General Relativity and Gravitation, Vol. 13, No. 1, 1981

ORTOCARTAN—A New Computer Program for Analytic Calculations in General Relativity

ANDRZEJ KRASIŃSKI

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

and

MAREK PERKOWSKI

Institute of Automatic Control, Warsaw Technical University, Nowowiejska 15/19, 00-665 Warszawa, Poland

Received February 19, 1980

Abstract

This paper is meant to announce the appearance of a new computer program for symbolic calculations in general relativity, named ORTOCARTAN. The program calculates the curvature quantities from an orthonormal tetrad of forms representing the metric, both the tetrad and the coordinate components. The paper describes the main features of the program's input and output, and compares the program's speed to a few general-purpose systems, notably LAM, ALTRAN, FORMAC, REDUCE, and SYMBAL. A general overview of technical parameters of the program, and conceptual features of the algorithm is given.

§(1): *Introduction*

Although a few general-purpose formula-manipulation systems and a few tens of formula-manipulation programs specialized to narrow classes of problems are now available in the world (see, e.g., [1-3]), one still meets difficulties in obtaining them with the full set of documentation. The practical choice for a specific user is limited by the computer type to which he has access: most programs or systems run on one type only. Therefore every new program increases the supply and makes a wider distribution possible. Moreover, it is most useful to have a program together with a local group of experts which would be responsive to the requests of the users. For these reasons we decided to write a

0001-7701/81/0100-0067$03.00/0

new program for calculating the curvature tensors in general relativity. The program described in this article was produced by the authors, with occasional cooperation of Mr. Z. Otwinowski, in the years 1977–1979. Though directed towards calculating the curvature tensors, it contains several subprograms easily applicable to other kinds of calculation, and might become the core of a general-purpose system. The work to be done in this direction is much less than the work already done. At present, we find our program useful enough to be made generally available. It can be obtained free, together with the complete documentation: user's manual, listing, and detailed description of the code.

The fastest version now existing consists of a large code, written in the LISP-programming language, which is read by the LISP interpreter, and a small package of most frequently called subroutines, written in the LAP code which is read by the function READLAP. In the future we plan to produce a compiled version which would be at least five times faster.

In this article we want to give the readers enough material to weigh the value of our program against the other systems existing, and to encourage them to make use of it.

§(2): *Use and Application of the Program*

Our working assumption was that an ideal program requires, on the user's part, no previous experience with computers and as little introductory learning as possible. The introductory learning now consists of reading 52 standard pages of typewritten manuscript, of which only 27 suffice to start the work with simple problems. The manual mostly defines quite natural rules just to make them explicit, and we believe that our program is indeed very easy to use. This is partly at the cost of speed and core economy.

The program, named ORTOCARTAN, is readily implementable with the following requirements fulfilled:

(1) Language of implementation: LISP, the University of Texas version 4.1. Other versions of LISP may require minor modifications in the code. For ease of implementation the source version in pure LISP interpreter code is available.

(2) Computer: CDC Cyber 73 was our computer, but in fact any computer with LISP implemented will manage to run the program.

(3) Operating system: SCOPE 3.4.4. Under other operating systems slight modifications may be required.

ORTOCARTAN is directly applicable to calculating the curvature tensors from an orthonormal tetrad representation of the metric tensor. Its input data are the components of the tetrad forms. It calculates routinely the inverse tetrad of vectors, the Ricci rotation coefficients, and the tetrad components of the Riemann, Ricci, and Weyl tensors. On request of the user the program can calculate in addition the metric tensor (just to test the correctness of the tetrad), the inverse

metric, the Christoffel symbols (of the second kind only), the tetrad components of the Einstein tensor, and the coordinate components of the Riemann, Ricci, Weyl, and Einstein tensors, with any arbitrary positions of indices. The calculation may be stopped at any earlier stage, before the Weyl tensor is calculated. The user may introduce large amounts of algebraic substitutions at any stage of the calculation, thus making the results more tidy and/or concise.

With more effort from the user, the program ORTOCARTAN may be used to produce other specialized programs, e.g., for solving sets of linear algebraic equations, solving algebraic equations of higher degrees, writing the Dirac equation in different curved spaces, calculating curvature tensors in the coordinate formalism or in other tetrads, and so on. The program is equipped with fairly general subprograms for algebraic simplification, differentiation, printing the results in the mathematical format, inverting matrices, and a convenient scheme of the user-program interface. These are accessible by the usual methods of handling LISP programs. Such access requires a careful study of the description of our program (and, of course, the knowledge of the LISP language). The description is available on request, together with the listing of our program and the reference manual of LISP 4.1, from the authors of this article.

§(3): *Comparison with Other Systems*

The commonly accepted test problems for systems to be used in general relativity are: calculating the Einstein tensor for the Bondi-van der Burg-Metzner (BVM) metric [4], and the same calculation for the metric of Levy [5], which contains small corrections of the first order to a background metric. The first calculation tests the program's speed, the other one, its ability for handling non-trivial substitutions.

The BVM metric is

$$ds^2 = \{(V/r)^{1/2} e^{\beta} [dt + (r/V)\, dr]\}^2 - [(r/V)^{1/2} e^{\beta}\, dr]^2 - [re^{\gamma}(d\vartheta - U\, dt)]^2 - [r \sin \vartheta\, e^{-\gamma}\, d\varphi]^2 \quad (1)$$

where β, γ, U, and V are arbitrary functions of t, r, and ϑ, and the choice of the orthonormal tetrad of forms is unambiguously suggested by (1).

ORTOCARTAN calculated the tetrad components of the Einstein tensor for the metric (1), including printing all the intermediate results, in 536 sec at 53 248 (150 000 octal) words of core. This is to be compared [1] with 104 sec/400 kbyte for LAM, 255 sec/350 kbyte for ALTRAN, 162 sec/300 kbyte for FORMAC, 856 sec/500 kbyte for REDUCE, and more than 3600 sec for SAC-1, all on IBM 360/375, and with 35 sec/33 000 words for SYMBAL on CDC 6600. The program CLAM [6, 7] calculated the same quantities in 33 sec at 40 960 words on CDC Cyber 73.

The comparison of bare numbers gives the impression that ORTOCARTAN does not perform too well. It should be stressed, however, that the BVM metric is a very unfavorable case for the exterior calculus in the orthonormal tetrad, as the tetrad components of the curvature tensors happen to be much larger than the corresponding coordinate components. The most natural tetrad for the BVM metric is a null one, not the orthonormal. For this reason ORTOCARTAN had simply much more work to do with the BVM metric than did the systems which use the algorithm of the tensor calculus. A more informative comparison is, e.g., the static spherically symmetric metric in the standard Schwarzschild coordinates for which the size of the expressions processed is nearly the same in the coordinate calculus as in the orthonormal tetrad calculus. We compared this calculation by ORTOCARTAN with that by REDUCE version for CDC computers. Here, ORTOCARTAN performed the calculation in 42 sec at 27 136 (65 000 octal) words of core, to be compared with 110 sec at 22 528 words for REDUCE, on CDC Cyber 73.

The Levy metric is

$$ds^2 = (e^{2u} + \epsilon h_{00})\,dt^2 - e^{2k-2u}(dr^2 + dz^2) - r^2 e^{-2u}\,d\varphi^2 + \epsilon h_{ij}\,dx^i\,dx^j \quad (2)$$

where $i,j = 1,2,3, x^1 = r, x^2 = z, x^3 = \varphi$. The quantities u, k, h_{00} and all the h_{ij} are arbitrary functions of t, r, and z, while ϵ is a small parameter whose powers higher than the first are to be replaced by zero. Also, each time-differentiation is assumed to produce the factor ϵ.

The orthonormal tetrad of forms for the metric (2) we have taken to be

$$\begin{aligned} e^0 &= (e^u + \tfrac{1}{2}\,\epsilon e^{-u} h_{00})\,dt \\ e^1 &= e^{k-u}\,dr - \epsilon e^{u-k}(\tfrac{1}{2}\,h_{11}\,dr + h_{12}\,dz + h_{13}\,d\varphi) \\ e^2 &= e^{k-u}\,dz - \epsilon e^{u-k}(\tfrac{1}{2}\,h_{22}\,dz + h_{23}\,d\varphi) \\ e^3 &= (re^{-u} - \tfrac{1}{2}\,\epsilon e^u h_{33}/r)\,d\varphi \end{aligned} \quad (3)$$

Here the only inconvenience for the user is the necessity to replace the determinant of the base of forms calculated by the program to be

$$re^{2k-2u} - \tfrac{1}{2}\,\epsilon(-re^{2k-4u}h_{00} + rh_{11} + rh_{22} + e^{2k}h_{33}/r) \quad (4)$$

by the hand-calculated expression:

$$\{e^{2u-2k}/r + \tfrac{1}{2}\,\epsilon\,[-e^{-2k}h_{00}/r + (e^{4u-4k}/r)(h_{11} + h_{22}) + e^{4u-2k}h_{33}/r^3]\}^{-1} \quad (5)$$

equal to (4) up to terms linear in ϵ. This is necessary in order that the inverse tetrad of vectors, and all the further quantities, be expressible in terms of polynomials in ϵ, and not rational functions. Other substitutions are fairly obvious, as seen from the Appendix. The calculation was successfully performed even though ORTOCARTAN has no built-in handling of truncated power series, and the approximations had to be introduced ad hoc by the user.

To calculate the tetrad components of the Einstein tensor and print all the intermediate results for the metric (2) took ORTOCARTAN 1104 sec, to be compared with [1] 210 sec for LAM, 691 sec for ALTRAN, 321 sec for FORMAC, 234 sec for REDUCE, and 47 sec for SYMBAL. Memory occupied and computer used were in each case the same as before.

The complete input data for ORTOCARTAN in both cases, and a sample output for the BVM metric, are given in the Appendix.

§(4): *General Features of ORTOCARTAN*

The characteristics given below closely follow Table I of [1] so that ORTOCARTAN may be thoroughly compared with the six other systems reviewed in [1].

(1) Version available: 1979.

(2) Minimum core memory: 29 696 words (72 000 octal) to run small size problems on CDC Cyber 73. Of these 20 480 (50 000 octal) are necessary to load the LISP system.

(3) Core memory for the BVM metric: 53 248 (150 000 octal) words were used in the test, but 40 960 (120 000 octal) would suffice, with a slightly longer time of execution.

(4) Implementation language: LISP (U.T. version 4.1).

(5) Computer: any, with LISP implemented.

(6) Distribution: free; new users are warmly welcome.

(7) Distributed as: magnetic tape or punched cards.

(8) Maintenance: continuous, partly responsive to user's requests.

(9) Correction of system: easy, with complete regeneration.

(10) Dialog version: not existing. However, though working in the batch-mode, the system gives the user much opportunity to control the run of the calculation.

(11) Syntax: mixed FORTRAN-LISP-like, far trivialized, on input, normal mathematical on output.

(12) Declaration of variables: only spelling, with assignment to the classes: constants, coordinates, arbitrary functions, symbols of explicit expressions.

(13) Debugging facilities: some, rather simple. Those of LISP 4.1 are very powerful, but accessible only to more expert users.

(14) Output form: adjustable pagewidth, partial suppressing possible, no further choice.

(15) User's documentation: detailed manual intended for extreme nonexperts, with sample calls and prints.

(16) System documentation: full listing, detailed description of the code with directions for installation and maintenance.

(17) Exact arithmetics: infinite precision.

(18) Floating-point arithmetics: floating-point numbers illegal.

(19) Most general expression: only real expressions, no further limitations.

(20) Knowledge of elementary functions: most.

(21) Definition of differentiation rules: built-in for arbitrary functions, known elementary functions and symbols of explicit expressions, user-defined rules possible to introduce.

(22) Pattern matching: limited, only in user's substitution rules.

(23) Analysis of expressions: poor.

(24) Rational function algorithms: none, but ad hoc transformations of definite expressions introduced by the user may do the work in each case.

(25) Modernity of algorithms: fair.

(26) Handling of truncated power series: none built in, but possible with user-defined substitutions.

(27) Formalism for vectors and matrices: some for matrices, but not accessible to nonexpert user.

(28) Gamma-matrix algebra: no.

(29) Noncommutative algebra: no.

New users wishing to obtain a copy of ORTOCARTAN or its documentation should contact the authors.

§(5): *Essential Features of the Algorithms Used*

Here we describe the features of the algorithms used in ORTOCARTAN which we think might be of some interest.

The procedure for printing expressions in the mathematical format is recursive and can handle exponential expressions built upwards to an arbitrary height.

The simplifying subprogram is organized so that to every mathematical operation there is a specialized simplifying procedure to be called together with it. For instance, when a few expressions are to be added, then on the list of them the function SPLUS is called, which looks for the opportunity to make only those simplifications which are specific to addition (e.g., adding numerical coefficients of identical expressions, or deleting terms whose numerical coefficients summed up to zero). The terms added are assumed to be internally simplified before.

We devised a subroutine for substitutions which not only does not repeat simplifications, but also avoids unnecessary copying. It copies the expression in which a substitution is performed only upwards from the level where the substitution indeed occurred.

Determinants are calculated by making the array triangular, and so the algorithm is applicable also for large arrays. However, the algorithm of inverting matrices by making them triangular, inverting in the triangular form, and then constructing the searched inverse array by "detriangularization," which is known to be quite fast in numerical programming, poses here nontrivial problems with

factorization. Therefore, arrays are inverted by the ordinary rule of dividing the appropriate minors by the determinant.

The printing subprogram operates only at the end of the execution of a job, so it resides on a separate file as an overlay, to be loaded after the calculation. The simplifying procedure and the differentiating procedure remain in core during all the calculation. Most of the other procedures (more than half the code) are needed only in consecutive short intervals of the calculation. Therefore, they are read from the input file only when first needed, and removed from core immediately after the last use. In this way the system gradually destroys itself while it runs, so that it is not able to perform more than one calculation in each job, but on the other hand, thanks to this, at any moment less than half the code really resides in core. This has the obvious effect of increasing the space available for calculation.

§(6): *A Subjective Evaluation of the Program*

We shall first describe how the program was debugged. During two years, since summer 1977 till summer 1979, every change or correction of the code was followed by testing the program on the set of 20 to 30 metrics, all taken from the published literature, most of them giving some simple well-known result. In addition to the already-mentioned metrics of BVM [4] and Levy [5], and some unusual metrics meant only to test error messages or special facilities, they included the empty-space solution of Schwarzschild [8], the empty-space solutions with the Λ term of Schwarzschild and de Sitter (Reference 8, p. 184), types C and D of Krasiński [9], and that of Nariai [10, 11]; the conformally flat metrics of de Sitter (Reference 8, p. 184) and Robertson and Walker (Reference 8, p. 210); the dust-like solutions of Gödel [12] in the coordinate system of [13] and of Lanczos [14]; along with more general metrics, like the most general spherically symmetric metric in the standard coordinates [15], the same metric in the isotropic coordinates [11], the solution of Robinson and Trautman [16], and the solution of Kowalczyński [17] in a special coordinate system. Needless to say, in each case the correct result was finally obtained, and, just in passing, an error in the sign in the formula (17) of [16], suspected earlier, was confirmed. The solution of Kowalczyński [17] strained the abilities of our program to the extreme, being very complicated and requiring large amounts of a great variety of substitutions to keep the formulas of a manageable size. In fact, this solution forced us to introduce a few more facilities for the user's convenience. At present the Kerr metric in the ellipsoidal coordinates [18] is under testing, and it is even more straining as the orthonormal tetrad is very unsuitable here and the calculation deals with terrifying sets of rational functions. What makes us optimistic is the fact that our program can process formulas of that complexity and size (up to three pages of computer paper per one component of the Riemann tensor).

The program has no unique advantages as each of them can be found in at least one of the other well-known systems [1]. However, we think we have achieved a convenient combination of useful features which makes the program strongly user oriented. It has an easy input format, which seems to be its advantage over CLAM, and is comparable to other systems. It has a neat output which looks nearly like print in a book, which is its strong advantage over ALTRAN, CAMAL [19], and SYMBAL, and to some extent also over FORMAC and REDUCE, being comparable to LAM and CLAM (with the disadvantage, however, that no care is taken in our program to transfer the print to the next line so that the break would occur only + or −; instead the break may occur anywhere). It requires very little introductory learning and practice, which makes it comparable to CLAM, and seems to be an advantage over most other systems. It can process functions and functional expressions without simulating them by atomic variables, which is a strong advantage over ALTRAN, SYMBAL, and SCHOONSHIP [20], and makes it comparable to other systems. It is fairly completely documented, seemingly in contrast with most other systems (the lack of documentation may, however, be simulated by the custom of not answering letters, which seems to be common among computer people). The only real disadvantage is ORTOCARTAN's relative slowness (sometimes, however, it performs better than REDUCE), which we hope to cure in the future. The lack of dialog version we do not consider to be a disadvantage as calculations of such complexity as are met in general relativity usually cannot be comprehended readily by a person sitting at a console. One has to look carefully through the output to conceive a substitution or to solve an equation, and this is better done when one can sit down in a quiet place with the print in front of one. However, with the dialog mode of LISP 4.1, ORTOCARTAN can be quite easily (in a month's time or so) reworked into an interactive system.

Acknowledgments

We express our gratitude to Mr. Z. Otwinowski for his very active and valuable participation in optimizing our program and improving the algorithms. We acknowledge the contribution of Mr. M. Wardecki, whose simplifying procedure was the heart of the earliest version of ORTOCARTAN. We appreciate the great patience of Dr. J. Kowalczyński, whose complicated metric has been the major challenge for our program for more than one year. Finally, we thank Dr. M. Pindor and Dr. T. Hofmokl for enabling us to use the computer terminal of the Department of Physics, Warsaw University, on very convenient terms.

Appendix

Table I below shows the complete input data for the BVM metric, as written on cards.

Table I. Input Data OCTOCARTAN in the Case of the BVM Metric

```
ORTOCARTAN ((
   (BONDI METRIC)
   (COORDINATES T R THETA PHI)
   (FUNCTIONS  V (T R THETA)  U (T R THETA)  BET (T R THETA)
                GAM (T R THETA) )
   (EMATRIX  (V ** (1 2) * (EXP BET) / R ** (1 2))
             (R ** (1 2) * (EXP BET) / V ** (1 2))   0   0   0
             (R ** (1 2) * (EXP BET) / V ** (1 2))   0   0
             (- R * U * (EXP GAM))   0   (R * (EXP GAM))   0   0   0   0
             (R * (SIN THETA) * (EXP (- GAM)))   )
   (TENSORS EINSTEIN)
   (STOP AFTER EINSTEIN)
  ))
```

Table II. Sample Output for the BVM Metric

```
> RIE        = - 2 R EXP (- 2 GAM) SIN (THETA) BET,    - 2 R EXP (- 2
    I 3 1 3                                    R

> GAM) SIN (THETA) GAM,    + R  EXP (- 2 GAM) SIN (THETA) GAM,     - R
                  R                                  R

> EXP (- 2 GAM) SIN (THETA) GAM,      + 2 R  EXP (- 2 GAM) SIN (THETA)
                                R R

> BET,   GAM,
     R      R
```

Table II shows the coordinate component $R_{13\,13}$ of the Riemann tensor for the BVM metric, as an example of the output form of ORTOCARTAN. The markers > at the left point to the base-level lines.

Table III shows the complete input data, as written on cards, for ORTOCARTAN in the case of the Levy metric.

Most of the statements in the tables should be self-explanatory. The entry (TENSORS EINSTEIN) in Tables I and III means that the tetrad components of the Einstein tensor are to be calculated (they are calculated only when this special request is found in the data).

The trick used in the Levy metric is that the functions do not depend directly on the time coordinate x^0, but on $t = \epsilon x^0$, and thus each differentiation with respect to x^0 automatically produces the factor ϵ. The entry (METRIC SUBSTITUTIONS . . .) just tells the program that t is a symbol for the explicit expression $\epsilon \cdot x^0$. The entry (SIMPLIFY SUBSTITUTIONS . . .) specifies the substitutions to be made during the calculation. Each specification consists of the list of names of the quantities in which the equation that follows should be applied. The right-hand side of each equation is substituted for its left-hand side.

Table III. Input Data for the Levy Metric

```
ORTOCARTAN ((
   (LEVY METRIC UP TO EPSILON LINEAR)
   (CONSTANTS EPS)
   (COORDINATES X0 R Z PHI)
   (FUNCTIONS  U (T R Z)  K (T R Z)  H00 (T R Z)  H11 (T R Z)
               H12 (T R Z)  H13 (T R Z)  H22 (T R Z)  H23 (T R Z)
               H33 (T R Z) )
   (METRIC SUBSTITUTIONS T = (EPS * X0))
   (EMATRIX  ((EXP U) + (1 2) * EPS * (EXP (- U)) * H00)  0  0  0  0
             ((EXP (K - U)) - (1 2) * EPS * (EXP (U - K)) * H11)
             (- EPS * (EXP (U - K)) * H12)
             (- EPS * (EXP (U - K)) * H13)      0    0
             ((EXP (K - U)) - (1 2) * EPS * (EXP (U - K)) * H22)
             (- EPS * (EXP (U - K)) * H23)    0    0    0
             (R * (EXP (- U)) - (1 2) * EPS * (EXP U) * H33 / R)
   (TENSORS EINSTEIN)
   (SIMPLIFY SUBSTITUTIONS
        DETERMINANT IE AGAMMA RIEMANN    (EPS ** 2) = 0
        DETERMINANT IE AGAMMA RIEMANN    (EPS ** 3) = 0
        DETERMINANT IE AGAMMA    (EPS ** 4) = 0
        DETERMINANT    ACTUAL = (((EXP (2 * U - 2 * K)) / R
              - (1 2) * EPS * (EXP (- 2 * K)) * H00 / R
              + (1 2) * EPS * (EXP (4 * U - 4 * K)) * H11 / R
              + (1 2) * EPS * (EXP (4 * U - 4 * K)) * H22 / R
          + (1 2) * EPS * (EXP (4 * U - 2 * K)) * H33 / R ** 3) ** -1)
   )
   (STOP AFTER EINSTEIN)
        ))
```

In the last case, the left-hand side being ACTUAL, the program just replaces the old expression, whatever it was, with the right-hand side of the equation.

References

1. Cohen, H. I., Leringe, Ö., and Sundblad, Y. (1976). *Gen. Rel. Grav.*, **7**, 269.
2. Barton, D., and Fitch, J. P. (1972). *Rep. Progr. Phys.*, **35**, 235.
3. Krasiński, A., and Perkowski, M. (1978). *Postepy Astronomii*, **26**, 33 (in Polish).
4. Bondi, H., van der Burg, M., and Metzner, A. (1962). *Proc. R. Soc. London Ser. A*, **269**, 21.
5. Levy, H. (1968). *Proc. Cambridge Phil. Soc.*, **64**, 1081.
6. d'Inverno, R. A. (1975). *Gen. Rel. Grav.*, **6**, 567.
7. d'Inverno, R. A., and Russell-Clark, R. A. (1971). *Computer J.*, **17**, 229.
8. Rindler, W. (1977). *Essential Relativity; Special, General and Cosmological* (Springer Verlag, New York), p. 138.
9. Krasiński, A. (1975). *Acta Phys. Polon.*, **B6**, 223.
10. Nariai, H. (1950). *Sci. Rep. Tôhoku Univ.*, **34**, 160; (1951). *ibid.*, **35**, 62.
11. Krasiński, A., and Plebański J. (1978). "*n*-dimensional complex Riemann–Einstein spaces with $O(n-1, C)$ as the symmetry group." Preprint, in press in *Rep. Math. Phys.*
12. Gödel, K. (1949). *Rev. Mod. Phys.*, **21**, 447.
13. Krasiński, A. (1974). *Acta Phys. Polon.*, **B5**, 411.
14. Lanczos, K. (1924). *Z. Phys.*, **21**, 73.

15. Misner, C. W., Thorne, K. S., and Wheeler, J. A. (1973). *Gravitation* (W. H. Freeman, San Francisco), p. 843.
16. Robinson, I., and Trautman, A. (1962). *Proc. R. Soc. London Ser. A*, **265**, 463.
17. Kowalczyński, J. K. (1978). *Phys. Lett.*, **65A**, 269.
18. Krasiński, A. (1978). *Ann. Phys. (N. Y.)*, **112**, 22.
19. Wainwright, J. (1978). "CAMAL Programs for GRT: A User's Guide," University of Waterloo preprint.
20. Roskies, R. (1977). Private communication.