```
AL,C4*C8* OK=(1+7*EP+33*EP**2)/3/EP**3
AL,C8**2*KDK=3*Z3/EP
AL,C7**2*KDK=3*Z3/EP
AL,PR2*C6**2=1/8/EP *3+49/48/EP**2+531/96/
AL PR2*C3 KDK=(1+8 EP+44*EP *2)/3/EP**3
AL,PR2 C2*QDQ=-(1+11*EP)/6/EP**2+Z3/EP
AL,PR2* 3*C7=-(1+11*EP)/6/EP**2
     AL, R2*C6*C7=-(1+11*EP)/6/EP**2+Z3/EP
     AL,PR2*C7**2=-(1+ 1*EP)/6/EP**2+Z3/EP
     AL,PR2*C1*C7=-(1+11*EP /6/EP**2
     AL,PR2*C2*C3=-(1+11*EP)/6/EP**2
     PRINT NOUT
     *YEP
AL,PR2*C4**2=-(1+27/2*EP+401/4*EP**2)/24/EP**
AL,PR2*C5**2=-(1+27/2* P+401/4*EP**2)/24/EP**
AL,PR2*C * DQ=(1+1 /2*E +199/4*EP *2)/6/EP**3
AL,PR2*C *QDQ=(1+19/2*EP+245/4*EP* 2)/12/EP
AL,PR2*C6*C3=(1+19/2 EP+245/4*EP**2)/12/EP**3
PRINT NOUT
*YEP
ID,PR2*C2*C4=-(1+10*EP) 3/EP**2
AL,PR2*C5*C7=-(1+10*EP)/3/EP**2
AL,PR2*( DC**2=1/8/EP**3 49/48/EP**2+531/96/EP
AL,PR2*C7*K K= *Z3/EP
AL,PR2*C2*K K=2*Z3/EP
A , R2*C1*C4=1/12/EP
AL,PR2*C5*C8=1/12/EP
AL,PR2* *C6=-(1+23/2*EP)/12/EP**2
AL,PR2*C8*QDQ=-(1+23/2*EP) .2/EP**2
AL,PR2*C3*C6=(1+8*E 44*EP**2)/6/EP**
AL,PR2*C3*QDQ=(1+8*EP+44*EP**2)/6/EP**3
PRINT NOUT
*YEP
ID,PR1*C C3=0
```

АНАЛИТИЧЕСКИЕ ВЫЧИСЛЕНИЯ НА ЭВМ И ИХ ПРИМЕНЕНИЕ В ТЕОРЕТИЧЕСКОЙ ФИЗИКЕ

```
ID,PR1*C7* DQ=-(1+11*EP)
AL,PR1*C2*C8=-(1+11*EP)/6/EP* 2
```

Д11-85-791

Труды Международного совещания по аналитическим вычислениям на ЭВМ и их применению в теоретической физике

17-20 сентября 1985 года, Дубна

International Conference on Computer Algebra and its Applications in Theoretical Physics

Dubna, 17-20 September, 1985

Дубна 1985

С О Д Е Р Ж А Н И Е

International Conference on Computer Algebra and its Applications in Theoretical Physics. Edited by N.N. Govorun, Joint Institute of Nuclear Research, Dubna 1985.

THE PROGRAM ORTOCARTAN FOR APPLICATIONS IN THE RELATIVITY THEORY

Andrzej Krasiński,
N. Copernicus Astronomical Center, Polish Academy of Sciences,
Bartycka 18, 00 716 Warszawa, Poland

1. A short history of the program

Almost every physicist working in the relativity theory had the painful experience of calculating the curvature tensors from a given metric tensor. The algorithm of the calculation is simple, but the chains of algebraic operations are long and in each step the expressions processed become increasingly complex before (if at all) they finally simplify. The idea that such calculations should be done by a computer was thus very natural and it occurred to several individuals at different times. The first algebraic program for use in relativity (called ALBERT) was implemented in mid-sixties /1/. Several more were created later; SHEEP /2/ is the most widely used today. We therefore first thought of having one of the existing programs installed in Warsaw. Soon we discovered that most of them would not run on our CDC Cyber 73. For a while we used CLAM /3/, but after noticing a bug in it we realized we will not avoid tampering with the program's code. And after a few trials we saw that it will be easier to write our own program than to squeeze out of our institutions the very modest fees required for copies of systems like REDUCE. We started the project in 1977 in a loose team which included M. Perkowski, this author, and, later, Z. Otwinowski. The program worked for the first time in 1977, but it took two more years of error-fixing and optimizing before we decided we can go public with it. The program is now available in 4 machine-implementations (see below) together with a set of documentation /4, 5, 6/. This note will describe shortly the essential features of the algorithms in the program. A user-oriented review /7/ and a more extended (but slightly outdated now) description of the algorithms /8/ were published elsewhere.

2. The calculatons performed by ORTOCARTAN

The details of the calculation are irrelevant from our point of view here (see refs. 4, 5 and 7), so only a general outline will be presented. The program takes as input a 4*4 matrix composed of functions of at most 4 variables. The functions are either unknown (to be later found as solutions of certain differential equations) or are explicit algebraic expressions (in the second case the program checks whether the matrix fulfills the equations). The inverse matrix is first found, then the components of the input matrix are differentiated with respect to all the variables, the resulting expressions are combined into polynomials with coefficients taken from the inverse matrix, the polynomials being further differentiated and multiplied among themselves, and finally the results of several such operations are added up. Consequently, the program is able to differentiate functional expressions, calculate determinants and inverse matrices, perform substitutions and simplify the resulting expressions algebraically. Within ORTOCARTAN these elementary operations are performed in a fixed sequence, the user having a chance to change only the input. The program is not interactive, nevertheless, substitutions can be defined in the input data and directed to arbitrary intermediate expressions, so that the user has effectively the same power at his disposal as in an interactive system. A few secondary programs give the user a direct access to such operations as inverting symbolic matrices of arbitrary rank, differentiation,algebraic simplification and performing series of substitutions. These functions were added to ORTOCARTAN as an "algebraic abacus" and cannot be combined into other programs unless one is able to do some programming in LISP, their base-language.

3. Objects processed by ORTOCARTAN.

These objects divide into 3 classes:

I. Numbers

They may be integers or rational numbers which are internally represented as dotted pairs, e.g. (1 . 2) for 1/2. Floating-point numbers are not allowed. All the elementary arithmetics of rational numbers is included. Exponentiations to fractional powers are also processed (though formally they belong to class III below). The program will simplify e.g. $\sqrt{27}$ to $3\sqrt{3}$ or $8^{1/6}$ to $2^{1/2}$, but may occasionally fail, e.g. it would not recognize that $\sqrt{2}*\sqrt{6} = 2*\sqrt{3}$. The substitutions may always be used to correct such failures.

II. Variables

These fall further into 4 categories: constants, coordinates (the independent variables for differentiaton), arbitrary functions and symbols for explicit expressions. The variables are assigned to classes by the user on input. Each variable carries a number with it, called its "priority" and stored on the variable´s property-list. The numbers are used to order sums and products uniquely, according to descending priorities. Numbers have by definition higher priorities than variables, lists which represent algebraic expressions have by definition lower priorities than atomic symbols. The priorities are assigned by the program (the user need not even know about this business) and do not change from run to run, unless the input data were changed by including new symbols or reordering them.

III. Algebraic expressions

They are represented internally in the prefix notation as lists, the first element of a list being the name of a function. However, on output all algebraic expressions are printed in the standard infix notation. Input syntax is also essentially infix, with only a few nuisances which ensure uniqueness of notation (e.g. obligatory use of symbols for multiplication and exponentiation). Of two lists, the higher priority is assigned to the one whose first element has the higher priority; if both first elements have equal priorities, the remaining parts of the lists are compared in the same way. Two expressions may have the same priority only if they are identical.

4. Algebraic simplification

ORTOCARTAN follows a fixed set of rules in simplifying algebraic expressions, e.g. it will expand each exponentiated sum unless the exponent is greater than 3, it will also expand each product containing a factor being a sum. However, each of the rules can be changed by the user through input commands or substitutions. A thorough simplification on all levels of the algebraic expressions is performed only in the user´s input. Later, the kind of simplification needed is known in advance. For example, after several expressions are added, only cancellations of terms and adding up of numerical factors should be performed, re-analysing the temrs is not necessary. Such limited simplifications are performed by 6 specialized functions with self-explanatory names: SMINUS, SPLUS, SLOG, STIMES, SEXPT and SEXP. In this way, the program avoids repeated simplifications.

The program cannot factorize polynomials and simplify rational functions. These operations require a different kind of algorithm which is rather long and complicated, and are rarely useful in the practice of the relativity theory. The user can however handle rational functions quite efficiently through substitutions.

5. Differentiation

ORTOCARTAN has no difficulties differentiating functions whose arguments are other functions. The user-command (DERIV X <a function>) is always understood as the total derivative with respect to X. For instance, if x, y, z, t are independent variables and f = f (g, h, x), g = g (u, t), h = h (x, y, z), $u = x^2 + y^2 + z^2$, then (DERIV X F) will be printed as $2\ X\ F_{,G}\ G_{,U} + F_{,H}\ H_{,X} + F_{,X}$, where commas denote partial derivatives. The arbitrary functions may be processed either with their arguments written out explicitly or with the arguments omitted, as in the example given. The arguments will be written out in subsequent calculations when the user writes them out on input. The dependencies of functions on their arguments are specified in a separate piece of input.

6. Substitutions

In addition to the replacement of a given sub-expression in a formula by another expression, ORTOCARTAN can execute more tricky operations. It can replace parts of sums and products, e.g. the substitution B + D = U will be performed in (A + B + C + D + E) producing (A + C + E + U). Such a substitution is equivalent to identifying a subset in a larger set and replacing it with another subset. The most recent version of ORTOCARTAN can also do some pattern-matching in the substitutions. The user can define certain variables (e.g. M and N) as MARKERS which can represent any expression. Then, (TAN M) = ((SIN M) / (COS M)) will mean that tan of any argument is to be replaced by sin/cos of the same argument. This is especially useful in processing truncated power series where by writing (E ** N) = 0 one ensures that only the terms linear in the parameter E will be kept while all higher powers will be discarded. The matching procedure recognizes the equality of functional forms of two expressions and assigns a value to each MARKER, the value being the sub-expression presently represented by the MARKER. The values are reassigned anew in each formula, so the same MARKER can represent different expressions in different formulae. The value is then inserted on the right-hand side of the equation in place of the MARKER, and the substitution is performed in the ordinary way. Each substitution is automatically followed by the appropriate algebraic simplification. The substitutions and the subsequent simplifications are performed in an economical way: the program begins to simplify a given expression only at the level where a replacement of a sub-expression actually occurred and proceeds from that level upwards. If no replacement occurred in an expression, then no simplification and no copying of list structures is performed.

7. Core-saving

At no point of the calculation is it necessary to reference all the functions in the program. Therefore the program is divided into 3 parts which are loaded into core separately. Part 1 contains the program for printing; it forms a separate overlay and during the initial phase of work (ca 80% of the time) is not active. Part 2 contains the programs for algebraic simplification and differentiation which must be active during the whole calculation, it also forms a separate overlay. Part 3 contains definitions of those functions which are needed only briefly (e.g. for rewriting the input into the internal code or for inverting the matrix). Part 2 is first loaded into core, and a small group of definitions from part 3 is read. The functions from part 3 do their work and their definitions are then erased. Further definitions from part 3 are read only when they are first needed. The results of the calculation are not printed immediately, but stored in their internal format on a disk file called PRINTS. When part 2 is ready with its work, the overlay with part 1 overwrites the core, and the file PRINTS becomes the input file. In this way, core-requirements are much reduced. Simple calculations can be done at 30 000 words or 300 kbytes of core, and 50 000 words were sufficient for the most complicated ones ever tried.

8. Availability

ORTOCARTAN is available in 4 implementations: 1. In Cambridge LISP on IBM 360/370 computers; 2. In University of Texas LISP 4.1 on CDC Cyber computers; 3. In SLISP/360 on IBM 360/370 and Siemens 4004 computers; 4. In LISP 1108 on UNIVAC 1100 computers. The IBM and UNIVAC versions were not updated since 1983, only the CDC version is constantly maintained. In particular, the pattern-matching is available only in the CDC version.

9. An example of application

Since the main application of ORTOCARTAN would not be intelligible for most readers of this note, we shall show how to use an "abacus" program to verify that 1/r where $r^2 = x^2 + y^2 + z^2$, is a solution of the Laplace equation. The input would be the following:

```
CALCULATE ((
     (LAPLACE EQUATION)
     (COORDINATES X Y Z)
     (SYMBOLS R = ((X ** 2 +  Y ** 2 + Z ** 2) ** (1 2)) )
     (OPERATION ((DERIV X X (1 / R)) + (DERIV Y Y (1 / R))
               + (DERIV Z Z (1 / R))) )
     (SUBSTITUTIONS (Z ** 2) = (R ** 2 - X ** 2 - Y ** 2) )
       ))
```

The printout would look as follows:

```
I ASSUME YOU REQUEST THE FOLLOWING EXPRESSION TO BE SIMPLIFIED
 -1        -1        -1
R  ,    + R  ,    + R  ,
    XX        YY        ZZ
HERE YOU HAVE IT
RESULT  = 0
       1
```

The subscripts at RESULT are useful when there is more than one OPERATION to be executed within a single call to CALCULATE. In this way, the same variables do not have to be defined more than once.

REFERENCES

1. J. G. Fletcher, R. W. Clemens, R. A. Matzner, K. S. Thorne, B. A. Zimmerman, Astrophys. J. Lett. 148, L91 (1967).
2. I. Frick, SHEEP user´s guide. USIP Report 77-15, University of Stockholm 1977.
3. R. A. d´Inverno, R. A. Russell-Clark, CLAM (CDC LISP algebraic manipulator). University of Southampton Report (not dated).
4. A. Krasiński, M. Perkowski, The system ORTOCARTAN - user´s manual. Third edition, University of Cologne 1983 (available as a computer printout or a magnetic tape recording).
5. A. Krasiński, M. Perkowski, Z. Otwinowski, The system ORTOCARTAN for analytic calculations - detailed description. Second edition, Warsaw 1984 (available as above).
6. A. Krasiński, The system ORTOCARTAN - user´s manual. Supplement to the second edition. Warsaw 1984 (available as above).
7. A. Krasiński, M. Perkowski, Gen. Rel. Grav. 13, 67 (1981).
8. A. Krasiński, SIGSAM Bull. 17, 12 (1983).