Επιστημονικός Υπολογισμός Ι
ΗΥ 343: ΔΙΑΛΕΞΗ 11

Ε. Γαλλόπουλος
Τμήμα Η/Υ & Πληροφορικής
Πανεπιστήμιο Πατρών

Αντιστάθμιση (compensation) - βασική ιδέα

Πόρισμα Αν οι αριθμοί $x \geq y$ είναι στο σύστημα α.κ.υ. IEEE και χρησιμοποιούμε στρογγύλευση προς τον πλησιέστερο και αν

$$s = \mathrm{fl}(x+y), w = \mathrm{fl}(s-x), c = \mathrm{fl}(y-w)$$

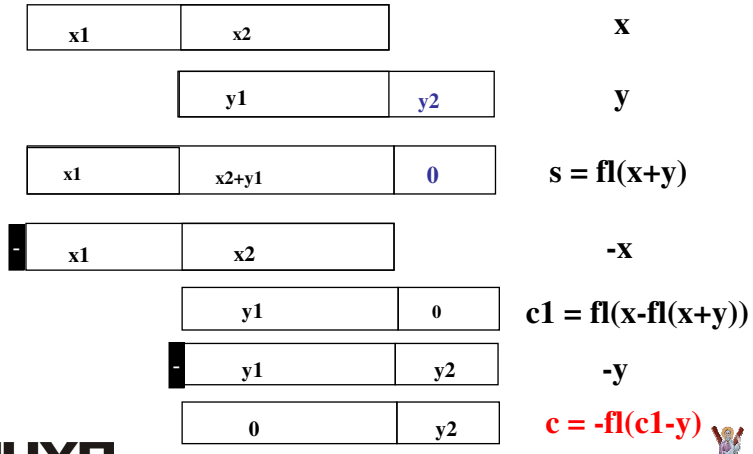
τότε

$$s + c = x + y$$

ΔΗΛΑΔΗ για κάθε ζευγάρι προσθετέων (x, y) μπορούμε να υπολογίσουμε ακριβώς την προσέγγιση του αθροίσματος $s := \mathrm{fl}(x+y)$ και έναν (γενικά πολύ μικρό) α.κ.υ. c που σε αριθμητική άπειρης ακρίβειας μπορεί να διορθώσει το s και να το επαναφέρει στην τιμή του ακριβούς αθροίσματος $x+y$.

```
function [s,e] = kahan0(x,y);
    if (abs(x)< abs(y))
      temp =a; a=b; b=temp;
    end
    s = x+y;
    e = y-(s-x);
```

Η διαδικασία σχηματικά (λίγο παραλλαγμένη η σειρά)



Πως χρησιμοποιείται το c (σαν carry);

- Το fl(s) είναι το καλύτερο δυνατό (ακριβή στρογγύλευση)
- άρα η διόρθωση fl(fl(c)+fl(x+y)) δεν οδηγεί σε ακριβέστερο άθροισμα
- Αλλά αν έχουμε να προσθέσουμε και άλλον αριθμό, z, τότε μπορούμε να «διορθώσουμε» με το c

```
function [s] = kahan1(x)
s = 0; e = 0;
for j = 1:length(x)
  temp = s
  y = x(j) + e
  s = s + y
  e = (temp-s)+y
end
```

A Floating-Point Technique for Extending the Available Precision

T. J. DEKKER*

Received July 26, 1971

Abstract. A technique is described for expressing multilength floating-point arithmetic in terms of singlelength floating point arithmetic, i.e. the arithmetic for an available (say: single or double precision) floating-point number system. The basic algorithms are exact addition and multiplication of two singlelength floating-point numbers, delivering the result as a doublelength floating-point number. A straightforward application of the technique yields a set of algorithms for doublelength arithmetic which are given as ALGOL 60 procedures.

Let x and y be singlelength floating-point numbers and let

$$z = fl(x+y);$$

i.e. z is the result of a singlelength floating-point addition of x and y. Let zz be the correction term exactly satisfying

$$z+zz = x+y.$$

It will be shown that, under various conditions, zz can be obtained by the formula

$$zz = fl((x-z)+y).$$

We shall derive some formulas for calculating zz which use only singlelength floating-point addition and subtraction.

First we consider the formula

(4.3) $$w = fl(z-x), \quad zz = fl(y-w)$$

and prove some theorems stating sufficient conditions for the validity of this formula. For practical computation, formulas (4.1) and (4.3) can be written as the following sequence of ALGOL 60 statements:

(4.4) "$z:=x+y; \quad zz:=y-(z-x)$",

in which w remains anonymous.

Let $x, y \in R$ be representable such that their exponents satisfy

(4.5) $$ex \geqq ey.$$

In particular, this holds if $x, y \in R$ satisfy

(4.7) **Theorem.** If R has the form (2.2), where $\beta = 2$ or 3 and M is a multiple of β, and if, moreover, floating-point addition is optimal and subtraction faithful, then, for all x and y satisfying (4.5) and for z obtained according to (4.1), formula (4.3) yields the correction term zz defined by (4.2).

Αντιστάθμιση - βασική ιδέα

Πόρισμα Αν οι αριθμοί $x \geq y$ είναι στο σύστημα α.κ.υ. IEEE και χρησιμοποιούμε στρογγύλευση προς τον πλησιέστερο και αν

$$s = \mathrm{fl}(x + y), w = \mathrm{fl}(s - x), c = \mathrm{fl}(y - w)$$

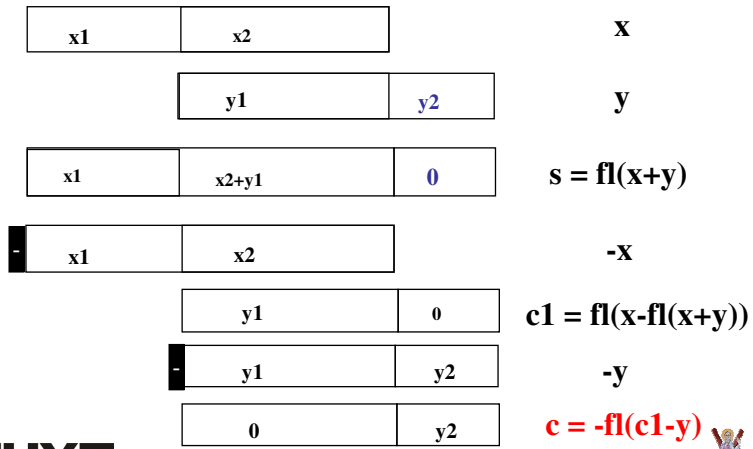
τότε

$$s + c = x + y$$

ΔΗΛΑΔΗ για κάθε ζευγάρι προσθετέων (x, y) μπορούμε να υπολογίσουμε ακριβώς την προσέγγιση του αθροίσματος $s := \mathrm{fl}(x + y)$ και έναν (γενικά πολύ μικρό) α.κ.υ. c που σε αριθμητική άπειρης ακρίβειας μπορεί να διορθώσει το s και να το επαναφέρει στην τιμή του ακριβούς αθροίσματος $x + y$.

```
function [s,e] = kahan0(x,y);
    if (abs(x)< abs(y))
      temp =a; a=b; b=temp;
    end
    s = x+y;
    e = y-(s-x);
```


Η διαδικασία σχηματικά (λίγο παραλλαγμένη η σειρά)



Πως χρησιμοποιείται το c (σαν carry);

- Το **fl(s)** είναι το καλύτερο δυνατό (ακριβή στρογγύλευση)
- άρα η διόρθωση **fl(fl(c)+fl(x+y))** δεν οδηγεί σε ακριβέστερο άθροισμα
- Αλλά αν έχουμε να προσθέσουμε και άλλον αριθμό, **z**, τότε μπορούμε να «διορθώσουμε» με το **c**

```
function [s] = kahan1(x)
s = 0; e = 0;
for j = 1:length(x)
  temp = s
  y = x(j) + e
  s = s + y
  e = (temp-s)+y
end
```


Ιδιότητα αντισταθμισμενης άθροισης

- Γενικά πιο ακριβής από κανονική άθροιση.
- Αποδεικνύεται ότι το άθροισμα **n** στοιχείων

$$fl(s) = \Sigma(1+\mu_j)\, x_j$$

όπου $|\mu_j| \le 2u + O(nu^2)$

- Δηλαδή

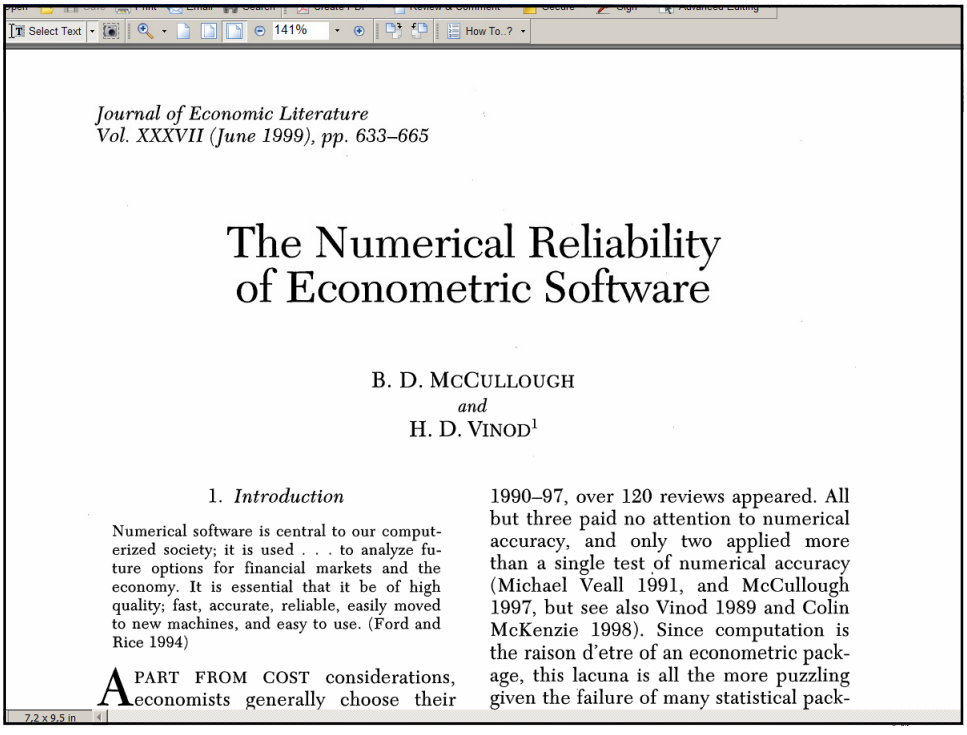
*σε πρώτη προσέγγιση, το πίσω σφάλμα είναι ανεξάρτητο από το **n**.*

Παράδειγμα

- Έστω x = 1/n * ones(n,1)
  - Τότε θεωρητικά, sum(x) = 1
- Εκτελούμε σε MATLAB

n	sum(x)	kahan1(x)	abs(1-sum(x))	abs(1-kahan1(x))
10	9.999999999999999e-01	1	1.1102e-016	0
100	1.000000000000001e+00	1	6.6613e-016	0
10^4	9.999999999980838e-01	1	1.9162e-012	0

ΤΜΗΥΠ
Πανεπιστήμιο Πατρών



Journal of Economic Literature
Vol. XXXVII (June 1999), pp. 633–665

The Numerical Reliability of Econometric Software

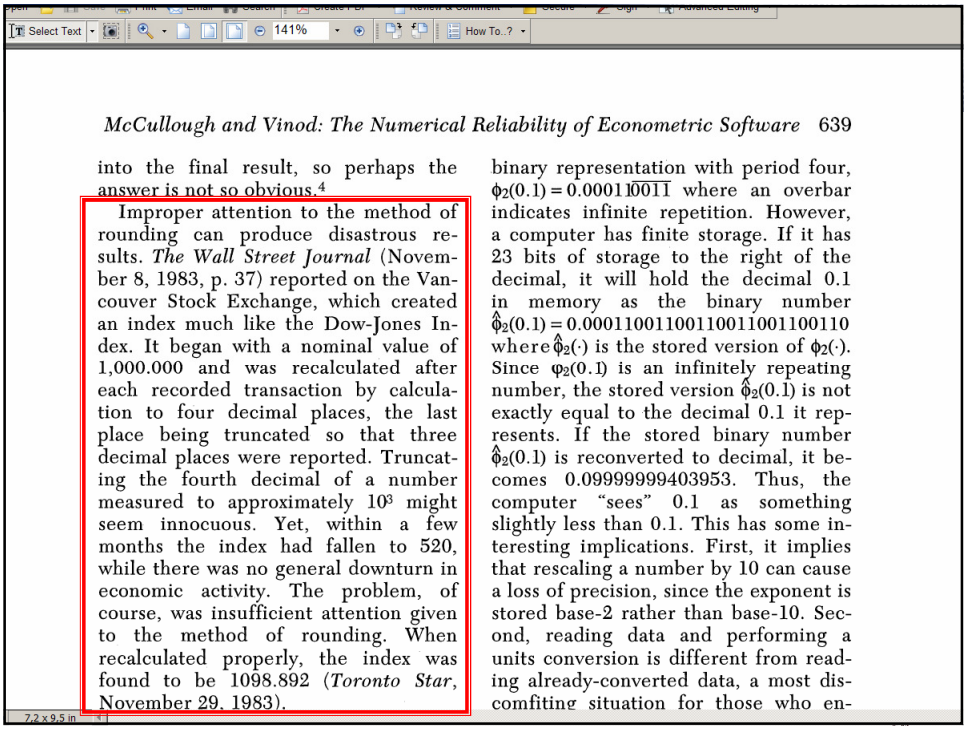
B. D. McCullough
and
H. D. Vinod[1]

1. *Introduction*

Numerical software is central to our computerized society; it is used . . . to analyze future options for financial markets and the economy. It is essential that it be of high quality; fast, accurate, reliable, easily moved to new machines, and easy to use. (Ford and Rice 1994)

APART FROM COST considerations, economists generally choose their

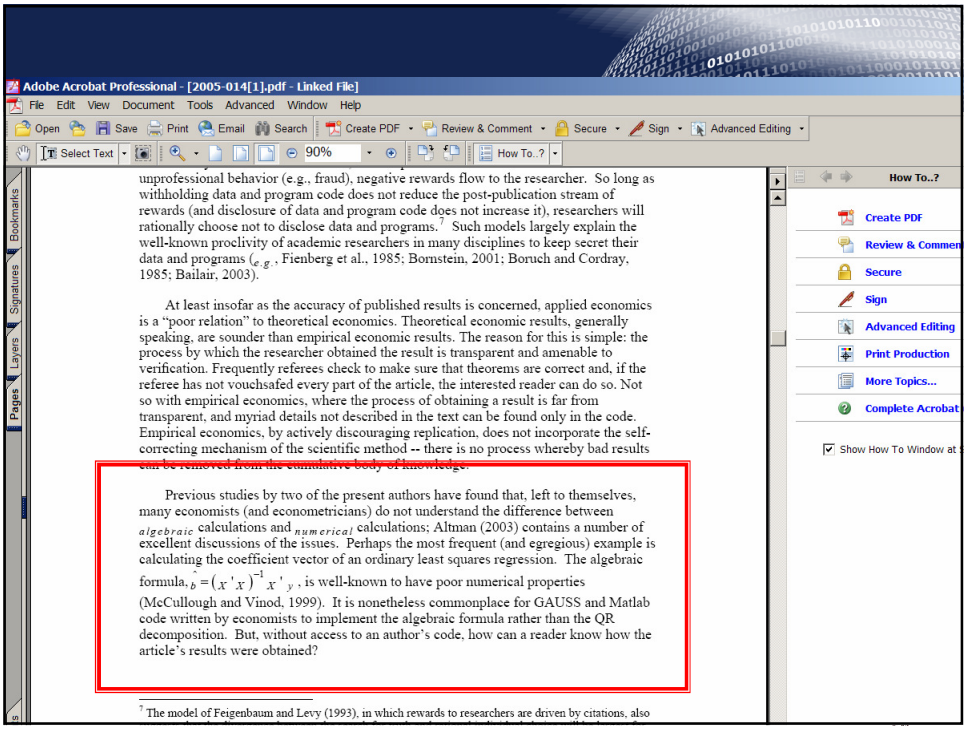
1990–97, over 120 reviews appeared. All but three paid no attention to numerical accuracy, and only two applied more than a single test of numerical accuracy (Michael Veall 1991, and McCullough 1997, but see also Vinod 1989 and Colin McKenzie 1998). Since computation is the raison d'etre of an econometric package, this lacuna is all the more puzzling given the failure of many statistical pack-



into the final result, so perhaps the answer is not so obvious.[4]

Improper attention to the method of rounding can produce disastrous results. *The Wall Street Journal* (November 8, 1983, p. 37) reported on the Vancouver Stock Exchange, which created an index much like the Dow-Jones Index. It began with a nominal value of 1,000.000 and was recalculated after each recorded transaction by calculation to four decimal places, the last place being truncated so that three decimal places were reported. Truncating the fourth decimal of a number measured to approximately 10^3 might seem innocuous. Yet, within a few months the index had fallen to 520, while there was no general downturn in economic activity. The problem, of course, was insufficient attention given to the method of rounding. When recalculated properly, the index was found to be 1098.892 (*Toronto Star*, November 29, 1983).

binary representation with period four, $\phi_2(0.1) = 0.0001\overline{1001}1$ where an overbar indicates infinite repetition. However, a computer has finite storage. If it has 23 bits of storage to the right of the decimal, it will hold the decimal 0.1 in memory as the binary number $\hat{\phi}_2(0.1) = 0.00011001100110011001100110$ where $\hat{\phi}_2(\cdot)$ is the stored version of $\phi_2(\cdot)$. Since $\phi_2(0.1)$ is an infinitely repeating number, the stored version $\hat{\phi}_2(0.1)$ is not exactly equal to the decimal 0.1 it represents. If the stored binary number $\hat{\phi}_2(0.1)$ is reconverted to decimal, it becomes 0.09999999403953. Thus, the computer "sees" 0.1 as something slightly less than 0.1. This has some interesting implications. First, it implies that rescaling a number by 10 can cause a loss of precision, since the exponent is stored base-2 rather than base-10. Second, reading data and performing a units conversion is different from reading already-converted data, a most discomfiting situation for those who en-



Adobe Acrobat Professional - [2005-014[1].pdf - Linked File]

unprofessional behavior (e.g., fraud), negative rewards flow to the researcher. So long as withholding data and program code does not reduce the post-publication stream of rewards (and disclosure of data and program code does not increase it), researchers will rationally choose not to disclose data and programs.[7] Such models largely explain the well-known proclivity of academic researchers in many disciplines to keep secret their data and programs (*e.g.*, Fienberg et al., 1985; Bornstein, 2001; Boruch and Cordray, 1985; Bailair, 2003).

At least insofar as the accuracy of published results is concerned, applied economics is a "poor relation" to theoretical economics. Theoretical economic results, generally speaking, are sounder than empirical economic results. The reason for this is simple: the process by which the researcher obtained the result is transparent and amenable to verification. Frequently referees check to make sure that theorems are correct and, if the referee has not vouchsafed every part of the article, the interested reader can do so. Not so with empirical economics, where the process of obtaining a result is far from transparent, and myriad details not described in the text can be found only in the code. Empirical economics, by actively discouraging replication, does not incorporate the self-correcting mechanism of the scientific method -- there is no process whereby bad results can be removed from the cumulative body of knowledge.

Previous studies by two of the present authors have found that, left to themselves, many economists (and econometricians) do not understand the difference between *algebraic* calculations and *numerical* calculations; Altman (2003) contains a number of excellent discussions of the issues. Perhaps the most frequent (and egregious) example is calculating the coefficient vector of an ordinary least squares regression. The algebraic formula, $b = (X'X)^{-1}X'y$, is well-known to have poor numerical properties (McCullough and Vinod, 1999). It is nonetheless commonplace for GAUSS and Matlab code written by economists to implement the algebraic formula rather than the QR decomposition. But, without access to an author's code, how can a reader know how the article's results were obtained?

[7] The model of Feigenbaum and Levy (1993), in which rewards to researchers are driven by citations, also

High-Precision Floating-Point Arithmetic
in Scientific Computation
David H. Bailey
28 January 2005

Abstract

At the present time, IEEE 64-bit floating-point arithmetic is sufficiently accurate for most scientific applications. However, for a rapidly growing body of important scientific computing applications, a higher level of numeric precision is required: some of these applications require roughly twice this level; others require four times; while still others require hundreds or more digits to obtain numerically meaningful results. Such calculations have been facilitated by new high-precision software packages that include high-level language translation modules to minimize the conversion effort. These activities have yielded a number of interesting new scientific results in fields as diverse as quantum theory, climate modeling and experimental mathematics, a few of which are described in this article. Such developments suggest that in the future, the numeric precision used for a scientific computation may be as important to the program design as are the algorithms and data structures.

3. Climate Modeling

It is well-known that weather or climate simulations are fundamentally chaotic—if microscopic changes are made to the present state, within a certain period of simulated time the future state is completely different. Indeed, ensembles of these calculations are required to obtain statistical confidence in global climate trends produced from such calculations. As a result, computational scientists involved in climate modeling applications have long resigned themselves that their codes quickly diverge from any "baseline" calculation, even if they only change the number of processors used to run the code. As a result, it is not only difficult for researchers to compare results, but it is often problematic even to determine whether they have correctly deployed their code on a given system.

Recently Helen He and Chris Ding, two researchers at LBNL, investigated this non-reproducibility phenomenon in a widely-used climate modeling code. They found that almost all of the numerical variation occurred in one inner product loop in the atmospheric data assimilation step, and in a similar operation in a large conjugate gradient calculation. He and Ding found that a straightforward solution was to employ double-double arithmetic (using the DDFUN package above) for these loops. This single change dramatically reduced the numerical variability of the entire application, permitting computer runs to be compared for much longer run times than before. Details of their work can be read in [21].

In retrospect, it is not clear that handling these sums in this manner is the best solution—there may be other ways to preserve meaningful numerical accuracy, while at the same time yielding reproducible results. Such phenomena deserve further study and refinement and are being investigated. But in the meantime, He and Ding's solution is a straightforward and effective means of dealing with this problem.

ΤΜΗΥΠ
Πανεπιστήμιο Πατρών

9. Computational Geometry and Grid Generation

Grid generation, contour mapping and several other computational geometry applications crucially rely on highly accurate arithmetic. This is because small numerical errors can lead to geometrically inconsistent results. Such difficulties are inherent in the mathematics of the formulas commonly used in such calculations and cannot be easily remedied. For example, William Kahan and Joseph Darcy have shown that small numerical errors in the computation of the point nearest to a given point on a line of intersection of two planes can result in the computed point being so far from either plane as to rule out the solution being correct for a reasonable perturbation of the original problem [23].

Two commonly used computational geometry operations are the orientation test and the incircle test. The orientation test attempts to unambiguously determine whether a point lies to the left of, to the right of, or on a line or plane defined by other points. In a similar way, an incircle test determines whether a point lies inside, outside, or on a circle defined by other points. Each of these tests is typically performed by evaluating the sign of a determinant that is expressed in terms of the coordinates of the points. If these coordinates are expressed as single or double precision floating-point numbers, roundoff error may lead to an incorrect result when the true determinant is near zero. In turn, this misinformation can lead an application to fail or produce incorrect results, as noted above.

To remedy such problems, Jonathan Shewchuk has produced a software package that performs "adaptive" floating-point arithmetic, which dynamically increases numeric precision until an unambiguous result is obtained. This software and some related information

$$
\begin{aligned}
\pi\sqrt{3} &= \frac{9}{32}\sum_{k=0}^{\infty}\frac{1}{64^k}\left(\frac{16}{6k+1}-\frac{8}{6k+2}-\frac{2}{6k+4}-\frac{1}{6k+5}\right)\\
\pi^2 &= \frac{1}{8}\sum_{k=0}^{\infty}\frac{1}{64^k}\left[\frac{144}{(6k+1)^2}-\frac{216}{(6k+2)^2}-\frac{72}{(6k+3)^2}-\frac{54}{(6k+4)^2}+\frac{9}{(6k+5)^2}\right]\\
\pi^2 &= \frac{2}{27}\sum_{k=0}^{\infty}\frac{1}{729^k}\left[\frac{243}{(12k+1)^2}-\frac{405}{(12k+2)^2}-\frac{81}{(12k+4)^2}-\frac{27}{(12k+5)^2}\right.\\
&\quad \left.-\frac{72}{(12k+6)^2}-\frac{9}{(12k+7)^2}-\frac{9}{(12k+8)^2}-\frac{5}{(12k+10)^2}+\frac{1}{(12k+11)^2}\right]\\
\log 3 &= \frac{1}{729}\sum_{k=0}^{\infty}\frac{1}{729^k}\left(\frac{729}{6k+1}+\frac{81}{6k+2}+\frac{81}{6k+3}+\frac{9}{6k+4}+\frac{9}{6k+5}+\frac{1}{6k+6}\right)\\
\pi\log 2 &= \frac{1}{256}\sum_{k=0}^{\infty}\frac{1}{4096^k}\left[\frac{4096}{(24k+1)^2}-\frac{8192}{(24k+2)^2}-\frac{26112}{(24k+3)^2}+\frac{15360}{(24k+4)^2}\right.\\
&\quad -\frac{1024}{(24k+5)^2}+\frac{9984}{(24k+6)^2}+\frac{11520}{(24k+8)^2}+\frac{2368}{(24k+9)^2}-\frac{512}{(24k+10)^2}\\
&\quad +\frac{768}{(24k+12)^2}-\frac{64}{(24k+13)^2}+\frac{408}{(24k+15)^2}+\frac{720}{(24k+16)^2}\\
&\quad \left.+\frac{16}{(24k+17)^2}+\frac{196}{(24k+18)^2}+\frac{60}{(24k+20)^2}-\frac{37}{(24k+21)^2}\right]\\
0 &\stackrel{?}{=} -2J_2-2J_3-2J_4+2J_{10}+2J_{11}+3J_{12}+3J_{13}+J_{14}-J_{15}-J_{16}\\
&\quad -J_{17}-J_{18}-J_{19}+J_{20}+J_{21}-J_{22}-J_{23}+2J_{25}
\end{aligned}
$$

Figure 2: Some new math identities found by high-precision computations

ΤΜΗΥΠ
ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ ΗΛΕΚΤΡΟΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ & ΠΛΗΡΟΦΟΡΙΚΗΣ
Πανεπιστήμιο Πατρών

40 A. M. Turing

tion of a process, but the logical control of the proposed calculator has been designed largely with such cases in veiw, and will have no difficulty on this score. The problem proposed is one which is well within the scope of the machine, and could be run off in a few minutes, assuming it was done as one of a sequence of similar problems. It is quite outside the scope of hand methods.

Problem 3 The solution of simultaneous linear equations. In this problem we are likely to be limited by the storage capacity of the machine. If the coefficients in the equations are essentially random we shall need to be able to store the whole matrix of coefficients and probably also at least one subsidiary matrix. If we have a storage capacity of 6400 numbers we cannot expect to be able to solve equations in more than about 50 unknowns. In practice, however, the majority of problems have very degenerate matrices and we do not need to store anything like as much. For instance problem (2) above can be transformed into one requiring the solution of linear simultaneous equations if we replace the continuum by a lattice. The coefficients in these equations are very systematic and mostly zero. In this problem we should be limited not by the storage required for the matrix of coefficients, but by that required for the solution or for the approximate solutions.

Problem 4 To calculate the radiation from the open end of a rectangular wave-guide. The complete polar diagram for the radiation could be calculated, together with the reflection coefficient for the end of the guide and interaction coefficients for the various modes; this would be done for any given wavelength and guide dimensions.

Problem 5 Given two matrices of degree less than 30 whose coefficients are polynomials of degree less than 10, the machine could multiply the matrices together, giving a result which is another matrix also having polynomial coefficients. This has important applications in the design of optical instruments.

Problem 6 Given a complicated electrical circuit and the characteristics of its components, the response to given input signals could be calculated. A standard code for the description of the components could easily be devised for this purpose, and also a code for describing connections. There is no need for the characteristics to be linear.

A. M. TURING'S ACE REPORT OF 1946
AND OTHER PAPERS
edited by
B. E. Carpenter and R. W. Doran

The MIT Press
Cambridge, Massachusetts
London, England
and
Tomash Publishers
Los Angeles/San Francisco

- 15 -

problem proposed is one which is well within the scope of the machine, and could be run off in a few minutes, assuming it was done as one of a sequence of similar problems. It is quite outside the scope of hand methods.

Problem 3.- The solution of simultaneous linear equations. In this problem we are likely to be limited by the storage capacity of the machine. If the coefficients in the equations are essentially random we shall need to be able to store the whole matrix of coefficients and probably also at least one subsidiary matrix. If we have a storage capacity of 6400 numbers we cannot expect to be able to solve equations in more than about 50 unknowns. In practice, however, the majority of problems have very degenerate matrices and we do not need to store anything like as much. For instance problem (2) above can be transformed into one requiring the solution of linear simultaneous equations if we replace the continuum by a lattice. The coefficients in these equations are very systematic and mostly zero. In this problem we should be limited not by the storage required for the matrix of coefficients, but by that required for the solution or for the approximate solutions.

Problem 4.- To calculate the radiation from the open end of a rectangular wave-guide. The complete polar diagram for the radiation could be calculated, together with the reflection coefficient for the end of the guide and interaction coefficients for the various modes; this would be done for any given wavelength and guide dimensions.

Problem 5. Given two matrices of degree less than 30 whose coefficients are polynomials of degree less than 10, the machine could multiply the matrices together, giving a result which is another matrix also having polynomial coefficients. This has important applications in the design of optical instruments.

Problem 6.- Given a complicated electrical circuit and the