Algorithm 707
CONHYP: A Numerical Evaluator of the Confluent Hypergeometric Function for Complex Arguments of Large Magnitudes

MARK NARDIN
University of Michigan, Ann Arbor

W. F. PERGER and ATUL BHALLA
Michigan Technological University

A numerical evaluator for the confluent hypergeometric function for complex arguments with large magnitudes using a direct summation of Kummer's series is presented. Extended precision subroutines using large arrays to accumulate a single numerator and denominator are ultimately used in a single division to arrive at the final result. The accuracy has been verified through a variety of tests and they show the evaluator to be consistently accurate to 13 significant figures, and on rare occasion accurate to only 9 for magnitudes of the arguments ranging into the thousands in any quadrant in the complex plane. Because the evaluator automatically determines the number of significant figures of machine precision, and because it is written in FORTRAN 77, tests on various computers have shown the evaluator to provide consistently accurate results, making the evaluator very portable. The principal drawback is that, for certain arguments, the evaluator is slow; however, the evaluator remains valuable as a benchmark even in such cases.

Categories and Subject Descriptors: G.4 [**Mathematics of Computing**]: Mathematical Software; J.2 [**Computer Applications**]: Physical Sciences and Engineering

General Terms: Algorithms

Additional Key Words and Phrases: Confluent hypergeometric, special functions, evaluator

1. INTRODUCTION

CONHYP is FORTRAN program capable of calculating the confluent hypergeometric function (CHF) with complex arguments having magnitudes lim-

Authors' addresses: M. Nardin, Electrical Engineering Department, University of Michigan, Ann Arbor, MI; W. F. Perger and A. Bhalla, Electrical Engineering and Physics Departments, Michigan Technological University, Houghton, MI 49931.
This work was supported in part by NSF grant CCR-8823031 and by IBM grant RSP1057.

ited only by the computer speed and memory; arguments as large as one thousand have been successfully evaluated. The technique used is a straightforward evaluation of Kummer's series using specialized extended precision complex subroutines; a thorough description of this approach and the methods used to test the evaluator are described in a previous paper [1].

The confluent hypergeometric function is a solution of the differential equation

$$Z_1F_1''(A;B;Z) + (B - Z)_1F_1'(A;B;Z) - A_1F_1(A;B;Z) = 0 \qquad (1)$$

where A, B, and Z may all be complex. An exact solution of this equation is given by Kummer's series [2]. Previously, summing this series on a computer to obtain a solution when the magnitude of Z was large and either complex or negative proved difficult due to limited computer precision. For example, if A and B are equal to each other and $Z = 0 + 140i$, the series will grow to an intermediate value on the order of 10^{60}. But, as the terms cancel each other, the final solution of the series would have a magnitude of one. This means that at least 60 digits of precision are required just to get one digit of accuracy for the solution.

The typical solution to this problem has been to use an asymptotic expansion for large Z, such as that given by Slater [3]. When comparing the exact series expansion to the asymptotic expansion in the transition regions for Z, however, we found that there were many values of the arguments for which the CHF was not converging. It was found that "large Z" could range from 10 or less and to a thousand or more depending on the values of the arguments A and B.

2. REQUIREMENTS OF CONHYP

Because CONHYP uses arrays to store the intermediate sum of the series, the more memory available will translate to a wider range of arguments that can be handled. The user must be aware of the possibility that overwriting memory may occur if the input parameter IP (to be explained in more detail below) exceeds 777, the current value of the DIMENSION statements. A difficulty with the use of CONHYP is that it is not always possible to predict how much memory will be required. However, because the user can input as an option, IP, the number of array elements requested, experience has shown that rerunning the evaluator with two distinct values of this parameter provides a check on the integrity of the value returned. This point will become clear through the explanation of the sample case.

3. LIMITATIONS OF CONHYP

CONHYP has a limitation for which the user should be made aware. When the input arguments are received by CONHYP, any portion of any argument, e.g., the real part of argument B, that is too close to zero will be taken as zero. The definition of "too close to zero" is directly related to the intrinsic

machine precision, i.e., the number of bits in the mantissa, but, as an example, on the Sun SPARCStation 1, the number of available bits in the mantissa is 53, which translates to roughly 15 significant (decimal) figures. In this example, if the user attempts to input an argument with magnitude less than 10^{-12}, CONHYP will interpret that argument as zero (and print back a warning message to the user).

Note that CONHYP will apply this same definition to screen the argument B to check if it is a negative integer (in which case the CHF is not defined). Again using the Sun SPARCStation as an example, if $B = (-10.0D0, 1.0D - 14)$, CONHYP will treat the imaginary part of B as zero, determine that the real part of B is a negative integer, print an error message, and terminate execution.

Finally, the argument A is also screened to determine if it is a negative integer, in which case the CHF series terminates and consequenty CONHYP will terminate at the correct term in the series.

4. SAMPLE USE OF CONHYP

A sample case has been chosen which will serve to demonstrate the use of CONHYP. This case was selected because the result is one that can be readily compared with the result of evaluating an asymptotic expansion [1]. The ease of operation of the evaluator can be seen by examination of the sample driver for this case:

```
      PROGRAM SAMPLE
      COMPLEX*16 A, B, Z, CONHYP, CHF
      INTEGER LNCHF, IP
      DOUBLE PRECISION AR, AI
*
      OPEN (5, FILE = 'DATA', STATUS = 'OLD')
      READ (5, *) A
      WRITE (6, *) ' A = ', A
      READ (5, *) B
      WRITE (6, *) ' B = *, B
      READ (5, *) Z
      WRITE (6, *) ' Z = ', Z
      READ (5, *) LNCHF
      WRITE (6, *) ' LNCHF = ', LNCHF
      READ (5, *) IP
      WRITE (6, *) ' IP =  ', IP
      CHF = CONHYP(A, B, Z, LNCHF, IP)
      WRITE (6, 300)
      WRITE (6, 301) CHF
      AR = 2.31145634403113D-12
      AI = 1.96169649634905D-11
      WRITE (6, 302) AR, AI
  300 FORMAT (32X,'REAL PART', 16X, 'IMAG PART')
  301 FORMAT (1X, 'RESULT FROM CONHYP = ', 1P, 2D25.12)
  302 FORMAT (1X, 'EXPECTED RESULTS = ', 3X, 1P, 2D25.12)
      STOP
      END
```

The variables are defined as follows:

A = the first parameter of the confluent hypergeometric function, ${}_1F_1[A; B; Z]$.
B = the second parameter of ${}_1F_1[A; B; Z]$
Z = the argument of ${}_1F_1[A; B; Z]$
LNCHF = 0 (return ${}_1F_1[A; B; Z]$
= 1 (return the natural logarithm of ${}_1F_1[A; B; Z]$
IP = the number of array positions to be used

The last parameter warrants some discussion. For many of the special function applications, a value of IP = 7 is sufficient. Experience has shown, however, that the more difficult cases may require this parameter to be 100 or more, and it is not always trivial to predict which cases these might be. Consequently, a more pragmatic approach is to simply run the evaluator using a value of IP = 7, then increase IP, run the evaluator again and compare the returned results. However, if IP = 0, then the program will make a rough estimate of the value of IP required. Work is currently underway to improve the quality of this estimate.

The input file, DATA, for the sample case is

```
(-15.0D0, 55.0D0)
(20.0D0, 25.0D0)
(-100.0D0, 200.0D0)
0
10
```

The output for the sample case is (on the RS6000, excluding the echoing of the input data):

```
                          REAL PART              IMAG PART
RESULT FROM CONHYP =  2.311456344031E-12  -1.961696496349E-11
EXPECTED RESULT =     2.311456344031E-12  -1.961696496349E-11
```

5. PORTABILITY

As a test of the portability of **CONHYP**, the program was compiled and executed, for a variety of sample cases, on the IBM 4381, IBM 3090, IBM RS6000, Sun SPARCStation 1, and the Sun 3/60. Note that for each of these compilers, the COMPLEX*16 data type statements was accepted, with no changes in the source code required.

ACKNOWLEDGMENTS

The authors acknowledge the support of IBM Corporation for access to the IBM 3090 vector facility at the Los Angeles Scientific Center and the generous allocation of computing time on the IBM 4381 at Michigan Techno-

logical University. The discussions that we had with W. Cody and D. Amos contributed directly to the testing of the evaluator.

REFERENCES

1. NADIN, M., PERGER, W. F., AND BHALLA, A. *J. of Comput. Appl. Math. 39*, (1992), 193–200.
2. ABRAMOWITZ, M., AND STEGUN, I. A. *Handbook of Mathematical Functions With Formulas, Graphs, and Mathematical Tables*. U.S. Government Printing Office, Washington, D.C., 1972 390–413, 504–535, 546–553.
3. SLATER, L. J. *Confluent Hypergeometric Functions*. Cambridge University Press, London, 1960, 58–60.

Received April 1991; revised June 1991; accepted September 1991