UNIVERSITÄT KARLSRUHE
Institut für Angewandte Mathematik
Prof. Dr. Ulrich Kulisch

Kaiserstraße 12 · Postfach 6980
D-7500 Karlsruhe 1

Telefon : (0721) 608-4202
Sekretariat: (0721) 608-2680

May 10, 1988

David H o u g h
Sun Microsystems, Inc.
2550 Garcia Avenue

Mountain View, CA 94043
USA

Dear David:

Thank you for sending me a copy of the announcement for Kahan's series of lectures at Sun Microsystems. When your letter arrived I had gotten already several copies from diverse colleaques.

Let me make the following remarks:

1. What sometimes is called "Kulisch-Miranker-Arithmetic" does not require a "hidden Super-Accumulator"

What we require is that whenever a system or a programming environment provides arithmetic operations they should obey the rules of a semimorphism. Further: Operations with this quality should be provided for the four basic types real, interval, complex and complex interval as well as for vectors and matrices over these types (for one or more precisions, see Figure 1 of our book and its context).

In order to achieve this in case of matrix multiplication, for instance, an approximation of the dot product has to be provided which differs from the correct result only by one rounding. The long accumulator is one way to implement these dot products. In particular on computers of the /370 architecture, and others with a modest exponent range, it is a very convenient and efficient way.

./.

But there are a number of different techniques available for an efficient implementation of semimorphic operations in particular for matrices or complex numbers. You don't even find the long accumulator in our book!

2. Semimorphism is **not** an "over-axiomatization"

In FORTRAN-SC, for instance, there are about 1500 arithmetic operators available for and between different numerical data types. They all are defined by the four or five rules of a semimorphism, which can be written down on less than half a page.

Ringoids and vectoids are consequences of semimorphic operations. They are not axiomatic rules themselves. But they are most useful to derive further properties of arithmetic operations on computers (even in non semimorphic cases).

Properties of Computer Arithmetics can be discussed at lengths. Essentials are only those which contribute to the structure of the resulting space. Therefore ringoids and vectoids are studied.

3. Semimorphic operations are **not** "too slow"

The opposite is true. However, we talked already about the fact that it is practically impossible to realize a semimorphic dot product using a processor which performs arithmetic of the IEEE standard. (This may perhaps be the reason for Kahan's resistance against semimorphism).

A **software** implementation of the semimorphic dot product on the processor 68020, for instance, performs with 50% of the speed which is obtained if the same dot product is computed in the traditional sense using the **hardware** coprocessor 68881 performing IEEE arithmetic! The latter may result in an incorrect answer!

All existing hardware and software implementations of a semimorphic dot product perform faster than a corresponding computation in the traditional sense. No normalizations and roundings need to be executed after each multiplication and addition in case of the semimorphic dot product!

4. Why comment on the "IBM ACRITH package 1983"?

ACRITH came in three releases 1983 (linear techniques only), 1984 and 1985. Most of it was preliminary work necessary for the development and implementation of FORTRAN-SC (1987). Since more

./.

than 20 years we require that computer arithmetic and programming languages should not be considered as separated subjects. The language FORTRAN-SC (see the enclosed brochures) allows the declaration of functions with arbitrary result type, operator overloading and definition, as well as dynamic arrays. All numerical data types and operators mentioned in the first section are predefined in FORTRAN-SC. Beyond the operations FORTRAN-SC provides 22 elementary functions for each of the four basic data types real, interval, complex and complex interval in two precisions. PASCAL-SC is a corresponding extension of PASCAL.

I am pleased to see that Kahan requires operator overloading also. Does he know where and when these ideas originated?

With best regards
sincerely yours

Ulrich Kulisch

April 1988

PASCAL-SC

A Pascal Extension for Scientific Computation

The new extended PASCAL-SC System is the result of a long-term effort by a team of scientists* to produce a powerful tool for solving scientific problems. Due to its properties, PASCAL-SC is also an excellent educational system. The highlights of the new version are:

- *PASCAL-SC contains* **Standard** *PASCAL*
- *Powerful language extensions like functions with* **arbitrary result** *type and* **user-defined operators**
- **Module concept**
- **Dynamic arrays** *and* **slicing**
- **String** *concept*
- **Overloading** *of procedures, functions and operators*
- **Highly accurate** *arithmetic*
- *New data type* **dotprecision, exact scalar product**
- *Highly accurate expression handling with* **dot product expressions**
- **Highly accurate** *standard functions*
- *Screen oriented* **editor** *with* **syntax** *and* **semantic check**
- *Libraries for*
  Interval *arithmetic,* **Complex** *arithmetic,* **Complex Interval** *arithmetic and all* **Vector** *and* **Matrix** *operations for these mathematical spaces*
- *Modules for*
  Linear systems, eigenvalues *and* **eigenvectors, polynomials,** ...

The implementation of this second version of PASCAL-SC is completed but not yet available.
The first version is available on IBM/PC and ATARI ST. For further information see

G. Bohlender/L. Rall/Ch. Ullrich/J. Wolff v. Gudenberg:
PASCAL-SC, Wirkungsvoll programmieren, kontrolliert rechnen, BI Mannheim 1986
PASCAL-SC, A Computer Language for Scientific Computation, Academic Press, New York 1987

U. Kulisch (Editor):
PASCAL-SC, A PASCAL Extension for Scientific Computation,
Information Manual and Floppy Disks (IBM/PC), Wiley-Teubner, Stuttgart 1987
PASCAL-SC, A PASCAL Extension for Scientific Computation,
Information Manual and Floppy Disks (ATARI ST), Teubner, Stuttgart 1987

---

* Institute for Applied Mathematics, University of Karlsruhe, West Germany