.

n	10	20	30	40	50	60
time inf pr	0.005093	0.018608	0.056311	0.132840	0.270150	0.497300
time 10 pr	0.000314	0.000340	0.000478	0.000626	0.000282	0.000338
time 50 pr	0.000344	0.000280	0.000308	0.000440	0.000534	0.000714
time 500 pr	0.000590	0.000216	0.000748	0.000568	0.000472	0.000748

Conclusion 2.9. *As the number of pricisions increases , the time needed to calculate the inverse of a hilbert matrix increases.*

3 Condition Number

For many types of problems we can compute a numberm called condition number that indicates the magnification of the changes.

Definition 3.1. *The condition number is defined by:* $Cond(A) = ||A|| \cdot ||A^{-1}||$

Proposition 3.2. *Let A be any square matrix (n by n matrix).The condition number of A is a measure of how ill-conditioned A is and can be found using A and* A^{-1}*, Where (Relative error in the output= Condition number*$\times$ *Relative error in the input).*

Proposition 3.3. *A problem with a low condition number is said to be well-conditioned, while a problem with a high condition number is said to be ill-conditioned.*

Proposition 3.4. *A problem is well-conditioned if small errors in the data produce small errors in the solution, while a problem is ill-conditioned if small errors in the data may produce large errors in the solution*

3.1 Properties of the condition number:

1. For any matrix A, $cond(A) >= 1$
2. For identity matrix, cond (I) = 1
3. For any matrix A and scalar α, $cond(\alpha) = cond(A)$
4. For any diagonal matrix $D = Diag(d_i)$, $cond(D) = \dfrac{max|d_i|}{min|d_i|}$
5. The det(A) has nothing to do with conditioning. is well-conditioned.
6. Let A be an orthogonal matrix, then $Cond_2(A) = 1$.

7. $Cond_2(A) = 1$ if and only if a is a scalar multiple of an orthogonal matrix.

Example 3.5. *For nb (5):*

$$A = \begin{bmatrix} 10^{-30} & 0 \\ 0 & 10^{-30} \end{bmatrix} \textit{iswell} - \textit{conditioned.}$$

Example 3.6. *For nb (6):*

$$A = \begin{bmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{bmatrix}$$

We have $A \cdot A^T = I_2$ *and* $Cond_2(A) = 1$.

Theorem 3.7. $(\frac{||\delta x||}{||x + \delta x||}) \leq Cond(A)\frac{||\Delta A||}{||A||}$, *where Ax=b and* $(A+\Delta A)(x+\delta x)=b$.

Proof. Substracting Ax=b and (A+ΔA)(x+δx)=b , we get δx=-A^{-1} ΔA(x+δ x)
Then we take the norm on both sides.
$||\delta x|| = A^{-1}\ \Delta\mathrm{A}(\mathrm{x}+\delta\ \mathrm{x}) \leq ||A^{-1}|| \cdot ||\Delta A|| \cdot ||x + \delta x|| = ||A^{-1}||\frac{||A||}{||A||}||\Delta A|| \cdot ||x + \delta x|| = Cond(A)\frac{||\Delta A||}{||A||}||x + \delta x||$.
$\Rightarrow\ (\frac{||\delta x||}{||x+\delta x||}) \leq Cond(A)\frac{||\Delta A||}{||A||}$ □

Example 3.8. *Given* $\begin{bmatrix} 1 & 1/2 & 1/3 \\ 1/2 & 1/3 & 1/4 \\ 1/3 & 1/4 & 1/5 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}$ *With* $\Delta A = \begin{bmatrix} 0 & 0 & 0.00003 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$

For the matrix above, the solution is $x = \begin{bmatrix} 3 \\ -24 \\ 30 \end{bmatrix}$

While for the perturbed problem, the solution is $x+\delta x = \begin{bmatrix} 2.991907283444917 \\ -23.967629133779667 \\ 29.973024278149719 \end{bmatrix}$

$(\frac{||\delta x||}{||x+\delta x||}) = 0.1114657219952 \leq 0.011163453834773 = Cond(A)\frac{||\Delta A||}{||A||}$

Example 3.9. *Given*

$$\begin{bmatrix} 1 & 1/2 & 1/3 & 1/4 & 1/5 & 1/6 & 1/7 & 1/8 \\ 1/2 & 1/3 & 1/4 & 1/5 & 1/6 & 1/7 & 1/8 & 1/9 \\ 1/3 & 1/4 & 1/5 & 1/6 & 1/7 & 1/8 & 1/9 & 1/10 \\ 1/4 & 1/5 & 1/6 & 1/7 & 1/8 & 1/9 & 1/10 & 1/11 \\ 1/5 & 1/6 & 1/7 & 1/8 & 1/9 & 1/10 & 1/11 & 1/12 \\ 1/6 & 1/7 & 1/8 & 1/9 & 1/10 & 1/11 & 1/12 & 1/13 \\ 1/7 & 1/8 & 1/9 & 1/10 & 1/11 & 1/12 & 1/13 & 1/14 \\ 1/8 & 1/9 & 1/10 & 1/11 & 1/12 & 1/13 & 1/14 & 1/15 \end{bmatrix} \begin{bmatrix} x1 \\ x2 \\ x3 \\ x4 \\ x5 \\ x6 \\ x7 \\ x8 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} \textit{with}$$

$$\Delta A = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.00003 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

For the matrix above , the solution is $x = \begin{bmatrix} -8 \\ 504 \\ -7560 \\ 46200 \\ -138600 \\ 216216 \\ -168168 \\ 51480 \end{bmatrix}$

While for the perturbed problem, the solution is $x+\delta x=$ $\begin{bmatrix} -8.00245004559999984 \\ 504.000932532499974 \\ -7560.00931452359964 \\ 46200.0020021301025 \\ -138600.007564412314 \\ 216216.000262624497 \\ -168168.008322145790 \\ 51480.0035441236032 \end{bmatrix}$

$$\left(\frac{||\delta x||}{||x+\delta x||}\right) = 4.888182976\times10^{-8} \leq 2.721795621\times10^{-7} = Cond(A)\frac{||\Delta A||}{||A||}.$$

Proposition 3.10. *We call a square matrix A ill-conditioned if it is invertible but can become non-invertible (singular) if some of its entries are changed ever so slightly.*

Remark 3.11. *The bigger the condition number is the more ill-conditioned A is.*

.

Remark 3.12. *Well-conditioned matrices have condition numbers close to 1.*

Definition 3.13. *If a matrix has a bad condition number, the solutions are unstable with respect to small changes in data. It implies that small errors in the input can cause large errors in the output.*

Example 3.14. $\begin{bmatrix} 400 & -201 \\ -800 & 401 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 200 \\ -200 \end{bmatrix}$

Then the solution is $x_1 = -100$ *and* $x_2 = -200$ *with condition number =2503.004601*

Now, let us make a slight change in one of the elements of the coefficient matrix.
Change A_{11} *from 400 to 401 and see how this small change affects the solution of the following.*

$$\begin{bmatrix} 401 & -201 \\ -800 & 401 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 200 \\ -200 \end{bmatrix}$$

This time the solution is $x_1 = 40000$ *and* $x_2 = 79800$.

3.2 Condition Number in Linear systems

Definition 3.15. *The solutions of some linear systems (that can be represented by systems of linear equations) are more sensitive to round-off error than others. For some linear systems a small change in one of the values of the coefficient matrix or the right-hand side vector causes a large change in the solution vector.*

Consider the following system of three linear equations in three unknowns.

Example 3.16. *:*

$$\begin{bmatrix} 1.0 & 0.50000 & 0.33333 \\ 0.50000 & 0.33333 & 0.25000 \\ 0.33333 & 0.25000 & 0.20000 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix}$$

Then the solution is $x_1 = 27$ *and* $x_2 = -192$ *and* $x_3 = 210$ *with condition number =524.0567776*

Now, let us make a slight change in one of the elements of the coefficient matrix.
Change A_{22} *from 0.3333 to 0.4333 and see how this small change affects the solution of the following.*

$$\begin{bmatrix} 1.0 & 0.50000 & 0.33333 \\ 0.50000 & 0.43333 & 0.25000 \\ 0.33333 & 0.25000 & 0.20000 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix}$$

This time the solution is $x_1 = -7.2173801247$ *and* $x_2 = -9.505268425$ *and* $x_3 = 38.91043211677$.

With a modest change in one of the coefficients one would expect only a small change in the solution. However, in this case the change in solution is

quite significant. It is obvious that in this case the solution is very sensitive to the values of the coefficient matrix A.
When the solution is highly sensitive to the values of the coefficient matrix A or the righthand side constant vector b, the equations are called to be ill-conditioned. Ill-conditioned systems pose particular problems where the coefficients or constants are estimated from experimental results or from a mathematical model. Therefore, we cannot rely on the solutions coming out of an ill-conditioned system. The problem is then how do we know when a system of linear equations is ill-conditioned.

Definition 3.17. *The condition number is defined more precisely to be the maximum ratio of the relative error in x divided by the relative error in b.*

Proof. Let e be the error in b. Assuming that A is a nonsingular matrix, the error in the solution $A^{-1}b$ is $A^{-1}e$. The ratio of the relative error in the solution to the relative error in b is

$$\frac{\frac{||A^{-1}e||}{||A^{-1}b||}}{\frac{||e||}{||b||}}$$

This is easily transformed to $\left(\frac{||A^{-1}e||}{||e||}\right) \cdot \left(\frac{||b||}{||A^{-1}b||}\right)$
The maximum value (for nonzero b and e) is then seen to be the product of the two operator norms as follows:

$$max\left(\frac{||A^{-1}e||}{||e||}\right) \cdot \left(\frac{||b||}{||A^{-1}b||}\right) = \left(max\frac{||A^{-1}e||}{||e||}\right) \cdot \left(max\frac{||b||}{||A^{-1}b||}\right)$$

$$= \left(max\frac{||A^{-1}e||}{||e||}\right) \cdot \left(max\frac{||Ax||}{||x||}\right) = ||A^{-1}|| \cdot ||A||. \qquad \square$$

You can also check [7] and [18]

3.3 Condition and Distance

Definition 3.18. *The condition number $\kappa_{rs}(A)$ can be expressed as the relativized inverse of the distance from the square matrix A to the set σ of singular matrices.*

Proposition 3.19. *Large $\kappa_{rs}(A)$ means that A is close to a singular matrix. In order to make this precise, we introduce the distance of $A \in R^{n*n}$ to the set Σ of singular matrices.*

Theorem 3.20. $d_{rs}(A, \Sigma) := min||A - B||_{rs} s.t B \in \Sigma$

Theorem 3.21. $d_{rs}(A, \sum) = \dfrac{1}{||A^{-1}||_{sr}}$

Theorem 3.22.

$$\kappa_{rs}(A) = \frac{||A||_{rs}}{d_{rs}(A, \Sigma)}$$

Proof. Let A be nonsingular and let A + E be singular.Then there exists an x $\in R^n$,such that (A + E)x = 0. This means that x $=-A^{-1}Ex$

Hence $||x||_r \leq ||A^{-1}E||_{rr}.||x||_r \leq ||A^{-1}||_{sr}.||E||_{rs}.||x||_r$
$\Rightarrow ||E||_{rs} \geq ||A^{-1}||_{sr}^{-1}$. Therefore $d_{rs}(A, \Sigma) \geq ||A^{-1}||_{sr}^{-1}$.

To prove the other inequality, it suffices to find a singular matrix M with $d_{rs}(A, M) \leq ||A^{-1}||_{sr}^{-1}$. Let y$\in R^n$ be such that $||A^{-1}||_{sr} = ||A^{-1}y||_r$ and $||y||_s = 1$ and let $x := A^{-1}y$ we have $||x||_r = ||A^{-1}||_{sr}$. There exists B$\in R^{n*n}$ such that $||B||_{rs} = 1$ and B$\dfrac{x}{||x||_r} = -y$
Then we have $d_{rs}(A, M) = ||E||_{rs} = ||x||_r^{-1}||B||_{rs} = ||A^{-1}||_{sr}^{-1}||B||_{rs} = ||A^{-1}||_{sr}^{-1}$.

which finishes the proof.

Now we can say that:$\forall$ non zero $A \in R^{n*n}$ we have :

$\kappa_{rs}(A) = \dfrac{||A||_{rs}}{d_{rs}(A,\Sigma)}$. where $\kappa_{rs}(A) = ||A||_{rs}.||A^{-1}||_{rs}$

Thus the condition number κ_{rs} can be seen as the inverse of a normalized distance of A to the set of ill-posed inputs $\sum$.

□

4 Time to do the inverse to number of random matrix

let m be the number of matrices and let n be the number of rows of the random square matrix and pr the number of precision .

we will need an algorithm which is with(LinearAlgebra);
timingmatrix := proc (size, nb)
local t, i, M, st, M2;
t := 0;
for i to nb do
M := RandomMatrix(size, size);
st := time();
M2 := MatrixInverse(M);
t := t+time()-st
end do;
return evalf(t/nb)
end proc;

n=10

pr \ m	5000
$infinite$	0.005403
10	0.0005478
30	0.0005247
100	0.0005324

for n=20:

pr \ m	5000
$infinite$	0.0367369
10	0.00024315
30	0.00025634
100	0.00029310

For n=30:

$pr \setminus m$	5000
$infinite$	0.1349496
$10pr$	0.000449
30	0.000412
100	0.000446

5 p-adic

Throughout the 20th century, many p-adic objects have been defined and studied as well: p-adic modular/automorphic forms, p-adic differential equations, p-adic cohomologies, p-adic Galois representations, etc. It turns out that mathematicians, more and more often, feel the need to make "numerical" experimentations on these objects for which having a nice implementation of p-adic numbers is of course a prerequisite.I refer the interested reader to [6] for a more complete exposition of the theory of p-adic numbers

p-adic numbers are very ambivalent objects which can be thought of under many different angles: computational, algebraic, analytic. It turns out that each point of view leads to its own definition of p-adic numbers: computer scientists often prefer viewing a p-adic number as a sequence of digits while algebraists prefer speaking of projective limits and analysts are more comfortable with Banach spaces and completions. Of course all these approaches have their own interest and understanding the intersections between them is often the key behind the most important advances.

We have seen that p-adic numbers are a wonderful mathematical object which might be quite useful for arithmeticians. However it is still not clear that it is worth implementing them in mathematical software. The aim of this subsection is to convince the reader that it is definitely worth it. I refer [16] for more details about $p-adic$ numbers.

Proposition 5.1. *The field of p-adic numbers noted $\mathbb{Q}_p$ is by definition the field of fractions of $\mathbb{Z}_p$.*

Remark 5.2. *Here are three strong arguments supporting this thesis:*

1. *p-adic numbers provide sometimes better numerical stability;*
2. *p-adic numbers provide a solution for "allowing for division by p in $\mathbb{F}_p$.*
3. *p-adic numbers really appear in nature.*

- A method for computing the inverse of an (n×n) integer matrix A using p-adic approximation is given. The method is similar to Dixon's algorithm, but ours has a quadratic convergence rate. The complexity of this algorithm (without using FFT or fast matrix multiplication) is $O(n^4(logn)^2)$, the same as that of Dixon's algorithm. However, experiments show that our method is faster. This is because our methods decrease the number of matrix multiplications but increase the digits

of the components of the matrix, which suits modern CPUs with fast integer multiplication instructions.

5.1 Exact solution of linear equations using P-adic expansions

- A method is described for computing the exact rational solution to a regular system $Ax = b$ of linear equations with integer coefficients. The method involves:

  1. Computing the inverse $(modp)$ of A for some prime p ;
  2. Using successive refinements to compute an integer vector $\bar{x}$ such that $A\bar{x} \equiv b(modp^m)$ for a suitably large integer m;
  3. Deducing the rational solution x from the p-adic approximation $\bar{x}$. For matrices A and b with entries of bounded size and dimensions $n \times n$ and $n \times 1$, this method can be implemented in time $\mathcal{O}(n^3(\log n)^2)$ which is better than methods previously used.

Example 5.3. *In order to avoid having to introduce too advanced concepts here, I have chosen to focus on a simpler example which was pointed out in Vaccon's PhD thesis [17,1.3.4] the computation of the inverse of the Hilbert matrix.*

Now we will compare the numerical calculation done in (2.2) for the hilbert matrix H_4 using the p-adic , So let us examine now the computation goes when H_n is viewed as a matrix over Q_2.Making experimentations using again Gaussian elimination and the straightforward analogue of floating-point arithmetic with 53 bits of precision, we observe the following behavior:

matrix size	*5*	*6*	*7*	*8*	*9*	*10*	*11*	*12*	*13*
correct degits	*52*	*52*	*51*	*51*	*51*	*51*	*51*	*49*	*48*

The computation of H_n seems quite accurate over Q_2 whereas it was on the contrary highly inaccurate over R. As a consequence, if we want to use numerical methods to compute the exact inverse of H_n over Q, it is much more interesting to go through the 2-adic numbers. Of course this approach does not make any sense if we want a real approximation of the entries of H_n ; in particular, if we are only interesting in the size of the entries of H_n (but not in their exact values) passing through

the p-adics is absurd since two integers of different sizes might be very close in the p-adic world.
For more details check [6]

- The phenomena occurring here are actually easy to analyze. The accuracy of the inversion of a real matrix is governed by the condition number which is defined by:
$Cond_R(H_n) = ||H_n||_R \cdot ||H_n^{-1}||_R$,where $||\cdot||_R$ is some norm on $M_n(R)$.According to (Theorem 2.3) ,the entries of Hn are large: as an example, the bottom right entry of Hn is equal to $(2n-1) \cdot \binom{2n-2}{n-1}$ and thus grows exponentially fast with respect to n. As a consequence the condition number $cond_R(H_n)$ is quite large as well; this is the source of the loss of accuracy observed for the computation of H_n^{-1} over R.
Over Q_p, one can argue similarly and consider the p-adic condition number:

$$cond_{Q_p}(H_n) = ||H_n||_{Q_p} \cdot ||H_n^{-1}||_{Q_p}$$

. where $|| \cdot ||_{Q_p}$is the infinite norm over $M_n(Q_p)$ (other norms do not make sense over Qp because of the ultrametricity). Since H_n^{-1} has integral coefficients, all its entries have their p-adic norm bounded by 1. Thus $||H_n^{-1}||_{Q_p} \leq 1$. As for the Hilbert matrix H_n itself, its norm is equal to p^v where v is the highest power of p appearing in a denominator of an entry of H_n ,*i.e.* v is the unique integer such that $p^v \leq 2n-1 < p^{v+1}$. Therefore $||H_n||_{Q_p} \leq 2n$ and $CondQ_p(H_n) = \bigcirc(n)$.
The growth of the p-adic condition number in then rather slow, explaining why the computation of H_n is accurate over Q_p.

5.2 Time to calculate inverse

Here we will compare time needed to calculate the inverse of a matrix with p-adic algorithm by changing the prime number or by changing the accuracy m or by changing the size of a Matrix .

Journal of Computational and Applied Mathematics :

[4] if you need more details about this .

Table 1:

Base prime p moves :

p	2	11	1009	10007	100003
$Time$	0.15405	0.25406	0.74027	1.09738	1.51910

Matrix size n = 30 and accuracy m = 100 are fixed.

Table 2:

Accuracy m moves:

m	10	50	100	200	1000
$Time$	0.11987	0.39809	0.67099	2.02269	18.4427

p = 541 (100th prime) and matrix size n = 30 are fixed.

Table 3:

Matrix size n moves:

n	10	20	30	40	50
$Time$	0.03659	0.21385	0.66252	1.54727	2.90654

p = 541 (100th prime) and accuracy m = 100 are fixed.

5.3 Comparison between p-adic and Dixon algorithms

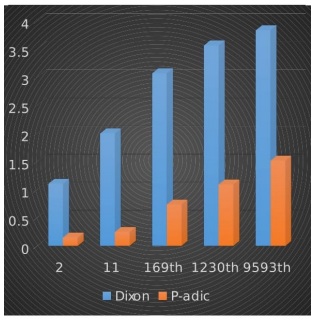

Figure 1: Time with changing P

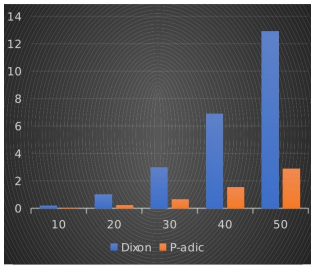

Figure 2: Time with changing m

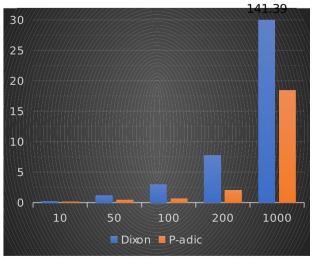

Figure 3: Time with changing n

Conclusion 5.4. *Experiments show that this method is faster than Dixon's method, having a linear conversion ratio.*

References

[1] J.D. Dixon, Exact solution of linear equations using p-adic expansions, Numer. Math. 40 (1982) 137–141.

[2] J.V.Z. Gathen, J. Gerhard, Modern Computer Algebra, Cambridge University Press, Cambridge, UK, 1999.

[3] X. Wang, V.Y. Pan, Acceleration of Euclidean algorithm and rational number reconstruction, SIAM J. Comput. 32 (2) (2003) 548–556.

[4] H. Haramoto, M. Matsumoto http://ac.els-cdn.com/S0377042708003907/1-s2.0-S0377042708003907-main.pdf?_tid=16601170-2b26-11e7-9aef-00000aacb362&acdnat=1493283004_eb815df0e07542438acaaf9edcbcf985

[4] [Peter Bürgisser,Felipe Cucker] The geometry of Numerical algorithm.

[5] Wayne Eberly, Mark Giesbrecht, Pascal Giorgi, Arne Storjohann, Gilles Villard. Solving Sparse Integer Linear Systems. 2006. ¡hal-00021456¿ https://hal.archives-ouvertes.fr/hal-00021456/document

[6] Xavier Caruso,Computations with p-adic numbers

[7] GREGORIO MALAJOVICH, Condition Number bounds for problems with integer coefficients .(February 12, 1999).

[8] JOACHIM VON ZUR GATHEN, Modern Computer Algebra,(Third edition) .

[9] E. Kaltofen. Analysis of Coppersmith's block Wiedemann algorithm for the parallel solution of sparse linear systems. Mathematics of Computation, 64(210):777–806, 1995

[10] B. D. Saunders and Z. Wan. Smith normal form of dense integer matrices, fast algorithms into practice

[11] https://en.wikipedia.org/wiki/Hilbert_matrix

[12] https://en.wikipedia.org/wiki/P-adic_number

[13] J.D. Dixon, Exact solution of linear equations using p-adic expansions Numer. Math., 40 (1982), pp. 137–141

[14] John D. Dixon Department of Mathematics and Statistics, Carleton University, Ottawa K1S 5B6, Canada .

[15] Wang, 1978 Wang,P.S., An Improved Multivariable Polynomial Factorising Algorithm. Math. Comp. 32(1978) pp. 1215-1231. Zbl. 388.10035.

[16] Yvette Amice. Les nombres p-adiques. PUF (1975)

[17,1.3.4] Tristan Vaccon. Precision p-adique. PhD Thesis (2015). Available at https://tel.archives-ouvertes.fr/tel-01205269

[18] https://en.wikipedia.org/wiki/Condition_number