ALGEBRAIC MANIPULATIONS ON COMPUTERS FOR LUNAR AND PLANETARY THEORIES

Jacques Henrard
Facultés Universitaires N.-D. de la Paix
Rempart de la vierge 8
B-5000 Namur Belgium

ABSTRACT. A new software to handle the large expressions involved in high accuracy analytical theories has been developped by H. Claes, M. Moons, J.M. Zune and J. Henrard. The main ideas on which this software is based are described here.

1. INTRODUCTION

The algebra involved in the analytical theories for high precision Celestial Mechanics is so huge that this kind of work was almost completely abandonned after the famous work of Delaunay and Brown on Lunar Theory.

With the advent of computers such work could start again and several new very accurate analytical theories have been and are still developped. I shall just mention the work done at the Bureau des Longitudes on planetary and lunar theories (Chapront-Touzé and Chapront 1983, G. Francou et al. 1983, P. Bretagnon 1984), the work done in Leningrad (Tupikova 1984, Taracevitch 1979) and in Namur (Henrard 1979, Standaert 1980, Moons 1986) although many more could be mentioned.

All these developments have been implemented with special purpose softwares which have not been compared or even in most cases been described in open litterature.

As we are testing a new version of our software (written by H. Claes, M. Moons, J.M. Zune and the author) which I believe is more general and more transportable than earlier versions, I think it is worth describing here some of the main ideas on which this software is based.

2. ALGEBRA OF POISSON'S SERIES

Typically, in problems of Celestial Mechanics, one would like to represent on the computer and manipulate functions of the type

J. Kovalevsky and V. A. Brumberg (eds.), Relativity in Celestial Mechanics and Astrometry, 59–62.

$$P = \sum_{i_1,\ldots,i_n} \sum_{j_1,\ldots,j_m} A^{i_1,\ldots,i_n}_{j_1,\ldots,j_m} x_1^{i_1} \ldots x_n^{i_n} \cos(j_1\phi_1+\ldots+j_m\phi_m)$$

which are called Poisson series. The number of trigonometric variables (m) and the number of polynomial variables (n) may vary from one problem to the other but are typically between 0 and 20 . The values of the exponents ($i_1,\ldots,i_n$) and the multipliers ($j_1,\ldots,j_m$) are integers and together form what is called the key of a term. The maximum absolute value of the components of the key may again vary from one problem to the other and even from one component to the other in the same problem. The number of terms in a series may reach several thousands but some series may have only a few terms.

It is thus essential to keep these options as open as possible. We choose a sheme by which at the beginning of an application, the number of variables (n , m) and the maximum absolute value for each component of the key is declared. This choice can be altered later on.

Each term can then be represented by its numerical coefficient and the value of its key packed within a few words. To this information should be added the information about the binary tree structure (see next section). A collection of such terms form a series.

3. BINARY TREE STRUCTURE

One of the more time consuming operations performed on Poisson's series is to find out whether a series contains a specific term and in that case its location within the series. For a software to be efficient, this search should be conducted with an algorithm close to the optimum.

The balanced tree algorithm (Knuth 1973) is such an algorithm which at the same time allows for easy insertion of new terms. To implement this algorithm we need to add to each term two pointers (the limbs of the tree) and a three-valued flag.

With the balanced tree algorithm the maximum number of comparisons needed to decide whether a given term is already present in a series of N terms, and if it is where is it, is close to $\log_2 N$.

4. SPACE ALLOCATION AND THE BLOCK MANIPULATOR

Series can be very large and it is not always possible (or convenient) to determine a priori how large they will be. We were often led with earlier versions of our software to situations where series would be too large to reside in the virtual memory.

To overcome this limitation, we cut the series in a collection of pages residing on a disk. When it is needed a page is brough dynamically into virtual memory. We create in a sense a super-virtual memory.

To handle this traffic between disk and virtual memory and to make it transparent to the user, J.M. Zune implemented a Block (of data) Manipulator.

Each series is considered as a block (of data). Several blocks can be bunched up into a file. A block is identified by an "identifier" and can be in one of two states. In the passive state, the block is one disk and the identifier contains the information needed to open the disk file. In the active state, the identifier contains the address of a dictionnary residing in core. The dictionnary itself contains the information about the location of each page of the data either in virtual memory or on disk.

When a new page has to be allocated in memory (and no page is available) the oldest page in memory is sent back on disk.

The Block Manipulator has already been used successfully for handling sparse matrices (Lescrenier 1985).

5. SERIES MANIPULATOR

The operations on series implemented in the series manipulator are the usual operations of the Algebra of Poisson Series : addition, multiplication, multiplication by a scalar and partial differentiation with respect to each variable. It is also possible to "read" or to create a series term by term allowing the user to construct its own special purpose functions.

The scheme is quite transparent. The user has only to know the identifier of a series to operate upon it. For instance, to perform the product of two series A and B and to accumulate the result into the series C , the user will write

```
CALL PROD(A,B,C,1.DO)
```

(the constant 1.DO indicates that the product A*B has to be multiplied by the scalar 1.) where A , B , C are FORTRAN variables containing the identifiers of the series.

The Block Manipulator and the Series Manipulator are implemented in FORTRAN with the exception of a few subroutines (coding, decoding and data transfer) written in machine language.

REFERENCES

BRETAGNON, P. 1984 : 'Accuracy of Long Term Planetary Theory', in Milankovitch and Climate, Berger et al. Eds., Reidel P.C.

CHAPRONT-TOUZE, M. and CHAPRONT, J. 1983 : 'The Lunar Ephemeris ELP 2000', Astron. Astrophys. **142**, 50-62.

FRANCOU, G., BERGEAL, L., CHAPRONT, J. and MORANDO, B. 1983 : 'Nouvelles Ephemerides du Soleil, de la Lune et des Planètes', Astron. Astrophys. **128**, 124-139.

HENRARD, J. 1979 : 'A New Solution to the Main Problem of Lunar Theory', Cel. Mech. **19**, 337-355.

KNUTH, D.E. 1973 : The Art of Computer Programming, Addison Wesley.

LESCRENIER, M. 1985 : 'Résolution d'un Système d'Equations Linéaires Structuré et de Très Grande Taille', Mémoire de Licence, Namur.

MOONS, M. 1986 : 'Libration of the Moon : Shape of the Earth and Motion of the Ecliptic Plane', these proceedings.

https://doi.org/10.1017/S0074180900147989 Published online by Cambridge University Press

STANDAERT, D. 1980 : 'Direct Perturbations of the Planets on the Moon's Motion', Cel. Mech. **22**, 357-369.

TARACEVITCH, C.V. 1979 : 'Algorithms in Celestial Mechanics', Institute of Theoretical Astronomy, Leningrad.

TUPIKOVA, I.V. 1984 : Nabludenia Isk. Sput. Zemli **23**, 249-261.

https://doi.org/10.1017/S0074180900147989 Published online by Cambridge University Press